



HOMER(v3.12, 6-8-2012)

Software for motif discovery and next generation sequencing analysis

HOMER (Hypergeometric Optimization of Motif EnRichment) is a suite of tools for Motif Discovery and next-gen sequencing analysis. It is a collection of command line programs for unix-style operating systems written in Perl and C++. HOMER was primarily written as a *de novo* motif discovery algorithm and is well suited for finding 8-20 bp motifs in large scale genomics data. HOMER contains many useful tools for analyzing ChIP-Seq, GRO-Seq, RNA-Seq, DNase-Seq, and numerous other types of functional genomics sequencing data sets.

News

(6-22-2012) (soon) New version v3.13 - Lots of updates, including better/updated annotation, including use of single annotation for each genome version [removal of 'separate' masked genomes] (can use hg18r for hg18 or add "-mask" to most commands and it sill still figure it out). **ALL** promoters/genomes must be updated to work with the new version. [Change Log](#)

(6-8-2012) New version v3.12 - Fixed bug in mergePeaks, other small bugs

(5-21-2012) New version v3.11 - mostly small fixes and bugs.

(3-22-2012) New version v3.10 - mostly small fixes.

(2-1-2012) New version fixes issue with HOMER-created bigWig files. UCSC Genome Browser updated their bigWig file processing on Feb 1st, which caused errors with bigWig files created previously with HOMER. The problem has been fixed, but you may need to remake your bigWig files with the new version for them to work.

(11-30-2011) Minor update (v3.8.1) for motif finding (parsing of small sequences) and support for variable region histograms (i.e. gene body)

(11-29-2011) New version v3.8 with bug fixes, added UCSC multi-wig hub support (makeMultiWigHub.pl), added support for Arabidopsis (tair10) and X. tropicalis (xenTro2), and added general support for genomes composed of scaffolds.

[Old News](#)

Program Download

[Download Page](#) - Get the latest version of HOMER

Supported Organisms: Human (hg17, hg18, hg19), Mouse (mm8, mm9), Rat (rn4), Frog (xenTro2), Zebrafish (danRer7), Drosophila (dm3), C elegans (ce6), S. cerevisiae (sacCer2), Arabidopsis (tair10), also works with custom genomes in FASTA format.

Using HOMER

Introduction to HOMER

[Installation](#)

[Configuration and Customization](#)

[Basic Description of HOMER](#)

[Index of HOMER Programs](#)

Duff-sponsored Tutorials

[Next-gen Sequencing Tutorial](#) [PDF]

Helps guide and explain how to use HOMER for ChIP-Seq/GRO-Seq/MNase-Seq/DNase-Seq/RNA-Seq etc. analysis

[Old tutorial: Analysis of ChIP-Seq experiments](#) (some of this is out-of-date)

[Motif Finding Tutorial](#) [PDF]

Instructions and advice for finding enriched regulatory elements from a set of genome positions, a list of genes, or raw FASTA files.

Doughnut Documentation (in shambles, coming soon - see pages above)

Primary Data Download

UCSC Visualization files (Updated for errors in UCSC processing 08/31/10)

[ChIP-Seq from the Glass, Murre, and Katzenellenbogen Labs](#)

HOMER Known Motifs - Genome-wide predictions and UCSC Track

These tracks display motif positions genome-wide for human and mouse. They are based on HOMER-motifs, and certainly miss many "weak" binding sites and incorrectly predict others. However, the predictions can still serve as a useful guide to where factors are likely to bind (if they're

expressed in the system you're studying).

[Mouse mm9 UCSC BigBed Track](#) (load as a custom track) - [[primary BED file](#)]

[Human hg18 UCSC BigBed Track](#) (load as a custom track) - [[primary BED file](#)]

[Human hg19 UCSC BigBed Track](#) (load as a custom track) - [[primary BED file](#)]

Data included in HOMER that may be useful for other purposes

[Custom Motif Matrices](#)

Credits

HOMER was developed primarily by Chris Benner, with significant contributions and suggestions by Sven Heinz, Kasey Hutt, Yin Lin, Gene Hsiao, Fernando Alcalde, Josh Stender, Amy Sullivan, Nathan Spann, Ivan Garcia-Bassets, Michael Lam, Michael Rehli, and many others. Primary supervision for the project was provided by Professors Christopher K. Glass and Shankar Subramaniam.

Development of HOMER was carried out in the [Glass Lab at UCSD](#).

For now, if you use HOMER in your research, please cite the following paper:

Heinz S, Benner C, Spann N, Bertolino E et al. Simple Combinations of Lineage-Determining Transcription Factors Prime cis-Regulatory Elements Required for Macrophage and B Cell Identities. Mol Cell 2010 May 28;38(4):576-589. PMID: [20513432](#)

Link to the raw data: [GSE21512](#)



Can't figure something out? Questions, comments, concerns, or other feedback:
cbenner@ucsd.edu



HOMER(v3.12, 6-8-2012)

Software for motif discovery and next generation sequencing analysis

HOMER (Hypergeometric Optimization of Motif EnRichment) is a suite of tools for Motif Discovery and next-gen sequencing analysis. It is a collection of command line programs for unix-style operating systems written in Perl and C++. HOMER was primarily written as a *de novo* motif discovery algorithm and is well suited for finding 8-20 bp motifs in large scale genomics data. HOMER contains many useful tools for analyzing ChIP-Seq, GRO-Seq, RNA-Seq, DNase-Seq, and numerous other types of functional genomics sequencing data sets.

News

(6-22-2012) (soon) New version v3.13 - Lots of updates, including better/updated annotation, including use of single annotation for each genome version [removal of 'separate' masked genomes] (can use hg18r for hg18 or add "-mask" to most commands and it sill still figure it out). **ALL** promoters/genomes must be updated to work with the new version. [Change Log](#)

(6-8-2012) New version v3.12 - Fixed bug in mergePeaks, other small bugs

(5-21-2012) New version v3.11 - mostly small fixes and bugs.

(3-22-2012) New version v3.10 - mostly small fixes.

(2-1-2012) New version fixes issue with HOMER-created bigWig files. UCSC Genome Browser updated their bigWig file processing on Feb 1st, which caused errors with bigWig files created previously with HOMER. The problem has been fixed, but you may need to remake your bigWig files with the new version for them to work.

(11-30-2011) Minor update (v3.8.1) for motif finding (parsing of small sequences) and support for variable region histograms (i.e. gene body)

(11-29-2011) New version v3.8 with bug fixes, added UCSC multi-wig hub support (makeMultiWigHub.pl), added support for Arabidopsis (tair10) and X. tropicalis (xenTro2), and added general support for genomes composed of scaffolds.

[Old News](#)

Program Download

[Download Page](#) - Get the latest version of HOMER

Supported Organisms: Human (hg17, hg18, hg19), Mouse (mm8, mm9), Rat (rn4), Frog (xenTro2), Zebrafish (danRer7), Drosophila (dm3), C elegans (ce6), S. cerevisiae (sacCer2), Arabidopsis (tair10), also works with custom genomes in FASTA format.

Using HOMER

Introduction to HOMER

[Installation](#)

[Configuration and Customization](#)

[Basic Description of HOMER](#)

[Index of HOMER Programs](#)

Duff-sponsored Tutorials

[Next-gen Sequencing Tutorial](#) [PDF]

Helps guide and explain how to use HOMER for ChIP-Seq/GRO-Seq/MNase-Seq/DNase-Seq/RNA-Seq etc. analysis

[Old tutorial: Analysis of ChIP-Seq experiments](#) (some of this is out-of-date)

[Motif Finding Tutorial](#) [PDF]

Instructions and advice for finding enriched regulatory elements from a set of genome positions, a list of genes, or raw FASTA files.

Doughnut Documentation (in shambles, coming soon - see pages above)

Primary Data Download

UCSC Visualization files (Updated for errors in UCSC processing 08/31/10)

[ChIP-Seq from the Glass, Murre, and Katzenellenbogen Labs](#)

HOMER Known Motifs - Genome-wide predictions and UCSC Track

These tracks display motif positions genome-wide for human and mouse. They are based on HOMER-motifs, and certainly miss many "weak" binding sites and incorrectly predict others. However, the predictions can still serve as a useful guide to where factors are likely to bind (if they're

expressed in the system you're studying).

[Mouse mm9 UCSC BigBed Track](#) (load as a custom track) - [[primary BED file](#)]

[Human hg18 UCSC BigBed Track](#) (load as a custom track) - [[primary BED file](#)]

[Human hg19 UCSC BigBed Track](#) (load as a custom track) - [[primary BED file](#)]

Data included in HOMER that may be useful for other purposes

[Custom Motif Matrices](#)

Credits

HOMER was developed primarily by Chris Benner, with significant contributions and suggestions by Sven Heinz, Kasey Hutt, Yin Lin, Gene Hsiao, Fernando Alcalde, Josh Stender, Amy Sullivan, Nathan Spann, Ivan Garcia-Bassets, Michael Lam, Michael Rehli, and many others. Primary supervision for the project was provided by Professors Christopher K. Glass and Shankar Subramaniam.

Development of HOMER was carried out in the [Glass Lab at UCSD](#).

For now, if you use HOMER in your research, please cite the following paper:

Heinz S, Benner C, Spann N, Bertolino E et al. Simple Combinations of Lineage-Determining Transcription Factors Prime cis-Regulatory Elements Required for Macrophage and B Cell Identities. Mol Cell 2010 May 28;38(4):576-589. PMID: [20513432](#)

Link to the raw data: [GSE21512](#)



Can't figure something out? Questions, comments, concerns, or other feedback:
cbenner@ucsd.edu



HOMER

Software for motif discovery and ChIP-Seq analysis

Change Log:

Major Bugs/Errors are shown in red

Major Additions are shown in blue

Minor stuff or upgrades that won't likely have a big impact are shown in black

Next Version (soon)

HOMER v3.13 (6/22/12)

- Update to annotation system, better annotation for ncRNA, more accurate UTR boundaries (some were off by a bp or two)
- No more separate "masked" genomes - **homerTools extract** now has option "-mask" that will replace soft masked sequences (e.g. lowercase letters) with N. Programs like **findMotifsGenome.pl** and **annotatePeaks.pl** now have option "-mask" or will interpret hg18r as shorthand for "hg18 ... -mask"
- In **makeTagDirectory** Fixed errors with CIGAR parsing with SAM files - improved RNA-Seq bedGraph visualization at splice junctions, use "-fragLength given" with makeUCSCfile or makeBigWig.pl etc.
- Fixed a bunch of other stuff I can't remember...

HOMER v3.12 (6/8/12)

- Bugs fixed and small options added.
- Fixed problem with **mergePeaks** crashing

HOMER v3.11 (5/21/12)

- Lots of bugs fixed and small options added.
- Fixed inconsistencies with treating BED files as zero-indexed
- **annotatePeaks.pl** maintains the peak order when making heatmaps
- Fixed IP efficiency calculation for "-style histone" or "-region" in **findPeaks** - Fixed bug with multi-processor support from some linux distros (should crash anymore) with motif finding
- Fixed bug in **getDifferentialPeaks** with -size parameter

HOMER v3.10 (3/22/12)

- Lots of bugs fixed and small options added.
- **annotatePeaks.pl** now works with bedGraph files in a manner similar to tag directories (option "-bedGraph <file>").
- Added "-precision <1|2|3>" option to **makeTagDirectory** to print values if format 1.0 or 1.00 or 1.000 (useful if normalizing or using fractional tag counts)
- Fixed **annotatePeaks.pl** "-center <motif>" option when using unbalanced peak size (i.e. "-size -200,50").
- Added program **removeOutOfBoundsReads.pl** to remove reads that are out of bounds, causing problems for UCSC (some alignment programs have a tendency to do this)
- several new options added to **compareMotifs.pl** (scale heights of logos to information content "-bits", skip similar matching/visualization "-basic", etc.)

HOMER v3.9 (2/1/12)

- Fixed bigWig/hub creation to work with updates at UCSC (**makeBigWig.pl/makeMultiWigHub.pl**) **makeBigWig.pl** now requires that you enter the genome as an argument, and when making bigWigs with **makeUCSCfile**, you need to specify the chrom.sizes file (makeBigWig.pl and makeMultiWigHub.pl take care of this automatically)
- **annotatePeaks.pl** can now process bedGraph files just like a tag directory by using the "-bedGraph" option (i.e. make histograms, calculate read density, etc.)

HOMER v3.8.2 (1/6/12)

- Fixed issue with **findPeaks** using too much memory. Added option "-minTagThreshold <#>" that controls the smallest peaks to consider. By default this is set at the uniform density (i.e. expected tags per peak region given a uniform tag coverage)
- Fixed bug with **findMotifsGenome.pl/findMotifs.pl/homer2** "-cache <#>" that caused a crash if too large of a cache was specified.
- Fixed bug with 5' adapter trimming and added the option to trim adapter sequence while allowing mismatches with **homerTools**.

HOMER v3.8.1 (11/30/11)

- (3.8.1) Fixed sequence parsing issues for short sequences for *de novo* motif finding, added support for % based histograms over variable length regions (i.e. gene bodies)
- Added support for [UCSC Hub Creation](#) (**makeMultiWigHub.pl**)
- Modified general routines to work better with large numbers of chromosomes (i.e. genomes composed of scaffolds like X. tropicalis)
- Added support for Arabidopsis (tair10) and X. tropicalis (xenTro2)
- Updated annotations
- Fixed bug with CpG/GC% calculations in **annotatePeaks.pl** when dealing with variable length peaks/regions
- "-forceBED" option is now standard for parsing BED files of sequence read alignments (**makeTagDirectory**), new option "-force5th" to use 5th column of BED file as read count

- Fixed issue with auto-detecting BAM files

HOMER v3.7 (11/02/11)

- Added q-value/FDR calculations to de novo motif finding. Unfortunately, due to the complexity behind the *de novo* algorithm, the only way to do this is to calculate it empirically by randomizing the data and recalculating motifs. As a result, it takes a long time to calculate FDR values (option -fdr <#>, # is the number of randomizations, **findMotifsGenome.pl**)
- Fixed bug in findMotifsGenome.pl causing the option "-size given" to crash.
- Fixed calculation of peak overlap significance in **mergePeaks**, in cases where "-d #" is used.

HOMER v3.6 (10/12/11)

- Fixed bugs in **mergePeaks**, v3.5 would crash in some extreme cases. Fixed significance calculations for peak overlaps (again) to deal with nearby peaks from the same file.
- Changed the way "-matrix <filename>" works with **mergePeaks/annotatePeaks.pl**.
- Added q-values (Benjamini multiple hypothesis testing corrections) to know motif finding (**findMotifs.pl/findMotifsGenome.pl**)
- fixed bug in findMotifs.pl that causes problems with custom promoter sets and failure to output de novo motifs

HOMER v3.5 (10/06/11)

- Changed how findPeaks interprets genomes size: When using "-gsize <#>", use the number of mappable bases, not 2x the number as previously suggested.
- Added "-nfr" flag to **findPeaks** to help find nucleosome free regions in histone modification data (works best with MNase datasets)
- Fixed bug in **mergePeaks** - when merging peaks when several peaks in the same file are within range (i.e. merging transcription factor peaks within 100000 bp), latest version would sometimes crash.
- **mergePeaks** now outputs the total number of peaks that contribute to each peak in the 8th column.
- fixed bug with **findPeaks** - normally findPeaks uses the average coverage of an experiment as the minimum when considering the enrichment over input signal - this value was divided by 2 in the previous versions. Weak ChIP-seq experiments will likely see less peaks in the output file now when using input to filter the experiment (but the peaks you do get back will be better)

HOMER v3.4 (09/30/11)

- Fixed problem with motif statistic reporting for de novo motifs (**findMotifsGenome.pl**, **findMotifs.pl**, **homer2**). Previous version calculated motif percentages using sequence that had more significant motifs masked,

causing some instances to be missed. HOMER now reports the % of sequences containing the sequence using the original sequences. The motifs found by the algorithm themselves are unaffected, just the reported statistics have changed. This change on reflects statistics found in the homerResults.html file. Other tasks, such as searching for motifs, are unaffected.

- Fixed GTF format parsing when the file is not sorted properly (caused some annotation to be dropped from consideration if the file wasn't presorted).
- Chuck facts must now be installed separately using the **configureHomer.pl** program ("-getFacts")

HOMER v3.3 (09/28/11)

- **makeTagDirectory** will now take gzipped (*.gz), zipped (*.zip), bzip2(*.bz2), and bam (*.bam) files directly - no need to decompress them. samtools needs to be installed and available on the executable path for homer to work with *.bam files
- Fixed total mapped tag normalization when running **annotatePeaks.pl** or **analyzeRNA.pl** with options like "-pc <#>" that limit the number of reads considered at a specific position. These programs now reference the "tagCountDistribution.txt" file in the tag directory to properly scale the total number of tags to be compatible with the limiting function.
- fixed problem with **mergePeaks** - peaks from the same file within the distance ("-d #") are also merged.
- **mergePeaks** now outputs how many peaks were merged in each output peak (useful when merging over large distances)
- mergePeaks can now work with a single peak file to merge peaks found within a given distance of one another. Can also be used to filter a peak file for peaks found within a given region.
- Slight modifications made to the calculations for peak overlap significance in **mergePeaks** (When using "-matrix ..." option). Total coverage of the peaks is not calculated to adjust numbers when peaks are overlapping (not done before, might have been a problem for a peak file with many overlapping peaks)
- Option for excluding Chuck Facts added ("-nofacts") to **findMotifs.pl**, **findMotifsGenome.pl**, and **compareMotifs.pl**. To permanently remove them, remove the file in homer/data/misc/
- **compareMotifs.pl**, which is used by the *de novo* motif finding programs to determine the similarity between *de novo* and known motifs, has been modified such that Pearson's correlation is the default for comparing motif matrices. When comparing matrices of different length they are elongated with 0.25 frequencies. Overall an improvement based on what a human would "expect".

HOMER v3.2 (08/11/11)

- Fixed problem with peak annotation (if peaks are overlapping some would not be annotated)
- Lots of small things, such as file format detection, etc.

HOMER v3.1 (05/25/11)

- Added "easy" Custom Genome support. Instead of specifying a "genome" such as "hg18" for programs such as **findMotifsGenome.pl** or **makeTagDirectory**, you can specify the path to the genomic FASTA files (either a single file or a directory with FASTA files named for each chromosome).
- No longer need a "reference file" for preparing the genome, now it will just randomly determine regions if one is not provided or cannot be found (i.e. if you have a custom genome).
- **"-chopify"** option will automatically split up large background regions to the size of the target regions. So if you're too lazy to explicitly select regions as background, you can provide a large FASTA file (with **findMotifs.pl** in FASTA mode) or a large region (i.e. a whole chromosome in **findMotifsGenome.pl**) and the "-chopify" option will tell HOMER to chop up the region into smaller, target-sized, chunks.
- **"-rna"** option for motif finding to output RNA style motifs and automatically searches only the + strand.
- Fixed SAM format auto detection. Works better now (If you have BAM, use [samtools](#) to convert BAM formatted files to SAM)
- Fixed problem with automatic genome size detection with smaller genomes in **findPeaks**.

HOMER v3.0 (05/09/11)

- New motif finding program **homer2**
New masking strategy increases sensitivity for finding co-regulated motifs
Autonormalization helps reduce problems caused by sequence composition bias
- Added zebrafish (danRer7) and yeast (sacCer2) support.
- Added GRO-Seq analysis routines (**findPeaks**, **analyzeRNA.pl**)
- Added read normalization and bundled QC into **makeTagDirectory**
- Bunch of other stuff...

HOMER v2.7 (12/14/10)

- added support for parsing alignment in SAM format. (If you have BAM, use [samtools](#) to convert BAM formatted files to SAM)

HOMER v2.6 (10/21/10)

- batchAnnotatePeaks.pl program for making histograms across multiple peak files.
- findPeaks now has "-region" mode for identifying variable-length regions of signal enrichment.
- tagDir2bed.pl script available to easily export tag data into bed file format
- findMotifsGenome.pl and annotatePeaks.pl are now BED file compliant! Since

everyone uses those, I guess Chuck can too. But his still prefers position/peak files! Not all subprograms work with BED files, so you may still need to use bed2pos.pl to switch formats to do certain tasks.

HOMER v2.5 (10/11/10)

- Fixed errors in mergePeaks script (gave negative chromosome coordinates before if compiled on certain systems)
- No longer keep around temporary/sequence files from the motif finding
- Added wikipathways to GO analysis
- Fixed bug in clonal filtering with peak finding (affects highly clonal experiments/lower organism analysis)
- getDistalPeaks.pl program now finds intra/inter-genic peaks
- updated promoter sets to use refseq
- New error checking for FASTA file input (previously HOMER required only ACGTN characters).

HOMER v2.4 (08/30/10)

- New annotation system (2 tier - promoter/exon/intron/intergenic and detailed i.e. repeats, etc.). Also, fixed a slight bug in annotation priority script.
- Genomic Gene annotations standardized on refGene.txt file from UCSC. (includes miRNA and some other non-coding RNAs)
- New **GenomeOntology.pl** program - significance association calculations with genomic annotations like repeats, other peak files, etc.
 - Annotations for exons/introns/promoters etc., repeats, gene deserts/gene rich regions, GO terms, peaks from published data.
 - Works with both peaks and with tag directories (for peak independent analysis)
 - Added as an
- New mergePeaks program - rewritten in C++ with added functionality
- Added strand specific tag counting to annotatePeaks.pl ("-strand" option)
- Added **analyzeRNA.pl** program to compute gene expression levels from RNA-Seq data. Will also work for repeats, but need a lot of memory.
- Added tts mode (to go along with tss mode) in annotatePeaks.pl.
- Added "motifFindingParameters.txt" file that remembers which parameters were used during motif finding
- Added "bias motifs" to known libraries help users identify motifs that are likely come from sequence bias or are just garbage.
- Added strand support to makeUCSCfile for UCSC genome browser tag pile-ups
- fixed problem with tag extensions exceeding the size of the chromosomes.
- Added MSigDB links and annotations to GO analysis.
- Updated all annotations, accessions as of 8/30/2010.

HOMER v2.3

- Fixed error in histogram creation (stupid round-off error shifted the location of

some peaks)

- assignGenomeAnnotation program rewritten in C++ for a dramatic speed up.
- Fixed problem of findMotifsGenome.pl crashing if non-unique peak ids are used
- Fixed redundant ID creation when generating background sequences (minor issue)
- Added peak file checking (i.e. for non-redundant IDs)
- Added more known motifs
- Bunch of other minor stuff I can't remember...



Can't figure something out? Questions, comments, concerns, or other feedback:
cbenner@ucsd.edu



HOMER

Software for motif discovery and ChIP-Seq analysis

Old News:

(10-12-2011) The new version (v3.5/v3.6) fixes errors in the mergePeaks program that arise when merging peaks over large distances, also added "-nfr" option to findPeaks to identify nucleosome free regions more effectively. Cleaned up peak overlap significance calculations, and added Benjamini multiple-hypothesis testing corrections to various programs such as known motif enrichment.

(09-30-2011) The new version (v3.4) - fixes error in motif percentages reported after de novo motif finding. Previous version calculated motif percentages using sequence with more significant motifs masked, causing some instances to be missed. HOMER now reports the % of sequences containing the sequence using the original sequences. The motifs found by the algorithm themselves are unaffected, just the reported statistics have changed.

(05-25-2011) The new version (v3.1) is much more "custom" genome friendly - allows you to directly specify genome sequence files instead of "configuring" genomes.

(05-20-2011) Added Yeast (sacCer2) and zebrafish (danRer7). Also restructured accession number/gene annotation information such that it is tied to the "promoters" of a given organism. This way you don't have to download all of it with the main program.

(05-09-2011) First major update of HOMER in a while (v3.0). Most importantly, the motif finding has been rewritten to improve accuracy and performance, with multiprocessor support. Pretty much everything else has also been worked on and improved. GC-normalization, GRO-Seq (nascent RNA) analysis, RNA motif analysis, and a number of other routines have been added.

Can't figure something out? Questions, comments, concerns, or other feedback:
cbenner@ucsd.edu





HOMER(v3.12, 6-8-2012)

Software for motif discovery and next generation sequencing analysis

HOMER (Hypergeometric Optimization of Motif EnRichment) is a suite of tools for Motif Discovery and ChIP-Seq analysis. It is a collection of command line programs for unix-style operating systems written in mostly perl and c++. Homer was primarily written as a *de novo* motif discovery algorithm that is well suited for finding 8-12 bp motifs in large scale genomics data.

Hardware Requirements (recommended): 2+ Gb memory (4-8+ Gb), 10+ Gb Hard Drive space (50+ Gb)

Software Requirements: Unix compatible OS (or cygwin), perl, gcc, make, wget, ghostscript, weblogo, blat

Full Program Download

Download file:

[configureHomer.pl](#) v3.12 (6-8-2012) [GPLv3]

Use this file for downloading and updating the full HOMER program and associated data. For sequencing data, there are a bunch of annotation files, etc., for each version of the genome, and this script helps manage those files. If you don't want to deal with configuring HOMER's next-gen sequencing functionality, but want to try it for motif finding, see below.

Instructions: Download [configureHomer.pl](#) (right click and select "save link as") and place in a directory dedicated for HOMER (such as homer/).

Run the script by typing "perl configureHomer.pl" - consult the links below for more information. Additional software and configuration will be required the first time you install HOMER (see Installation).

[Click for detailed installation instructions](#)

To upgrade, change your directory to where you installed HOMER, and type:

```
perl configureHomer.pl -update
```

- or -

```
perl configureHomer.pl -install homer (this is good for forcing the software to reinstall - preferred if you think there is something wrong)
```

If something appears to be wrong, redownload the "configureHomer.pl" script and use "perl configureHomer.pl -install homer" - this fixes a majority of issues.

Program Components

[homer2](#) program - key executable for HOMER motif discovery. (This archive actually contains all of the c++ executables, not just homer2). Unzip in the desired directory and simply type "make" to compile the program.

Update Information

[Change Log](#) - Short description of recent changes

[update.txt](#) - Current HOMER configuration list (Currently support human hg17/hg18/hg19, mouse mm8/mm9, rat rn4, X. tropicalis xenTro2, drosophila dm3, and C. elegans ce6, Zebrafish danRer7, yeast sacCer2, Arabidopsis tair10)



Can't figure something out? Questions, comments, concerns, or other feedback:
cbenner@ucsd.edu



HOMER

Software for motif discovery and next-gen sequencing analysis

Installation Guide:

Basic requirements

Homer is computationally intensive collection of programs. The following are minimum hardware requirements for running promoter analysis (ChIP-Seq in parenthesis).

- Unix-style operating system (UNIX/LINUX/Mac/Cygwin)
- 1 Gb of RAM (2-4 Gb)
- 1 Gb of Hard Drive Space (>10Gb)

Homer is a collection of perl and c++ programs designed for execution in a UNIX environment. Any Unix/Linux or Mac OS X system should have no trouble running Homer. Homer may also be run on Windows using Cygwin Linux emulation software. The following basic software must be available on your system.

- perl
- GNU make utility
- GCC C/C++ compiler
- wget (useful Unix utility)
- Basic utilities such as zip/unzip/gzip/gunzip/cut/tar
- [3rd party software](#) for sequence logos: seqlogo, gs (ghostscript), see below for details

If you are running Mac OS X, you will need to install “Developer Tools” from Apple if not done so already. In addition, Mac users should install “wget” from mac ports (<http://wget.darwinports.com/>) or at <http://www.statusq.org/archives/2008/07/30/1954/>.

If you are running Windows follow these steps for installing [Cygwin](#):

1. Download the cygwin install program from <http://www.cygwin.com/>
2. Run the install program, and when prompted to choose which packages to install, make sure to install the following:
 - GCC compiler, gcc-core, gcc-g++ (in Devel)
 - make (in Devel)
 - perl (in Languages)
 - zip/unzip (in Archive)
 - mingw64 (in Devel, for pthread support)
 - wget (in Web) – very useful for downloading things...

- ghostscript (in Images)

While running Homer is designed to be as simple as possible, some basic knowledge of UNIX commands is required. If you are new to UNIX, try googling “UNIX tutorial” for a more formal introduction.

Homer also requires a couple [3rd party software packages](#) to visualize motif logos.

Installing the basic HOMER software

HOMER will be installed in the same directory that you place the **configureHomer.pl** program. **configureHomer.pl** will attempt to check for required utilities and alert you to missing programs.

1. For the latest version of Homer, go to <http://biowhat.ucsd.edu/homer/>.
2. Download the “[configureHomer.pl](#)” script and place it in a directory where you would like homer to be installed (i.e. /home/chucknorris/homer/).
3. Run the configureHomer.pl script to install homer.
 - i.e. **perl /Users/chucknorris/homer/configureHomer.pl –install**
 - NOTE: Cygwin users may need to rename the files in homer/bin/ to remove the “.exe” (i.e. “homer.exe” to “homer”)
 - NOTE: If running SunOS or other proprietary UNIX environments, you may need to add the option “-sun” so that “gmake” is used instead of “make”.
4. Add the homer/bin directory to your executable path. For example, edit your **~/.bash_profile** file to include the line:
 - **PATH=\$PATH:/Users/chucknorris/homer/bin/**
 - NOTE: Cygwin users may need to use a different format:
PATH=/Users/chucknorris/homer/bin:\${PATH}
 - NOTE: Cygwin users, if having trouble, may also need to set [windows path variables](#)
 - NOTE: If using Mac OS X, the ~/.bash_profile file is hidden in Finder. To edit type “**open -a TextEdit ~/.bash_profile**” at the command line.
 - NOTE: If ~/.bash_profile doesn't exist in your mac, create a new file in your home directory, place the PATH=... line in it, and then rename the file using the command-line prompt: “**mv newfilename ~/.bash_profile**”
5. Reset your terminal so that the changes to the PATH variable take effect
 - **source ~/.bash_profile**

You should now be able to execute programs in the homer/bin directory by just typing their name.

Downloading Homer Packages

The basic Homer installation does not contain any sequence data. To download sequences for use with homer, use the **configureHomer.pl** script. To get a list of available packages:

perl /path-to-homer/configureHomer.pl --list

To install packages, simply use the --install option and the name(s) of the package(s).

perl /path-to-homer/configureHomer.pl --install mouse (to download the mouse promoter set)

perl /path-to-homer/configureHomer.pl --install mm8 (to download the mm8 version of the mouse genome)

perl /path-to-homer/configureHomer.pl --install hg19r (to download the hg19 repeat masked version of the human genome)

Additional information on configuring or customizing HOMER, go [here](#).

Updating Homer

To update Homer, simply type:

perl /path-to-homer/configureHomer.pl --update

Or, alternatively you can simply force the reinstallation of the basic software...

perl /path-to-homer/configureHomer.pl --install homer

Homer will automatically check which packages are out of date and replace them.

When in trouble, and nothing seems to be working correctly, either:

1. Delete the homer directory and start over!
2. Drop me a line (cbenner@ucsd.edu) so I can fix the problem, since there is a good chance it's a problem with the software.
3. You could try Chuck, but I'm not responsible for any bodily harm.

Installing 3rd Party Software

Homer uses WebLogo (Crooks et al.) to visualize motifs graphically. The WebLogo software uses Ghostscript to generate image files, so both must be installed to successfully create sequence logos. In addition, BLAT (Kent et al.) is used for certain specialized ChIP-Seq analysis, where it can be used to remove redundant sequences during the analysis (not necessary for 99% of users to install)

Software for sequence logos:

- 1.) Download and Install Ghostscript
 - a.) Download the appropriate file from <http://pages.cs.wisc.edu/~ghost/> (GPL Ghostscript)
 - b.) Unzip and Untar the file
tar zxvf ghostscript-xxx.tar.gz
 - c.) Change to the base ghostscript directory
cd ghostscript-xxx
 - d.) Run the following commands to install ghost script

```
./configure  
make  
sudo make install
```

(if you do not have root access, you will need to specify a directory that you have access to when you run the configure script) i.e.

```
./configure --prefix=/Users/chucknorris/software/gs  
(you may also want to add the ghostscript bin/ directory to your  
~/.bash_profile file  
to make sure the “gs” program is executable)
```

2.) Download and Install Weblogo (version 2.8.2 - **Does NOT work with version 3.0!!!!**)

- a.) Download the program from <http://weblogo.berkeley.edu/>
- b.) No additional steps needed to compile and install the program, except...
- c.) Need to add the weblogo base directory to your executable path
i.e. edit your **~/.bash_profile** file to include:
PATH=\$PATH:/Users/chucknorris/weblogo/

3.) Download and Install blat (this is used to check for redundant input sequences)

- a.) Download program from <http://genome-test.cse.ucsc.edu/~kent/exe/>
It is recommended that you download the precompiled blat suite
- b.) Unzip the file and compile (if you downloaded the source code)
- c.) Add base blat directory to your executable path
i.e. edit your **~/.bash_profile** file to include:
PATH=\$PATH:/Users/chucknorris/blat/

The commands **gs**, **seqlogo**, and **blat** should now work from the command line (if they don't and you think they should, remember to type: **source ~/.bash_profile**)

[Next: Configuring HOMER](#)



Can't figure something out? Questions, comments, concerns, or other feedback:
cbenner@ucsd.edu



HOMER

Software for motif discovery and next-gen sequencing analysis

Configuring and Customizing HOMER

In an effort to make sure things are standardized for analysis, HOMER organizes promoters, genome sequences and annotation into packages. Versions are based on assemblies from the [UCSC Genome Browser](http://genome.ucsc.edu/). Accession numbers, gene ontology definitions, motif libraries are all part of the standard HOMER installation.

Basic configuration of HOMER

Configuration is handled automatically through the **configureHomer.pl** script, which should reside in the directory where [HOMER is installed](#) (i.e. /path-to-homer/). To see which packages are available, run the **configureHomer.pl** script:

```
perl /path-to-homer/configureHomer.pl -list
```

Every time you run the configureHomer.pl script, it will attempt to update the available packages by downloading the [update.txt](#) file from biowhat.ucsd.edu. Using this, the program will assess which packages are installed and which are available to download.

To install or remove any packages, simply rerun the command using "**-install <package name>**" or "**-remove <package name>**".

```
perl /path-to-homer/configureHomer.pl -install human
```

This would configure HOMER for analysis of human promoters.

As of v3.1, HOMER attempts to be more user-friendly when it comes to custom genomes. For each program where you would normally specify the *genome* (i.e. mm9r), HOMER now accepts the path to the FASTA files or directories of FASTA files. In these cases you do not need to "configure" a genome with HOMER.

Organization of HOMER

What follows is a short description of how HOMER is organized - as some researchers may want to force HOMER to do things that aren't available out of the box, this might help them accomplish this successfully!

HOMER configuration is stored in a file named "**config.txt**" which is located in the base Homer directory. This is a tab-delimited file that is read by various programs to determine where certain data is stored. Directories to genome or promoter based data are stored here (given relative to the base Homer directory).

Other standard files, such as a **README.txt**, **COPYING**, and **Homer.pdf** documentation are also found in this directory, as well as the **configureHomer.pl** script and the **update.txt** file which is downloaded each time **configureHomer.pl** is evoked.

Sub-Directories:

bin/ - location of all perl scripts and executable programs that apart of HOMER. There is a lot of "stuff" in here, some of which are half finished, abandoned, or simply don't work. These pages only talk about the ones that do work :)

cpp/ - location of c++ source files. Parts of the program which need to be fast and/or memory efficient are written in c++. As time goes on, and data sets get bigger, I've been slowly migrating perl programs to c++. I love perl - it's much much faster to write a useful program, but in the end c++ is much much faster at executing.

data/ - location of all the data files for HOMER

data/accession/ - location of flat files for accession number conversion. For each organism, there is a **org2gene.tsv** and **org.description**, both of which are tab-delimited text files, which are used for ID conversion and annotation information.

data/GO/ - gene ontology files (*.genes) that are tab-delimited text files with GO ID, GO name, and a comma separated list of gene IDs for various "ontologies". These files are species independent (contain IDs from several organisms). The names of the files are hard-coded in the gene ontology program, so you can either replace the files with something you are interested in or change the hard-coded file names in **findGO.pl** program.

data/knownTFs/ - This directory contains motif libraries used for checking the identities of *de novo* motifs (**all.motifs** - most of which come from [JASPAR](#)), and a list of previously found motifs (**known.motifs**) used for checking the enrichment of known motifs. These files can be replaced with similar formatted files if you wish. There is also a sub directory, named "**data/knownTFs/motifs/**", which contains *.**motif** files for my own personal motif library (to be used with other applications such as **annotatePeaks.pl**).

data/misc/ - I guess if don't like reading about a legendary human being at the bottom of your motif finding results, you could delete or

change this file. Be warned - I'm not responsible if you end up getting a swift roundhouse kick to the face.

data/promoters/ - files used for promoter motif finding. For each promoter set (called "*name*"), there are several files:

- name.seq (promoter sequence)
- name.mask (repeat-masked promoter sequence)
- name.cgbins (assignments of promoters to different CpG classes)
- name.cgfreq (CpG/GC frequencies)
- name.pos (genomic positions of promoters)
- name.redun (mapping between redundant promoters i.e. share similar sequence)
- name.cons (might be missing - 0-9 phastCons score or promoter sequence)
- name.base (IDs to use as background - typically expressed or confident promoters)
- name.base.gene (name.base in terms of gene ids to use with gene ontology analysis)

data/genomes/ - each **genome** has it's own directory. Within each directory are the ***.fa** files or ***.fa.masked** files containing the genome sequence. In addition, there are several annotation files:

- *.fa or *.fa.masked files for each chromosome
- genome.tss (positions of refseq transcription start sites)
- genome.tts (positions of refseq transcription termination sites)
- genome.splice3p (positons of refseq 3' splice sites)
- genome.splice5p (positons of refseq 5' splice sites)
- genome.aug (positions of refseq translation start codons)
- genome.stop (positions of refseq translation stop codons)
- genome.rna (refseq RNA definition file)
- genome.repeats.rna (repeat RNA definition file)
- genome.basic.annotation (exon/intron/TSS/TTS/intergenic region annotations)
- genome.full.annotation (basic with CpG island and repeats annotated)
- conservation/ subdirectory (contains "FASTA-like" files with phastcons information)
- annotation/ subdirectory (contains annotation definitions for the GenomeOntology)

Customization

BTW, if you need help or believe you work on a common organism, drop me a line (cbenner@ucsd.edu) and maybe I can whip up these files.

If all you are interested in is Motif finding, findMotifs.pl accepts FASTA format

files, making it a fairly general tool. You can also call homer2 directly with FASTA files. However, if you want to perform analysis using custom genome or promoter data sets, this will help guide you through that process.

Custom Genomes:

As of v3.1, HOMER attempts to be more user-friendly when it comes to custom genomes. For each program where you would normally specify the *genome* (i.e. mm9r), HOMER now accepts the path to the FASTA files or directories of FASTA files. In these cases you do not need to "configure" a genome with HOMER. It's up to you to decide if you would rather try to configure a custom genome yourself or simply use the path to the FASTA file(s).

Genome FASTA file format:

Every FASTA file for genomic use must have the chromosome names in the FASTA headers (i.e. ">chr1 ...\n"). There must be whitespace after the name of the chromosome, and the chromosome names must match exactly to the chromosome names you specify in your input files (i.e. read alignment files or peak/BED files). If they don't match, there is a very very good chance that the genome you are trying use is not the same version of the genome that your alignment files for BED files came from. What this means is that "1" is not the same as "chr1", and "chr1" is not the same as "mm_ref_chr1". Only "chr1" is the same as "chr1".

As for the genomic FASTA files themselves, HOMER can choke them down if they are configured in one of the 3 different formats:

1. Single FASTA file - specify the path of this file on the command line. This is the easiest option, but is a little slower for the program since it has to more-or-less read through the entire genome to get data for a single chromosome. Do not use this option if trying to "configure" it with HOMER (see below and option 3).
2. Directory of chromosome FASTA files - specify the path of the directory (i.e. "/data/genomes/human/", DO NOT use something like "/data/genomes/human/*.fa"). In this case, the FASTA files should have the name of the chromosome followed by ".fa" or ".fa.masked" (i.e. chr1.fa). This is the preferred way from an efficiency stand point, and should be the way you store the genome if trying to "configure" it with HOMER (see below).
3. Directory with a single FASTA file named "genome.fa" - specify the path of the directory or the file in this case ("/data/genomes/human/"). Use this option if there are too many chromosomes (or scaffolds) for your organism (i.e. > 100) and you want to configure HOMER. Usually splitting the genome into individual chromosome files will leave you with a ton of files in these cases which is a pain (some organismes

have 1000s of scaffolds). Or, if you're treating mRNA as a "genome"...

In theory, tricking HOMER into using a configured custom genome is pretty easy. "Billy Mayes here, in just 3 easy steps:"

1. Create a new directory bearing the name of the genome you wish to install (i.e. /path-to-homer/data/genomes/*genomeName*/)
2. Place the *.fa or *.fa.masked files there.
3. Create a file named *genome.tss* (basically a peak file with positions from -2kb to +2kb relative to the TSS). This is the file annotatePeaks.pl looks for.
4. (Optional) Create a file named *genome.basic.annotation* (try to follow format of other annotation files).
5. Edit the config.txt file to include the name of your new genome - basically copy one of the existing genome entries, and change the appropriate information.

And that should be it...

Custom Promoter Sets:

Also *in theory*, the program **loadPromoters.pl** was developed to help with establishing custom promoter sets. This command attempts to automatically trick HOMER into using your custom idea of what a promoter is (which could, in theory, be exon junctions, enhancers, etc.).

This command can be specified using genomic positions:

```
loadPromoters.pl <name> <tss/peak file> <organism> <genome>  
<masked genome> <id type> [options]
```

Or it can be specified using a FASTA file:

```
loadPromoters.pl <name> null null null null <id type> [options]
```

Additional directions are available when running the **loadPromoters.pl** command with no additional arguments.



Can't figure something out? Questions, comments, concerns, or other feedback:
cbenner@ucsd.edu



HOMER

Software for motif discovery and ChIP-Seq analysis

Introduction to HOMER

The best way to learn about HOMER is to go through the tutorial pages. We've tried to spell out what happens in each step and explain the "why". A brief description of the Motif Finding component of HOMER is found below. Explanation of the sequencing analysis components of HOMER are integrated into the tutorials.

General Introduction to Motif Discovery with HOMER

HOMER is a collection of tools that are commonly needed for the analysis of gene expression profiling (microarray) and genome-wide location analysis experiments (ChIP-Seq or ChIP-Chip). There are also routines for other types of sequencing experiments, such as DNase-Seq or GRO-Seq.

Some of the things HOMER does NOT DO is find differentially expressed genes, cluster gene expression profiles, or search for all the instances Transfac motifs in order to make you hopelessly confused!!! The idea was not to completely reinvent the wheel if possible.

Unfortunately, HOMER must be run as a command-line tool, and may be difficult to use if you are new to UNIX. While commands have been distilled to be as simple and user-friendly as possible, basic knowledge of the UNIX environment and file system is critical (but can probably be learned quickly after typing "unix tutorial" into google). I am proud to say that many of the people using HOMER are completely new to UNIX, so it is indeed possible. In addition, a spreadsheet program (i.e. EXCEL) is needed to graph and visualize some of the results produced by HOMER.

Below is a description of how motif analysis is executed with HOMER. Documentation describing the steps of analysis for [Next-Gen Sequencing](#) (or genomic position analysis) or [Microarrays](#) (gene-based analysis) are covered in separate sections.

De Novo Motif Discovery Strategy

HOMER was designed as a de novo motif discovery algorithm that scores motifs by looking for motifs with differential enrichment between two sets of sequences. This means that HOMER uses two sets of sequences when performing motif

finding – 1. target sequences of interest (i.e. promoters of genes that are co-regulated) and 2. a set of background sequences (i.e. promoters of genes that are not regulated). Without background sequences a motif discovery algorithm must guess what sequences are expected to be found by chance, such as assuming background sequences are a random collection of A, C, G, and T. This can be extremely dangerous since real genomic sequence is anything but random.

In practice HOMER will try to select the appropriate background sequences for you, but results can vary depending on what is used as background and certain applications may require careful consideration of these sequences. By default HOMER will use confident, non-regulated promoters as background when analyzing promoters, and sequences in the vicinity of genes for ChIP-Seq analysis (i.e. from –50kb to +50kb). In each case sequences are matched for their GC content to avoid bias from CpG Islands.

Once target and background sequences are chosen, HOMER looks for motifs of a specific length that are overrepresented in the target set relative to the background set. This enrichment is measured using the cumulative hypergeometric distribution (or cumulative binomial distribution for large data sets), and places no requirement on the degeneracy of the motif or the number of occurrences. Motifs are found by first exhaustively checking the enrichment of simple motifs, then refining promising candidates into accurate probability matrices.

With v3.0 of HOMER, the motif discovery software has been rewritten and modernized (the homer2 executable). There is a subtle, but very important difference in how the new version of HOMER performs *de novo* motif analysis. The original HOMER divided the input sequences into short oligos to perform the analysis, and once a motif was found, only the oligos considered "bound" by the motif were removed from the analysis. The problem was that several oligos representing "offsets" of the original motif (think GGAAGT vs. GAAGTg) were left for the 2nd round of motif enrichment to find, creating results that often contained several versions of the original motif. The new version revisits the input sequences and removes all oligos that are slightly offset from the optimal motifs, making it much more sensitive to co-enriched motifs.

Known Motif Discovery Strategy

The biggest problem when looking for “known” motifs is defining how degenerate you should allow them to be. To circumvent this problem, we loaded motif derived from published ChIP-Seq experiments that were already optimized for degeneracy thresholds.

Interpretation of Motif Discovery Results

De Novo Results

Unfortunately, if you give HOMER random data, HOMER will find motifs,

and they may look significant. Due to the finite amount of data and many degrees of freedom in a motif probability matrix, it is easy to find a motif with a seemingly significant p-value. Because of this, we can only trust the most promising of motifs as likely to be real. For most promoter datasets, motifs with a p-value of more than $1e-10$ or even $1e-12$ are likely to be false positives. In general the p-value cutoff should be estimated by randomizing data labels and running the algorithm several times. In practice you should start ignoring results that are either below $1e-10$ or when the results start becoming very different from one another (in terms of sequence) yet have similar p-values. In addition, high quality motifs usually appear multiple times in the list with different offsets (i.e. nnnTGACTCAnn and nTGACTCAnnnn). HOMER attempts to remove extremely similar motifs, but different offsets of motifs are likely to be present if the signal is strong (remember motifs may appear as if on the negative strand).

Matching *De Novo* to Known Motifs

Homer makes every attempt to tell you if the motifs it discovered resembles a known motif. The difficulty of interpreting these results **SHOULD NOT BE UNDERESTIMATED!!!** Consider the following:

1. Databases of known motifs are a mixture of accurate and inaccurate motifs
2. Databases of known motifs are not complete
3. The literature (especially motif finding papers) is full of inaccurate assessments and motif annotations that are ludicrous.

HOMER tries to find the known motifs with the best correlation between the known motif and *de novo* motif. It then aligns the motifs from the top hits so that you can see it and judge the alignment for yourself. The top known motif match is not always the best match. The top match is not always annotated correctly. If you feel something is worth pursuing, look up the known binding sites of the transcription factor via PUBMED. Feedback I got when writing the program was to provide the name of the motif in the main result table – this was promptly followed by the misinterpretation of results because people are too lazy to look at the alignment to figure out if it makes any sense. These results do not write the paper for you – critical thinking and follow-up is required.

Additional Reading: [Tips for de novo motif finding](#)

Known Motif Enrichment

First and most important: There is a subtle but **IMPORTANT** difference between looking for motifs *de novo* and looking for known motif enrichment. *De novo* motif discovery allows you to directly query the sequence to discover which motifs are the **MOST** enriched sequences in your target set. Known motif discovery will simply tell you which of the known motifs is most enriched in your target set.

This may not seem important but consider the following scenario: You have a set of random GA-rich sequences and compare them to random genomic sequences. De novo motif finding will likely return a G/A-rich matrix that doesn't look anything like a transcription factor. Known motif finding will return astonishingly high p-values for motifs like PU.1 (GAGGAAGT) and ISRE (GAAACTGAAA). Because of this de novo motif finding results are much more trustful in terms of results.

The greatest advantage to using known motifs is found when you have a limited set of target sequences. The less data that is available or the weaker the true signal, it is difficult for de novo motif finding to accurately define a signal that is significant. Known motifs have the advantage of many less degrees of freedom and in many cases find the correct motifs when the enrichment falls below the $1e-10$ thresholds for believability when considering de novo results.

A more detailed description of the motif finding procedure is available in the [Motif Finding Tutorial](#).

[Next: Introduction to Homer Programs](#)



Can't figure something out? Questions, comments, concerns, or other feedback:
cbenner@ucsd.edu



HOMER

Software for motif discovery and ChIP-Seq analysis

HOMER Program Index

Below is a quick introduction to the different programs included in HOMER. Running each program without any arguments will provide basic instructions and a list of command line options.

FASTA file Motif Discovery

findMotifs.pl - performs motif analysis with lists of Gene Identifiers or FASTA files (See [FASTA file analysis](#))

homer2 - core component of motif finding (Called by everything else , See [FASTA file analysis](#))

Gene/Promoter-based Analysis

findMotifs.pl - performs motif and gene ontology analysis with lists of Gene Identifiers, both promoter and mRNA motifs (See [Gene ID Analysis Tutorial](#))

findGO.pl - performs only gene ontology analysis with lists of Gene Identifiers (Called by findMotifs.pl, See [Gene Ontology Analysis](#))

loadPromoters.pl - setup custom promoter sets for specialized analysis (See [Customization](#))

Next-Gen Sequencing/Genomic Position Analysis

findMotifsGenome.pl - performs motif analysis from genomic positions (See [Finding Motifs from Peaks](#))

makeTagDirectory - creates a "tag directory" from high-throughput sequencing alignment files, performs quality control (See [Creating a Tag Directory](#))

makeUCSCfile & makeBigWig.pl - create bedGraph file for visualization with the UCSC Genome Browser (See [Creating UCSC file](#))

findPeaks - find peaks in ChIP-Seq data, regions in histone data, de novo transcripts from GRO-Seq (See [Finding ChIP-Seq Peaks](#))

analyzeChIP-Seq.pl - automation of programs found above (See [Automation of ChIP-Seq analysis](#))

annotatePeaks.pl - annotation of genomic positions, organization of motif and sequencing data, histograms, heatmaps, and more... (See [Annotating Peaks, Quantification](#))

analyzeRNA.pl - quantification of RNA levels across transcripts (See [RNA quantification](#))

mergePeaks - find overlapping peak positions (See [Comparing ChIP-Seq Peaks](#))

homerTools - basic sequence manipulation (See [Sequence Manipulation](#))

tagDir2bed.pl - output tag directory as an alignment BED file (See [Miscellaneous](#))

bed2pos.pl, pos2bed.pl - convert between HOMER peak file format and BED file format (See [Miscellaneous](#))

checkPeakFile.pl - use this to see if your peak file is in the correct format

removeOutOfBoundsReads.pl - remove reads found outside acceptable chromosome limits

Motif Manipulation

compareMotifs.pl - checks a library of motifs for known motifs, creates in HTML output summarizing motif results (described [here](#)).

motif2Logo.pl - creates a PNG or PDF logo from any motif file.

revoppMotif.pl - creates a new motif file reflecting the nucleotide preferences of the opposite strand.

seq2profile.pl - creates a new motif file from a consensus sequence

Additional Utilities that may be useful (and sub-programs used by those above)

addData.pl, addDataHeader.pl, mergeData.pl - tools for joining/merging tab separated flat files

homerTools extract - extract genomic sequence for peaks from a peak file.

homerTools freq - finds nucleotide/dinucleotide frequencies of a collection of sequence and GC/CpG content.

getPeakTags - finds sequencing tags associated with genomic positions.

scanMotifGenomeWide.pl - look for all instances of a motif in the genome.

tagDir2bed.pl - convert *.tags.tsv file directory into a BED file for use with other programs

homer2 - new motif finding program

homer - original motif finding program (not used anymore)

getTopPeaks.pl - return peaks with the best peak scores.

getFocalPeaks.pl - return peaks with the highest focus ratios.

assignGenomeAnnotation - assign peaks to specific annotations in the

genome

fasta2tab.pl, tab2fasta.pl - convert between HOMER-style sequence file and a FASTA file.

changeNewLine.pl - converts mac and dos formatted text files (new lines of "\r" and "\r\n") to UNIX style ("\n").



Can't figure something out? Questions, comments, concerns, or other feedback:
cbenner@ucsd.edu



HOMER

Software for motif discovery and next-gen sequencing analysis

Next-Generation Sequencing Analysis

ChIP-Seq is the best thing that happened to ChIP since the antibody. It is 100x better than ChIP-Chip since it escapes most of the problems of microarray probe hybridization. Plus it is cheaper, and genome wide. But ChIP-Seq is only the tip of the iceberg - there are many inventive ways to use a sequencer. Below are a list of the the more popular methods that will be covered below:

ChIP-Seq: Isolation and sequencing of genomic DNA "bound" by a specific transcription factor, covalently modified histone, or other nuclear protein. This methodology provides genome-wide maps of factor binding. Most of HOMER's routines cater to the analysis of ChIP-Seq data.

DNase-Seq: Treatment of nuclei with a restriction enzyme such as DNase I will result in cleavage of DNA at accessible regions. Isolation of these regions and their detection by sequencing allows the creation of DNase hypersensitivity maps, providing information about which regulatory elements are accessible in the genome.

MNase-Seq: Micrococcal Nuclease (MNase) is a restriction enzyme that degrades genomic DNA not wrapped around histones. The remaining DNA represents nucleosomal DNA, and can be sequencing to reveal nucleosome positions along the genome. This method can also be combined with ChIP to map nucleosomes that contain specific histone modifications.

RNA-Seq: Extraction, fragmentation, and sequencing of RNA populations within a sample. The replacement for gene expression measurements by microarray. There are many variants on this, such as Ribo-Seq (isolation of ribosomes translating RNA), small RNA-Seq (to identify miRNAs), etc.

GRO-Seq: RNA-Seq of nascent RNA. Transcription is halted, nuclei are isolated, labeled nucleotides are added back, and transcription briefly restarted resulting in labeled RNA molecules. These newly created, nascent RNAs are isolated and sequenced to reveal "rates of transcription" as opposed to the total number of stable transcripts measured by normal RNA-seq.

Unsolicited advice: If you are going to perform RNA-Seq, use a protocol for **STRAND-SPECIFIC** RNA-Seq. Why would you throw away the strand information?

More Unsolicited advice: Run controls!!!

Yet More Unsolicited advice: Be careful about GC-bias!!

(mini tutorial for each of these data types are on their way...)

HOMER offers solid tools and methods for interpreting Next-gen-Seq experiments. In addition to UCSC visualization support and peak finding [and motif finding of course], HOMER can help assemble data across multiple experiments and look at positional specific relationships between sequencing tags, motifs, and other features. You do not need to use the peak finding methods in this package to use motif finding.

Basic Analysis can be separated into the following steps for each experiment type:

1. [Mapping to the genome](#) (NOT performed by HOMER, but important to understand)
2. [Creation Tag directories, quality control, and normalization](#) (**makeTagDirectory**)
3. [UCSC visualization](#) (**makeUCSCfile**, **makeBigWig.pl**)
4. [Peak finding / Transcript detection / Feature identification](#) (**findPeaks**)
5. [Motif analysis](#) (**findMotifsGenome.pl**)
6. [Annotation of Peaks](#) (**annotatePeaks.pl**)
7. [Quantification of Data at Peaks/Regions in the Genome/Histograms and Heatmaps](#) (**annotatePeaks.pl**)
8. [Quantification of Transcripts](#) (**analyzeRNA.pl**)

Additional analysis strategies:

- [General sequence manipulation tools](#) (**homerTools**)
- [Miscellaneous Tools for Sharing Data between programs, etc.](#) (**tagDir2bed.pl**, **bed2pos.pl**, **pos2bed.pl** ...)
- [Finding overlapping or differentially bound peaks](#) (**mergePeaks**, **getDifferentialPeaks**)
- [ChIP-Seq analysis automation](#) (**analyzeChIP-Seq.pl**)
- [Description of file formats](#)

NOTE: The current implementation is geared for single tag sequencing. ****Most**** of the types of experiments above don't necessarily gain much from paired-end sequencing (in terms of information/per bp).



Can't figure something out? Questions, comments, concerns, or other feedback:
cbenner@ucsd.edu



HOMER

Software for motif discovery and next-gen sequencing analysis

Next-Generation Sequencing Analysis

ChIP-Seq is the best thing that happened to ChIP since the antibody. It is 100x better than ChIP-Chip since it escapes most of the problems of microarray probe hybridization. Plus it is cheaper, and genome wide. But ChIP-Seq is only the tip of the iceberg - there are many inventive ways to use a sequencer. Below are a list of the the more popular methods that will be covered below:

ChIP-Seq: Isolation and sequencing of genomic DNA "bound" by a specific transcription factor, covalently modified histone, or other nuclear protein. This methodology provides genome-wide maps of factor binding. Most of HOMER's routines cater to the analysis of ChIP-Seq data.

DNase-Seq: Treatment of nuclei with a restriction enzyme such as DNase I will result in cleavage of DNA at accessible regions. Isolation of these regions and their detection by sequencing allows the creation of DNase hypersensitivity maps, providing information about which regulatory elements are accessible in the genome.

MNase-Seq: Micrococcal Nuclease (MNase) is a restriction enzyme that degrades genomic DNA not wrapped around histones. The remaining DNA represents nucleosomal DNA, and can be sequencing to reveal nucleosome positions along the genome. This method can also be combined with ChIP to map nucleosomes that contain specific histone modifications.

RNA-Seq: Extraction, fragmentation, and sequencing of RNA populations within a sample. The replacement for gene expression measurements by microarray. There are many variants on this, such as Ribo-Seq (isolation of ribosomes translating RNA), small RNA-Seq (to identify miRNAs), etc.

GRO-Seq: RNA-Seq of nascent RNA. Transcription is halted, nuclei are isolated, labeled nucleotides are added back, and transcription briefly restarted resulting in labeled RNA molecules. These newly created, nascent RNAs are isolated and sequenced to reveal "rates of transcription" as opposed to the total number of stable transcripts measured by normal RNA-seq.

Unsolicited advice: If you are going to perform RNA-Seq, use a protocol for **STRAND-SPECIFIC** RNA-Seq. Why would you throw away the strand information?

More Unsolicited advice: Run controls!!!

Yet More Unsolicited advice: Be careful about GC-bias!!

(mini tutorial for each of these data types are on their way...)

HOMER offers solid tools and methods for interpreting Next-gen-Seq experiments. In addition to UCSC visualization support and peak finding [and motif finding of course], HOMER can help assemble data across multiple experiments and look at positional specific relationships between sequencing tags, motifs, and other features. You do not need to use the peak finding methods in this package to use motif finding.

Basic Analysis can be separated into the following steps for each experiment type:

1. [Mapping to the genome](#) (NOT performed by HOMER, but important to understand)
2. [Creation Tag directories, quality control, and normalization.](#) (**makeTagDirectory**)
3. [UCSC visualization](#) (**makeUCSCfile**, **makeBigWig.pl**)
4. [Peak finding / Transcript detection / Feature identification](#) (**findPeaks**)
5. [Motif analysis](#) (**findMotifsGenome.pl**)
6. [Annotation of Peaks](#) (**annotatePeaks.pl**)
7. [Quantification of Data at Peaks/Regions in the Genome/Histograms and Heatmaps](#) (**annotatePeaks.pl**)
8. [Quantification of Transcripts](#) (**analyzeRNA.pl**)

Additional analysis strategies:

- [General sequence manipulation tools](#) (**homerTools**)
- [Miscellaneous Tools for Sharing Data between programs, etc.](#) (**tagDir2bed.pl**, **bed2pos.pl**, **pos2bed.pl** ...)
- [Finding overlapping or differentially bound peaks](#) (**mergePeaks**, **getDifferentialPeaks**)
- [ChIP-Seq analysis automation](#) (**analyzeChIP-Seq.pl**)
- [Description of file formats](#)

NOTE: The current implementation is geared for single tag sequencing. ****Most**** of the types of experiments above don't necessarily gain much from paired-end sequencing (in terms of information/per bp).



Can't figure something out? Questions, comments, concerns, or other feedback:
cbenner@ucsd.edu



HOMER

Software for motif discovery and next-gen sequencing analysis

Alignment of High-throughput Sequencing Data

Homer does not perform alignment - this is something that must be done before running homer. Several quality tools are available for alignment of short reads to large genomes. Check out [this link](#) for a list of programs that do short read alignment. BLAST, BLAT, and other traditional alignment programs, while great at what they do, are not practical for alignment of these types of data.

If you need help deciding on a program to use, I'll recommend [Bowtie](#) (it's nice and fast).

If you have a core that maps your data for you, don't worry about this step. However, in many cases there is public data available that hasn't been mapped to the genome or mapped to a different version of the genome or mapped with different parameters. In these cases it is nice to be able to map data yourself to keep a nice, consistent set of data for analysis.

Most types of ChIP-Seq/DNase-Seq/MNase-Seq and GRO-Seq simply need to be mapped to the genome, as they represent the sequencing of genomic DNA (or nascent RNA, which should not be spliced yet). If analyzing RNA-Seq, you may be throwing away interesting information about splicing if you simply align the data to the genome. If aligning RNA, I'd recommend sticking to the formal wear and trying [Tophat](#), which does a good job of identifying splice junctions in your data.

Which reference genome (version) should I map my reads to?

Both the organism and the exact *version* (i.e. hg18, hg19) are very important when mapping sequencing reads. Reads mapped to one version are NOT interchangeable with reads mapped to a different version. I would follow this recommendation list when choosing a genome (Obviously try to match species or sub species when selecting a genome):

1. Do you have a favorite genome in the lab that already has a bunch of experiments mapped to it? Use that one.
2. Do any of your collaborators have a favorite genome?
3. Use the latest stable release - I would recommend using genomes curated at UCSC so that you can easily visualize your data later using the [UCSC Genome Browser](#). (i.e. mm9, hg18)

Q: I'm changing genome versions, can I just "liftover" my data using UCSC liftover tool, or do I need to remap it to the new genome version?

If you want to do it right, you need to remap it. This is because some regions of the genome that are considered "unique" in one version may suddenly be found multiple times in the new version and vice versa, so using the liftover tool will yield different results from remapping. However, liftover is fine if you're looking for a quick and dirty solution. If you feel like cheating, as Chuck often does, try **convertCoordinates.pl**. - it's a wrapper that uses the "liftOver" program to migrate peak files and whole Tag Directories.

Should I trim my reads when mapping to the genome?

Depends. In the old days, the read quality dropped off quite a bit past ~30 bp, but these days even the end of sequencing reads are pretty high quality. In the end, I would recommend mapping ~32 bp reads with up to 3 mismatches, using only the uniquely alignable reads for downstream analysis. That will give you access to probably 80-90% of what is interesting in your data set.

I have barcodes and/or adapter sequences in my reads. Should I remove them first or just map them?

You should definitely remove the adapter sequences or other "non-biological" sequences before mapping. Various tools can accomplish this. You can check out [homerTools for trimming sequences and dealing with adapters](#). [Galaxy](#) also has a nice variety of tools for accomplishing this type of stuff.

Example - Alignment with bowtie:

Step 1 - Build Index (takes a while, but only do this once):

After installing [bowtie](#), the reference genome must first be "indexed" so that reads may be quickly aligned. You can download pre-made indices from the bowtie website (check for those [here](#) first). Otherwise, to perform make your own from FASTA files, do the following:

1. Download FASTA files for the unmasked genome of interest if you haven't already (i.e. from [UCSC](#))
2. From the directory containing the FASTA files, run the "bowtie-build" command. For example, for hg18:
 - **/path-to-bowtie-programs/bowtie-build chr1.fa,chr2.fa,chr3.fa,...chrY.fa,chrM.fa hg18**
 - Where ... are the rest of the *.fa files. This command will take a long time to run, but will produce several files named **hg18*.ebwt**
3. Copy the *.ebwt files to the bowtie indexes directory so that bowtie knows where to find them later:

- `cp *.ebwt /path-to-bowtie-programs/indexes/`

Step 2 - Align sequences with bowtie (perform for each experiment):

The most common output format for high-throughput sequencing is FASTQ format, which contains information about the sequence (A,C,G,Ts) and quality information which describes how certain the sequencer is of the base calls that were made. In the case of Illumina sequencing, the output is usually a "s_1_sequence.txt" file. In addition, much of the data available in the [SRA](#), the primary archive of high-throughput sequencing data, is in this format. To map this data, run the following command:

```
/path-to-bowtie-programs/bowtie -q --best -m 1 -p <# cpu>  
<genome> <fastq file> <output filename>
```

Where <genome> would be hg18 from the index made above, <fastq file> could be "s_1_sequence.txt", and <output filename> something like "s_1_sequence.hg18.alignment.txt"

The parameters "--best" and "-m 1" are needed to make sure bowtie outputs only unique alignments. There are many options and many different ways to perform alignments, with different trade-offs for different types of projects - well beyond the scope of what I am describing here.

NOTE: HOMER contains automated parsing for uniquely aligned reads from output files generated with bowtie in this fashion. Homer also accepts *eland_result.txt and *_export.txt formats from the Illumina pipeline. If different programs are used, or special parsing of output files are needed, please parse/reformat alignment files to general BED format, which is also accepted by HOMER. HOMER also accepts SAM formatted file. If using BAM files, use "samtools view input.bam > output.sam" to convert to a SAM file.



Can't figure something out? Questions, comments, concerns, or other feedback:
cbenner@ucsd.edu



HOMER

Software for motif discovery and ChIP-Seq analysis

Next-gen Sequencing Analysis: Creating a "Tag Directory" with makeTagDirectory

To facilitate the analysis of ChIP-Seq (or any other type of short read re-sequencing data), it is useful to first transform the sequence alignment into platform independent data structure representing the experiment, analogous to loading the data into a database. HOMER does this by placing all relevant information about the experiment into a "Tag Directory", which is essentially a directory on your computer that contains several files describing your experiment.

During the creation of tag directories, several quality control routines are run to help provide information and feedback about the quality of the experiment. During this phase several important parameters are estimated that are later used for downstream analysis, such as the estimated length of ChIP-Seq fragments.

Input Alignment Files

During this part of the analysis, any type of sequencing data can be use (ChIP-Seq/RNA-Seq/DNase-Seq, etc.). If these files are zipped (*.gz, *.zip, *.bz2), HOMER will automatically unzip, process, and re-zip them (if applicable) so you don't have to worry about this step.

To create a "Tag Directory", you must have alignment files in one of the following formats:

- BED format
- SAM format
- BAM format (HOMER will use "samtools view file.BAM > file.SAM" to covert to a SAM file, so "samtools" must be available)
- default bowtie output format
- *.eland_result.txt or *_export.txt format from the Illumina pipeline

If your alignment is in a different format, it is recommended that you convert it into a BED file format:

Column1: chromosome
Column2: start position
Column3: end position
Column4: Name (or strand +/-)

****Column5: Number of reads at this position****

Column6: Strand +/-

****Unfortunately, BED files are used in different ways by different groups. By default, HOMER assumes the 5th column of a BED file is the number of reads at that position. If this is NOT the case, add the option "-forceBED" in your command to tell HOMER to ignore this value.**

Alternatively (or in combination), you can make tag directories from existing tag directories or from tag files (explained below).

If your input files are named "s_1_sequence.txt", or have a suffix such as *.fq or *.fastq, then you probably have raw sequence files without raw sequence alignment information. You need to align these sequences to the genome first. [Learn about aligning data to the genome here.](#)

Paired-End Reads

Unfortunately, HOMER does NOT explicitly support paired-end reads... yet. This is primarily due to the fact that our group hasn't used paired-end sequencing for any of these applications (nor is there much data in the literature using paired-end reads for applications like ChIP-Seq, with the exception of RNA-Seq), but this is changing and support for them will be available in the near future.

For now, the best option is to separate the paired-end reads and treat them as separate single-end runs (or just use one of the two reads).

Creating Tag Directories

To make a tag directory, run the following command:

```
makeTagDirectory <Output Directory Name> [options] <alignment file1>
[alignment file 2] ...
```

Where the first argument must be the output directory (required). If it does not exist, it will be created. If it does exist, it will be overwritten.

An example:

```
makeTagDirectory Macrophage-PU.1-ChIP-Seq/ pu1.lane1.bed
pu1.lane2.bed pu1.lane3.bed
```

Several additional options exist for **makeTagDirectory**. The program attempts to guess the format of your alignment files, but if it is unsuccessful, you can force the format with "**-format <X>**".

```
makeTagDirectory Macrophage-PU.1-ChIP-Seq/ pu1.lane1.bed
pu1.lane2.bed pu1.lane3.bed -format bed
makeTagDirectory Macrophage-H3K4me1-ChIP-Seq/
s_1_sequence.align.sam -format sam
```

Sometimes BED file alignments contain stupid values in the 5th column, such as

quality information etc. HOMER will treat this value as the number of reads aligning to the same location. If this is not how the value is used, add "-forceBED" to ignore the value found in the 5th column of a BED file.

**makeTagDirectory Macrophage-PU.1-ChIP-Seq/ pu1.lane1.bed
pu1.lane2.bed pu1.lane3.bed -format bed -forceBED**

To combine tag directories, for example when combining two separate experiments into one, do the following:

makeTagDirectory Combined-PU.1-ChIP-Seq/ -d Exp1-ChIP-Seq/ Exp2-ChIP-Seq/ Exp3-ChIP-Seq/

What does makeTagDirectory do?

makeTagDirectory basically parses through the alignment file and splits the tags into separate files based on their chromosome. As a result, several *.tags.tsv files are created in the output directory. These are made to very efficiently return to the data during downstream analysis. This also helps speed up the analysis of very large data sets without running out of memory.

In the end, your output directory will contain several *.tags.tsv files, as well as a file named "**tagInfo.txt**". This file contains information about your sequencing run, including the total number of tags considered. This file is used by later programs to quickly reference information about the experiment, and can be manually modified to set certain parameters for analysis.

makeTagDirectory also performs several quality control steps shown below.

Basic Quality Control Analysis:

The following 4 basic quality control results are produced by default and are relatively inexpensive in terms of processing power to create while parsing alignment files into Tag Directories, and do not require any extra information to produce. These files are meant to be opened with a text editor or graphed using EXCEL or similar program. More detailed descriptions of these files and how to interpret them are found in the sequencing technique-specific tutorials.

Basic Tag information

tagInfo.txt - Contains basic configuration information, such as the total number of reads, the total number of unique positions with aligned reads (by genome and chromosome), and various other statistics. One of the more important parameters is "fragmentLengthEstimate=##", which provides an estimate of the length of fragments used for sequencing.

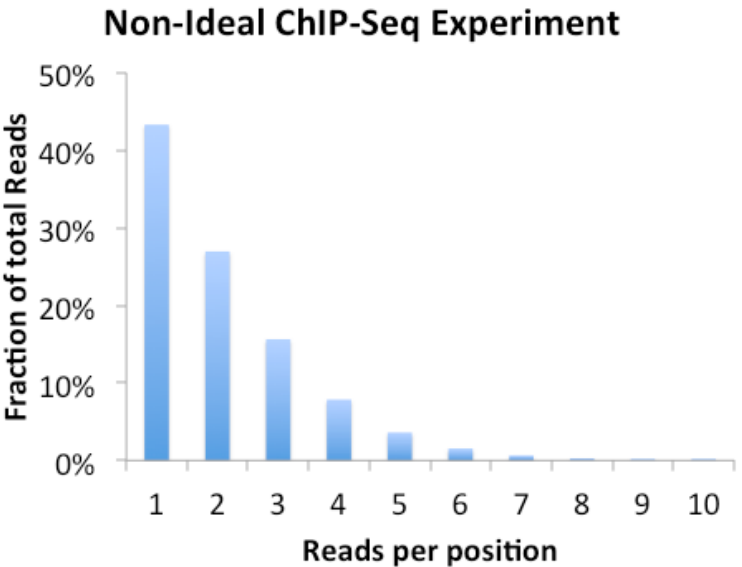
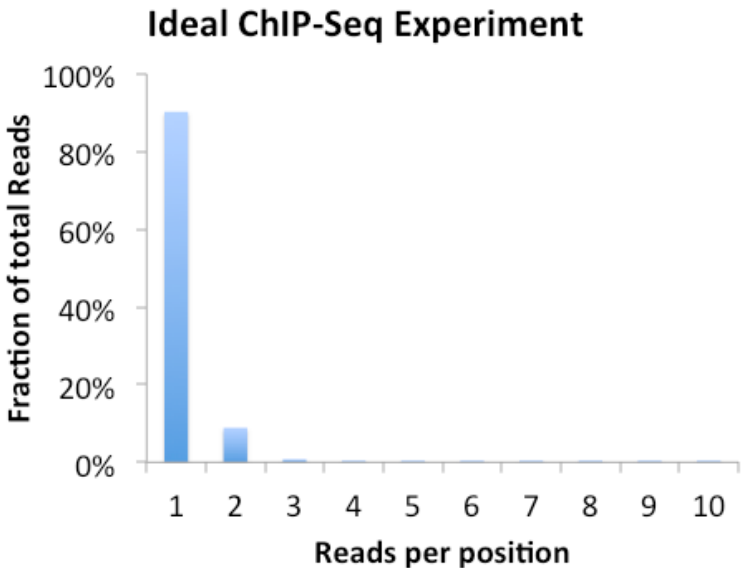
Read Length Distribution

tagLengthDistribution.txt - File contains a histogram of read lengths used

for alignment.

Clonal Tag Distribution

tagCountDistribution.txt - File contains a histogram of clonal read depth, showing the number of reads per unique position. If an experiment is "over-sequenced", you start seeing the same reads over and over instead of unique reads. Sometimes this is a sign there was not enough starting material for sequencing library preparation. Below are examples of ideal and non-ideal results - in the case of the non-ideal experiment, you probably don't want to sequence that library anymore.



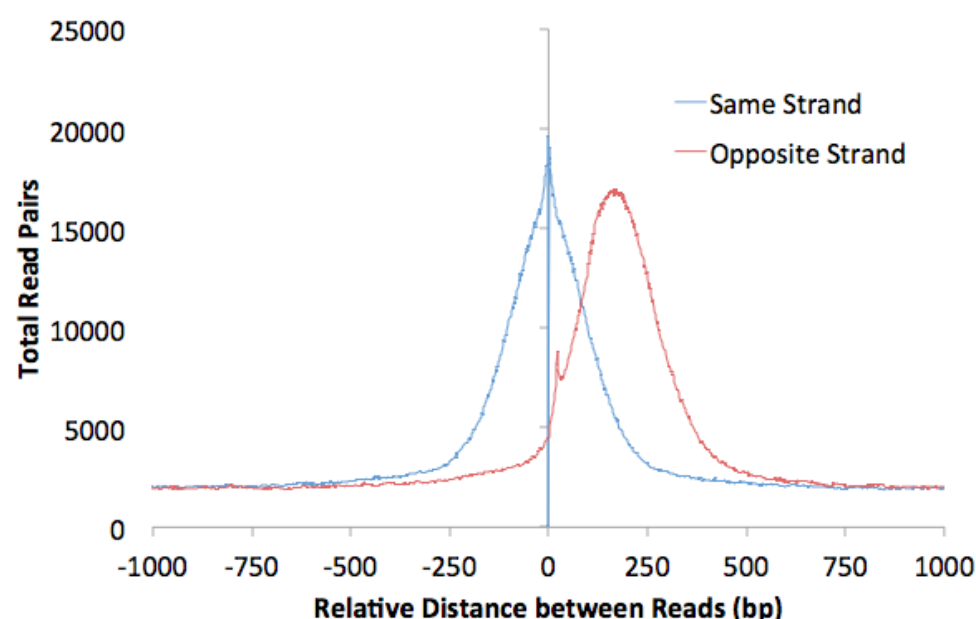
If the experiment is highly clonal and not expected to be, it might help to clean up the downstream analysis by forcing tag counts at each position to be no greater than x , where x is usually 1. To do this rerun the `makeTagDirectory` command and add `"-tbp <#>"` where $\#$ is the maxium tags per bp.

Autocorrelation Analysis

tagAutocorrelation.txt - The autocorrelation routine creates a distribution of

distances between adjacent reads in the genome. If reads are mapped to the same strand, they are added to the first column. If adjacent reads map to different strands, they are added to the 2nd column. The results from autocorrelation analysis are very useful for troubleshooting problems with the experiment, and are used to estimate the fragment length for ChIP-Seq and MNase-Seq. The fragment length is estimated by finding the position where the "opposite strand" distribution is maximum. HOMER will use this value as the fragment length unless overridden with the option **"-fragLength <#>"**.

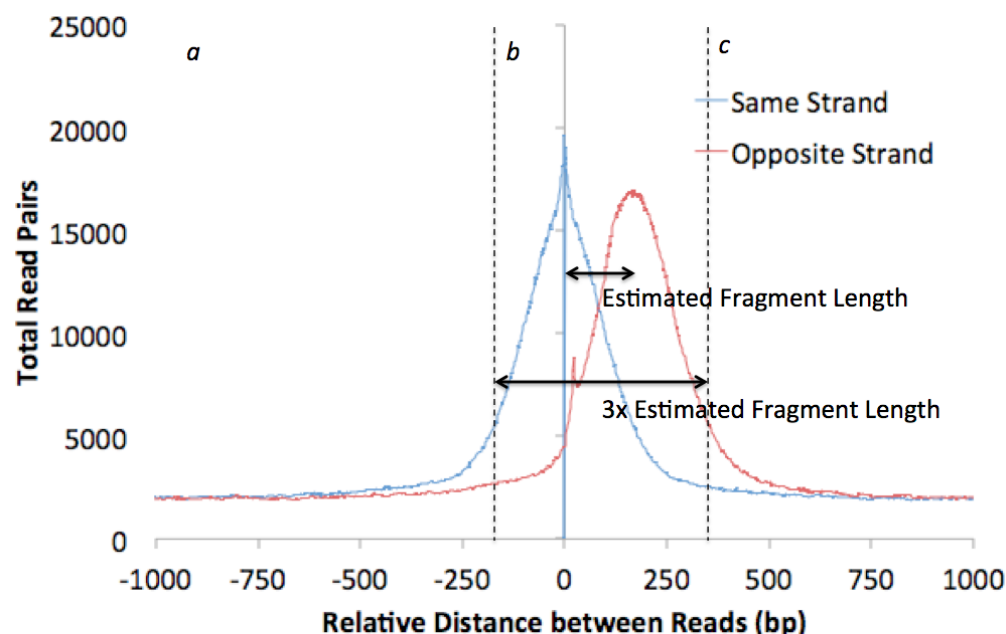
Different types of experiments (i.e. ChIP-Seq vs. DNase-Seq) produce different looking autocorrelation plots, and more detailed discussion of these differences can be found in the individual tutorials. Below is an example from a successful ChIP-Seq experiment:



HOMER also uses the autocorrelation results to guess what type of experiment you conducted. It computes 3 statistics:

- Same strand fold enrichment: Enrichment of reads on the same strand within 3x the estimated fragment length
- Diff strand fold enrichment: Enrichment of reads on different strands within 3x the estimated fragment length
- Same / Diff fold enrichment: Difference between enrichment of reads on the same strand or different strands

Below is a schematic to visualize how these are calculated (keep in mind that the background is calculated out to +/- 2kb):



Same strand and Diff Strand Fold Enrichment: $\frac{\text{Density } b}{\text{Density } a \text{ \& } c}$

Same/Diff Fold Enrichment: $\frac{\text{Density } b \text{ (same strand)}}{\text{Density } b \text{ (Opposite Strand)}}$

Depending on the value of the autocorrelation quality statistics, HOMER will guess what your experiment is:

- If the Same/Diff Fold Enrichment is > 8 fold, it's a great chance the sample is strand-specific RNA. If HOMER decides the sample is probably RNA due to the difference between same strand and different strand numbers, it will automatically set the estimated fragment length to 75 bp. This is because it is difficult to estimate the fragment length for RNA/GRO-seq. To manually set the fragment length, use "**-fragLength <#>**"
- If both the "Same Strand Fold Enrichment" and "Diff Strand Fold Enrichment" are both greater than 1.5 fold, there is good chance you're looking at a working ChIP-Seq experiment.

This is meant as immediate feedback, and not a definitive declaration of the quality or type of your data. We've found it useful to help identify wrong annotations in sample IDs, for example, and helped give us an idea about how well an experiment worked. Good ChIP-Seq antibodies give "Same Strand Fold Enrichment" values well above 5.0 for transcription factors. However, in many cases good results can be extracted out of experiments with lower enrichments, even those that yield a "Guessing sample is ChIP-Seq - may have low enrichment with lots of background" message.

Sequence Bias Analysis:

Invaluable information about a sequencing experiment can be found by examining the relationship between sequencing reads and the genomic sequence they came from. Sequence bias analysis is not performed by default. To perform this analysis, you must provide the *genome* and the "-checkGC" option.

makeTagDirectory <Output Directory Name> [options] -genome <genome> -checkGC <alignment file1> ...
i.e. **makeTagDirectory Macrophage-PU.1-ChIP-Seq -genome mm9 -checkGC pu1.alignment.bed**

This analysis will produce several output files in addition to the basic quality control analysis described above.

An important prerequisite for analyzing sequence bias is that the [appropriate genome must be configured for use with HOMER](#). In version v3.1, HOMER now handles custom/arbitrary genomes. Instead of installing/configuring a genome, you can specify the path to a file or directory containing the genomic sequence in FASTA format. The genome can be in a single FASTA file, or you specify a directory where each chromosome is in a separate file (named chrXXX.fa or chrXXX.fa.masked). In either case, the FASTA headers must contain the chromosome names followed by white space, i.e. ">chr blahblahblah", not ">chr1-blahblahblah", or preferably only ">chr1".

NOTE: If using a sequencing type other than ChIP-Seq or MNase-Seq, you may want to add "-fragLength <#>" since it may be difficult for HOMER to automatically determine the size of fragments used for sequencing.

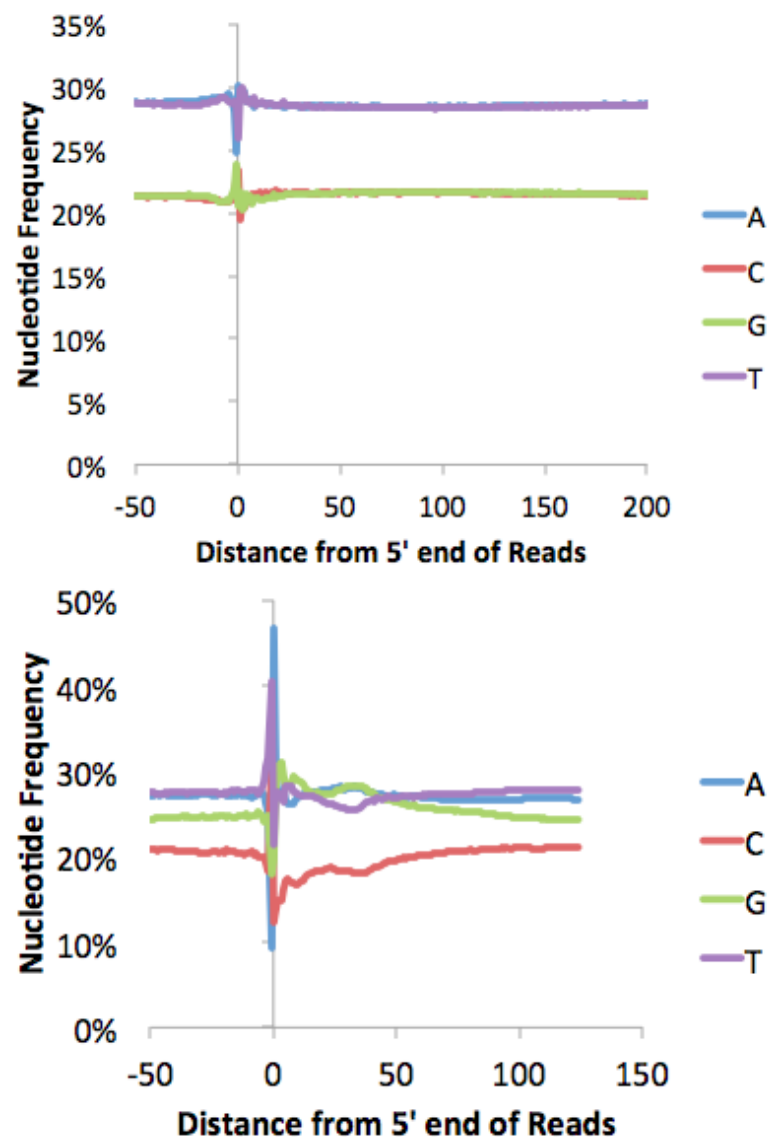
Genomic Nucleotide Frequency relative to read positions

tagFreq.txt - Calculates the nucleotide and dinucleotide frequencies as a function of distance from the 5' end of all reads.

tagFreqUniq.txt - Same as tagFreq.txt, however individual genomic positions are only counted once. If 10 reads mapped to the same position, those nucleotide counts would be added 10 times for "tagFreq.txt", but only once for "tagFreqUniq.txt".

By default, HOMER calculates this frequency from -50 bp to +50 bp relative to the end of the estimated fragment (e.g. not +50bp relative to the 36 bp read, but +50 bp relative to the 200 bp ChIP-fragment). This can be changed using the options "-freqStart <#>" and "-freqEnd <#>".

Below are some examples of this file graphed with EXCEL. One is a pretty typical ChIP-Seq file, the other Chuck wouldn't say where it came from, but he's pretty sure there might be a problem with how they performed their experiment... Quite a bit can be learned from looking at these types of plots (see individual tutorials of additional details)

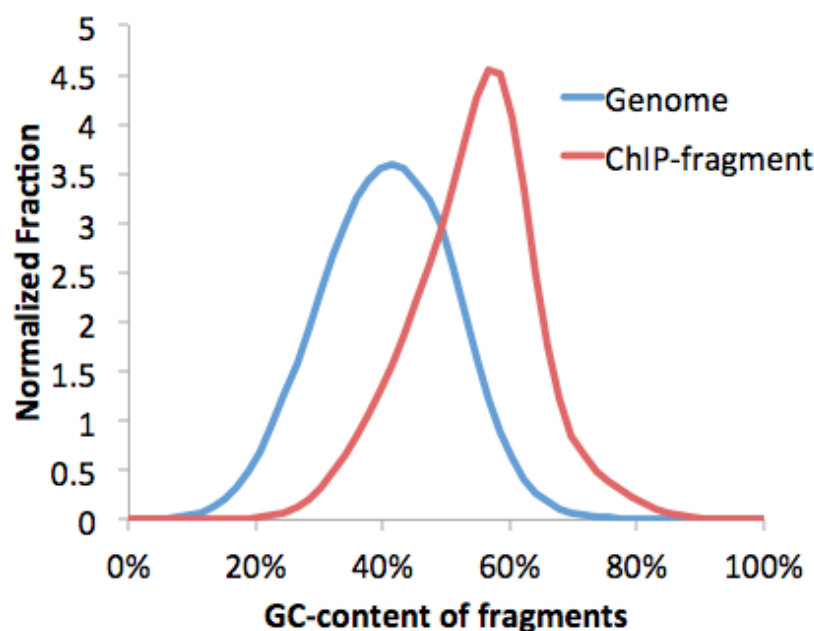
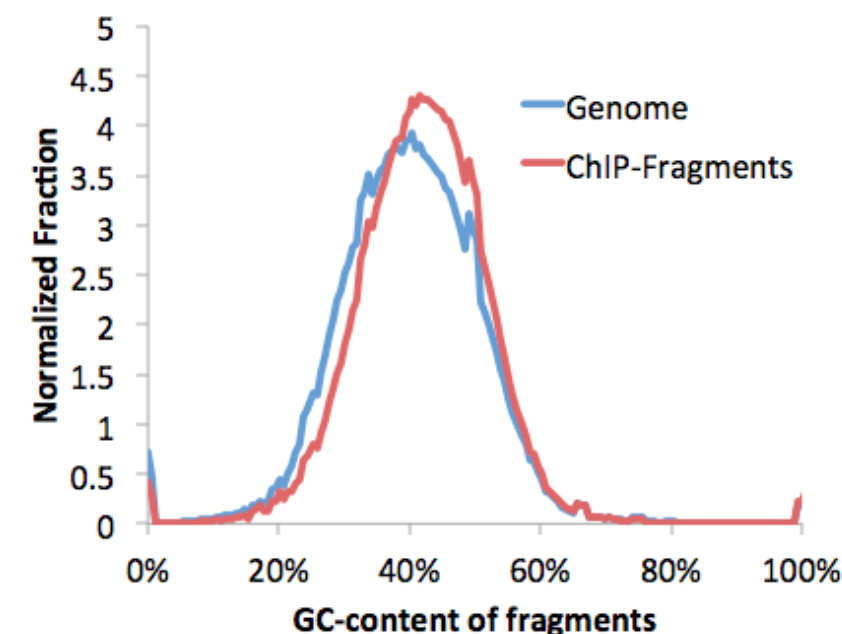


Fragment GC% Distribution

tagGCcontent.txt - Distribution of fragment GC%-content. GC% is calculated from the 5' end position of the read to end of the "fragment", not the end of the read. For ChIP-Seq and MNase-Seq, the fragment length is automatically estimated with pretty high confidence. If you are using another type of sequencing or want to set this manually, be sure to specify the desired fragment length manually ("**-fragLength** <#>").

genomeGCcontent.txt - Distribution of fragment GC%-content at each location in the genome (i.e. expected distribution)

These are very important files to check for any sequencing experiment. Due to the numerous steps of library preparation involving amplification, size selection and gel extraction, it is very easy for the average GC%-content of the sequencing library to "shift". This can be a disastrous problem. Consider the following:



The problem with a GC% shifted sample is that even if the sample is random sequence, you will start to show "enrichment" at places with high GC-content in the genome, such as at CpG Islands. This is unfortunate because most GC-rich areas are at transcription start sites, which might make you think the experiment worked, when in reality the sample was boiled instead of placed in the freezer. See below to learn about GC-normalization.

Sequence Bias GC normalization:

HOMER has built in routines for normalizing GC-bias in a sequencing experiment. In general, normalizing for GC-content is tricky. First, you need to decide if it is worth normalizing at all.

When should I normalize for GC-content in my sequencing experiment?

First you must be able to identify that there is a GC-bias problem. If the average GC% is within 5% of the expected genomic average, you're probably ok. If you are comparing several experiments of similar type, and they all have the same approximate GC-bias, you're also probably better off just comparing the experiments as they are. However, if one of your replicates is much more GC-rich than another, similar sample, you may want to consider normalizing it.

For some experiments, it's tough to know what the expected GC-content should be. For example, if sequencing H3K4me3 ChIP-Seq data, which is typically found near CpG Islands, you may expect the GC% to be higher. However, even with H3K4me3 ChIP-Seq, most ChIP experiments are not very efficient, and most of the DNA being sequenced is background. As a result, the average GC content should not be too far off the expected genomic average. You can always "try" normalizing.

If the GC-bias is severe, you might be better off repeating the experiment. Normalization essentially destroys information, so repeating the experiment (or at least the sequencing library preparation) may be a good move. When re-prepping the sample, try amplifying less and when extracting DNA from gels, dissolve the gels at room temperature.

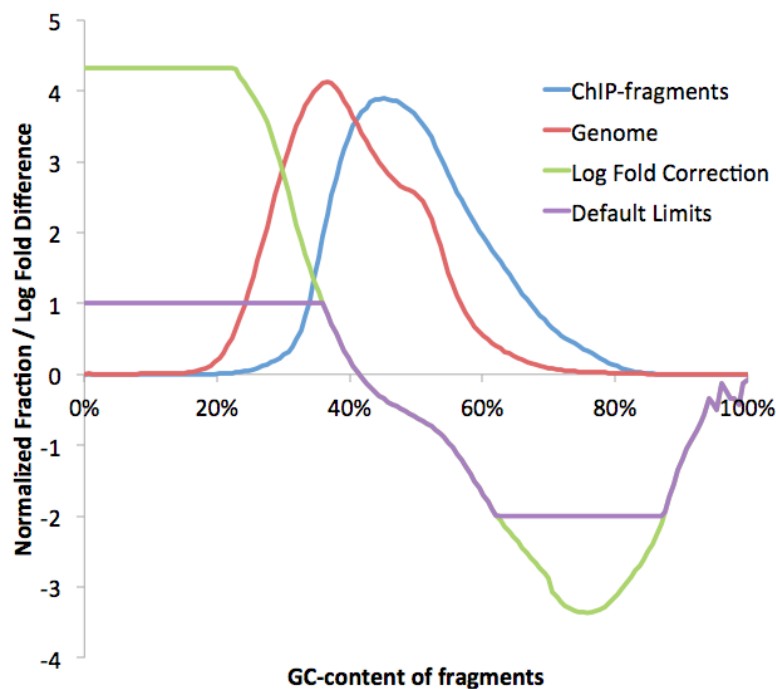
Normalizing tags for GC-bias

To normalize for GC-bias, run the same **makeTagDirectory** command, but you must specify the genome (i.e. "**-genome hg18**") and a target normalization file ("**-normGC <filename>**"). To normalize to the expected genomic average, use "**-normGC default**".

makeTagDirectory <Tag Directory> -genome <genome> -normGC <GC profile file profile> <alignment file 1> [alignment file 2] ...

i.e. makeTagDirectory Macrophage-PU.1-GC -genome mm9 -normGC default aligned.reads.bed

The normalization procedure is actually very simple. HOMER calculates the distribution of fragment GC%-contents, then for each range of GC%, normalizes the tag counts to the expected distribution. If your sample is more GC-rich than expected, reads in GC-rich regions will be reduced to a fractional value (limited by "**-minNormRatio <#>**"). Likewise, if reads are in AT-rich regions, their values will be increased (limited by "**-maxNormRatio <#>**"). HOMER is happy to deal with fractional tag values - there is no reason why a sequencing read can't be counted as a "half-a-read", for example. The biggest problem with **increasing** the value of a read is that it is akin to creating information that doesn't exist, so HOMER puts a tight cap on that adjustment to only 2-fold (change with "**-maxNormRatio <#>**"). The resulting normalization information is recorded in the **tagGCnormalization.txt** file.



Often, the expected genome GC-content is not appropriate. To normalize to a custom GC-profile, provide a file after **-normGC** option ("**-normGC <filename>**"). This file should be formatted just like the **tagGCcontent.txt** files found in the directories. In fact, the idea is that you can normalize one experiment to another by providing the experiments **tagGCcontent.txt** file as the argument for **-normGC**. One potential strategy is to combine several experimental files across different experiments into a single tag directory, and then use the tagGCcontent.txt file from this directory to normalize each of the other experiment, normalizing each experiment to the average from all of the experiments.

Command line options of makeTagDirectory command:

Usage: parseAlignment <directory> <alignment file 1> [file 2] ... [options]

Creates a platform-independent 'tag directory' for later analysis.

Currently BED, eland, bowtie, and sam files are accepted. The program will try to automatically detect the alignment format if not specified.

Existing tag directories can be added or combined to make a new one using -d/-t

If more than one format is needed and the program cannot auto-detect it properly, make separate tag directories by running the program separately, then combine them.

Options:

-fragLength <# | given> (Set estimated fragment length - given: use read lengths)

By default treats the sample as a single read ChIP-Seq experiment

-format <X> where X can be: (with column specifications underneath)

bed - BED format files:
(1:chr,2:start,3:end,4:+/- or read name,5:# tags,6:+/-)
bowtie - output from bowtie (run with --best -k 2 options)
(1:read name,2:+/-,3:chr,4:position,5:seq,6:quality,
7:NA,8:mismatch info)
eland_result - output from basic eland
(1:read name,2:seq,3:code,4:#zeroMM,5:#oneMM,6:#twoMM,7:chr,
8:position,9:F/R,10:-mismatches
eland_export - output from illumina pipeline (22 columns total)
(1-5:read name
info,9:sequence,10:quality,11:chr,13:position,14:strand)
eland_extended - output from illumina pipeline (4 columns total)
(1:read name,2:sequence,3:match stats,4:positions[,])
sam - SAM formatted files (use samTools to covert BAMs into SAM if you
have BAM)
-keep (keep one mapping of each read regardless if multiple equal mappings
exist)
-forceBED (if 5th column of BED file contains stupid values, like mapping quality
instead of number of tags, then ignore this column)
-d <tag directory> [tag directory 2] ... (add Tag directory to new tag directory)
-t <tag file> [tag file 2] ... (add tag file i.e. *.tags.tsv to new tag directory)
-single (Create a single tags.tsv file for all "chromosomes" - i.e. if >100
chromosomes)
-tbp <#> (Maximum tags per bp, default: no maximum)

GC-bias options:
-genome <genome version> (To see available genomes, use "-genome list")
-checkGC (check Sequence bias, requires "-genome")
-freqStart <#> (offset to start calculating frequency, default: -50)
-freqEnd <#> (distance past fragment length to calculate frequency, default:
+50)
-normGC <target GC profile file> (i.e. tagGCcontent.txt file from control
experiment)
Use "-normGC default" to match the genomic GC distribution
-minNormRatio <#> (Minimum deflation ratio of tag counts, default: 0.25)
-maxNormRatio <#> (Maximum inflation ratio of tag counts, default: 2.0)

Next: [Creating UCSC Genome Browser visualization files](#)



Can't figure something out? Questions, comments, concerns, or other feedback:
cbenner@ucsd.edu



HOMER

Software for motif discovery and next-gen sequencing analysis

Visualizing Experiments with the UCSC Genome Browser

The [UCSC Genome Browser](#) is quite possibly one of the best computational tools ever developed. Not only does it contain an incredible amount of data in a single application, it allows users to upload custom information such as data from their ChIP-Seq experiments so that they can be easily visualized and compared to other information.

Making Genome Browser Files

The basic strategy HOMER uses is to create a bedGraph formatted file that can then be uploaded as a custom track to the genome browser. This is accomplished using the **makeUCSCfile** program. To make a ucsc visualization file, type the following:

```
makeUCSCfile <tag directory> -o auto
```

i.e. **makeUCSCfile PU.1-ChIP-Seq/ -o auto**

(output file will be in the PU.1-ChIP-Seq/ folder named PU.1-ChIP-Seq.ucsc.bedGraph.gz)

The "-o auto" with make the program automatically generate an output file name (i.e. TagDirectory.ucsc.bedGraph.gz) and place it in the tag directory which helps with the organization of all these files. The output file can be named differently by specifying "-o outputfilename" or by simply omitting "-o", which will send the output of the program to *stdout* (i.e. add " > outputfile" to capture it in the file outputfile). It is recommended that you zip the file using **gzip** and directly upload the *zipped* file when loading custom tracks at UCSC.

To visualize the experiment in the UCSC Genome Browser, go to Genome Browser [page](#) and select the appropriate genome (i.e. the genome that the sequencing tags were mapped to). Then click on the "add custom tracks" button (this will read "manage custom tracks" once at least one custom track is loaded). Enter the file created earlier in the "Paste URLs or data" section and click "Submit".

Problems Loading UCSC Files

The most common problem encountered while loading UCSC files is to see "position exceeds chromosome length" or something to that effect. This is usually caused by one of two problems:

1. You are trying to load the file to the wrong genome assembly. Make sure the assembly is correct!
2. Some of your tags are mapping outside the reference chromosome - this can be caused by mapping to non-standard assemblies or by some alignment programs. To remove all reads outside of the UCSC chromosome lengths, you can run the program **removeOutOfBoundsReads.pl**.

```
removeOutOfBoundsReads.pl <tag directory> <genome>
```

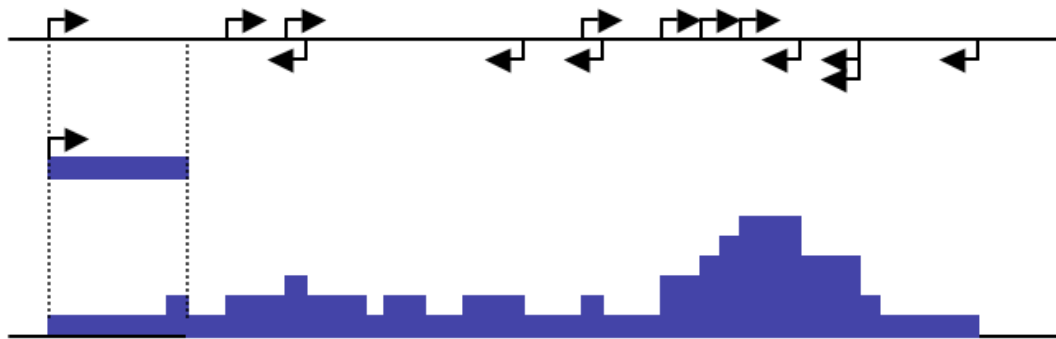
i.e. **removeOutOfBoundsReads.pl PU.1-ChIP-Seq/ mm9**

After running the program, you can rerun **makeUCSCfile**.

What does makeUCSCfile do?

The program works by approximating the ChIP-fragment density at each position in the genome. This is

done by starting with each tag and extending it by the estimated fragment length (determined by [autocorrelation](#), or it can be manually specified using "-fragLength <#>"). The ChIP-fragment density is then defined as the total number of overlapping fragments at each position in the genome. Below is a diagram that depicts how this works:



As great as the UCSC Genome Browser is, the large size of recent ChIP-Seq experiments results in custom track files that are *very* large. In addition to taking a long time to upload, the genome browser has trouble loading excessively large files. To help cope with this, the **makeUCSCfile** program works by specifying a target file size when zipped (default 50MB). In order to meet the specified target file size, makeUCSCfile merges adjacent regions of tag density levels by their weighted average to reduce the total number lines in the final bedGraph file. If you have trouble loading getting your file to load, try reducing the size of the file using the "-fsize <#>" option (i.e. "-fsize 2e7"). To force the creation of larger files, use a very large file size (i.e. "-fsize 1e50") - this will create a file that does not merge any regions and displays a "native" view of the data.

Tags can be visualized separately for each strand using the "-strand separate" option.

Changing the Resolution

By default, makeUCSCfile uses the "-fsize <#>" option to determine how many reads to essentially "skip" when making the output file. You can also manually set the resolution.

In an effort to reduce the size of large UCSC files, one attractive option is to reduce the overall resolution of the file. By default, **makeUCSCfile** will make full resolution (i.e. 1 bp) files, but this can be changed by specifying the "-res <#>" option. For example, "-res 10" will cause changes in ChIP-fragment density to be reported only every 10 bp.

Normalization of UCSC files

In order to easily compare ChIP-fragment densities between different experiments, **makeUCSCfile** will normalize density profiles based on the total number of mapped tags for each experiment. As with other programs apart of HOMER, the total number of tags is normalized to 10 million. This means that tags from an experiment with only 5 million mapped tags will count for 2 tags apiece. The total of tags to normalize to can be changed using the "-norm <#>" option.

Separating data from different strands / RNA-Seq

You can specify that HOMER separate the data based on the strand by using the "-strand <...>" option. This is useful when looking at strand-specific RNA-Seq/GRO-Seq experiments. The following options are available:

- strand both : default behavior, for ChIP-Seq/MNase-Seq etc.
- strand separate : separate data by strand, for RNA-Seq/GRO-Seq
- strand + : only show the positive strand (i.e. Watson strand) data
- strand - : only show the negative strand (i.e. crick strand) data

Creating bigWig files with HOMER

Some data sets of very large, but you still want to see all of the details from your sequencing in the UCSC Genome Browser. HOMER can produce [bigWig files](#) by running the conversion program for you (**bedGraphToBigWig**). The only catch is that [you must have access to a webserver](#) where you can post the resulting bigWig file - this is because instead of uploading the whole file to UCSC, the browser actually looks for the data file on YOUR webserver and grabs only the parts it needs. Slick, eh. Chuck uses this all the time for big experiments.

Before even trying to make bigWigs, you must download the [bedGraphToBigWig program from UCSC](#) and place it somewhere in your executable path (i.e. the /path-to-homer/bin/ folder). This called directly by HOMER to create the BigWig files.

Using the makeBigWig.pl Script

To make bigWig files easier to generate, HOMER includes a program creatively named "makeBigWig.pl" that automates all of the steps below.

```
makeBigWig.pl <tag directory> <genome> [special options] [makeUCSC file options] -
webDir /path-to-web-fold/ -url http://webserverURL/bigwigFold/
i.e. makeBigWig.pl PU.1-ChIP-Seq/ mm9 -webDir /var/www/bigWigs/ -url
http://ChuckNorrisU.edu/bigWigs/
```

If you are visualizing strand specific data (i.e. RNA-Seq), specify "-strand". The -url and -webDir are the directories are the web URL directory and file system directory where the bigWigs will be stored, respectively. Recent changes to UCSC require that the chromosome sizes be specified exactly. If having trouble, the current version of HOMER has the option "-chromSizes <filename>" so that you can specify the sizes explicitly.

Making bigWigs from scratch

This is a quick description of what HOMER is doing. To make a bigWig, add the "-bigWig <chrom.sizes file> -fsize 1e20" parameters to your makeUCSCfile command. When making a bigWig, you usually want to see all of the tag information, so make sure the "-fsize" options is large. You also need to specify an output file using "-o <bigwigfilename>" and also capture the stdout stream using "> trackfileoutput.txt". You can also use "-o auto". The "trackfileoutput.txt" will contain the header information that is uploaded as a custom track to UCSC. Recently, changes to UCSC require that HOMER know the exact size of the chromosomes when making the file - these should be placed in a file (<chrom.sizes> file). **makeBigWig.pl** and **makeMultiWigHub.pl** will generate these files automatically by analyzing the sequences in the genome directory.

After running the makeUCSCfile program with the bigWig options, you need to do the following:

1. Copy the *.bigWig file to your webserver location and make sure it is viewable over the internet.
2. Need to edit the "trackfileoutput.txt" file and enter the URL of your bigWig file (... bigDataUrl=http://server/path/bigWigFilename ...)
3. Upload the "trackfileoutput.txt" file to UCSC as a custom track to view your data.

For example:

```
makeUCSCfile <tag directory> -o auto -bigWig <chrom.sizes file> -fsize 1e20 >
trackInfo.txt
```

i.e.

```
makeUCSCfile PU.1-ChIP-Seq/ -o auto -bigWig chrom.sizes -fsize 1e20 > PU.1-
bigWig.trackInfo.txt
cp PU.1-ChIP-Seq/PU.1-ChIP-Seq.ucsc.bigWig /Web/Server/Root/Path/
** edit PU.1-bigWig.trackInfo.txt to have the right URL **
```

NOTE: As of now, a bigWig file can only be composed of a single track - if you want to separate the data by strands, do the following:

```
makeUCSCfile PU.1-ChIP-Seq/ -o PU.1.positiveStrand.bigWig -bigWig chrom.sizes -fsize
1e20 -strand + > PU.1-bigWig.trackInfo.positiveStrand.txt
makeUCSCfile PU.1-ChIP-Seq/ -o PU.1.negativeStrand.bigWig -bigWig chrom.sizes -
fsize 1e20 -strand - > PU.1-bigWig.trackInfo.negativeStrand.txt
cp PU.1.positiveStrand.bigWig PU.1.negativeStrand.bigWig /Web/Server/Root/Path/
cat PU.1-bigWig.trackInfo.positiveStrand.txt PU.1-bigWig.trackInfo.negativeStrand.txt >
PU.1-bigWig.trackInfo.both.txt
** edit PU.1-bigWig.trackInfo.both.txt to have the right URLs for both the negative and positive
strands **
```

Creating Multi-Experiment Overlay Tracks

UCSC has recently added the option to create overlay tracks, where several bigWig files can be viewed in the same space with the help of transparent colors. The first example of this was the Encode Regulation Track, which showed H3K4me1/3 data from several cell types at the same time. This is very useful for large-scale data sets will many different experiments. In these cases it is just about impossible to get them on the screen together.

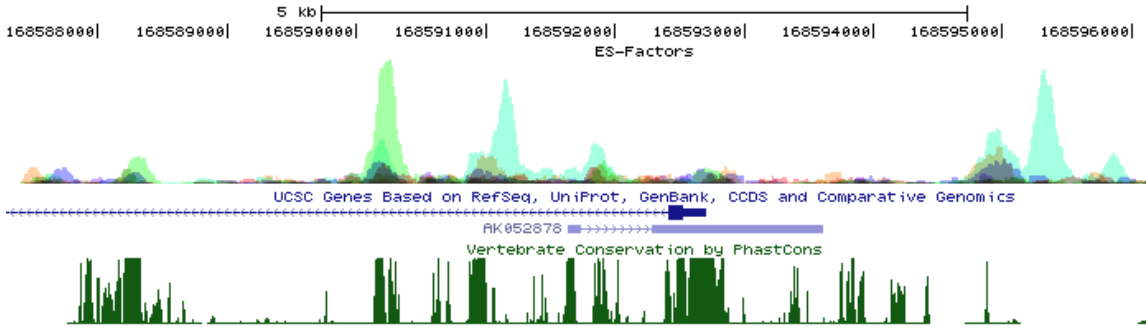
To make a "multi-wig hub", as we will refer to them, you need to make sure you have the [bedGraphToBigWig program from UCSC](#), and a working webserver to host your files. If you can handle bigWigs in the section above, you can make multi-wig hubs.

The HOMER program to handle multi-wig hubs is called **makeMultiWigHub.pl**. It works essentially the same way as the **makeBigWig.pl** script, however, the syntax is a little different. The basic usage is:

```
makeMultiWigHub.pl <hub name> <genome> [options] -d <tag directory1> <tag directory2> ...
i.e. makeMultiWigHub.pl ES-Factors mm9 -d mES-Oct4/ mES-Sox2/ mES-Nanog/ mES-Klf4/
mES-Esrrb/ mES-cMyc/ mES-Stat3/
```

NOTE: make sure you use the UCSC genome (e.g. mm9) and not the masked, bastardized HOMER version (mm9r).

The above example will produce a hub called "ES-Factors", composed of configuration files and bigWig files, and place it on your server in the directory specified by "**-webDir <directory>**". It will also provide you with a URL to the hub (dependent on the value of "**-url <base url>**"). To load the Hub, click on "Track Hubs" on the UCSC browser (next to custom tracks button), and paste the URL in to the dialog box. The example above will look something like this:



To figure out which factors correspond to which colors, click on the Blue Heading for the Hub in the settings area below the UCSC picture. Something like this should pop up:

ES-Factors Track Settings

ES-Factors

Display mode: Full Submit [Reset to defaults](#)

Overlay method: transparent

Type of graph: bar

Track height: 75 pixels (range: 11 to 100)

Vertical viewing range: min: 0 max: 127 (range: 0 to 127)

Data view scaling: auto-scale to data view Always include zero: OFF

Transform function: Transform data points by: NONE

Windowing function: maximum Smoothing window: OFF pixels

Draw y indicator lines: at y = 0.0: OFF at y = 0 OFF

[Graph configuration help](#)

List subtracks: only selected/visible all (7 of 7 selected)

☒

mES-cMyc

mES-cMyc

☒

mES-Esrrb

mES-Esrrb

☒

mES-Klf4

mES-Klf4

☒

mES-Nanog

mES-Nanog

☒

mES-Oct4

mES-Oct4

☒

mES-Sox2

mES-Sox2

☒

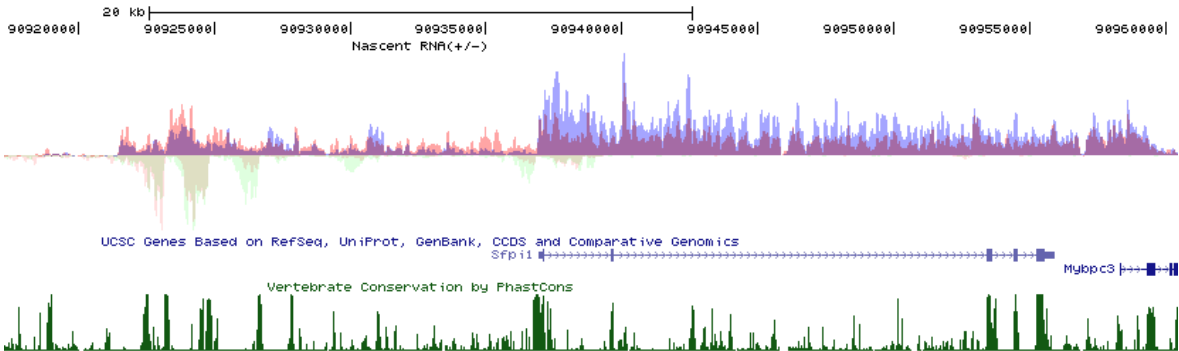
mES-Stat3

mES-Stat3

7 of 7 selected

Unfortunately, as of now editing hub information can only be done by directly modifying the hub files on the server. For example, to edit to colors, you must edit the `"/webserver/directory/hubName/genome/trackDB.txt"` file.

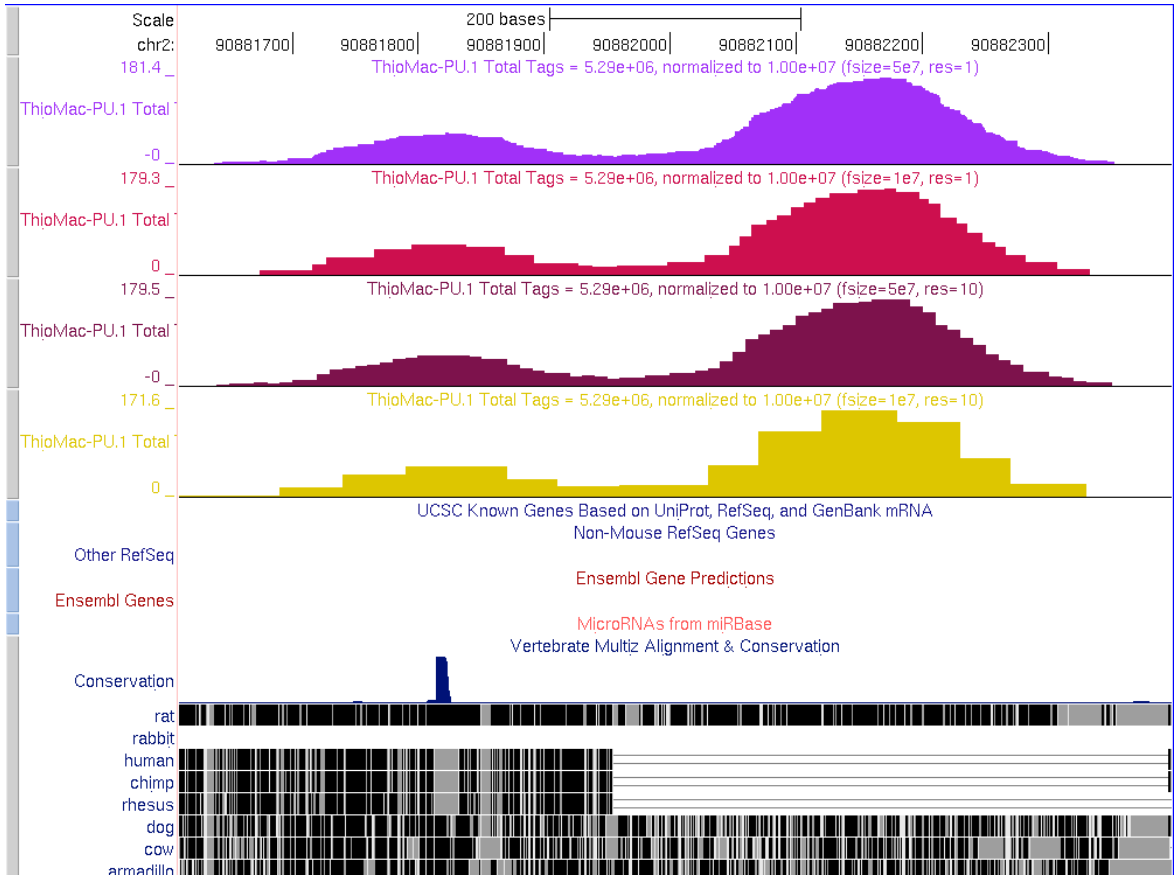
Because Hubs are so cool, HOMER will also do +/- strand RNA data right. Unfortunately, for now you can't mix stranded and non-stranded data in the same hub with the `makeMultiWigHub.pl` program. To visualize stranded information, add `"-strand"`. Below is an example:



Examples of UCSC bedGraph files

The following shows what the same data set looks like changing options for file size (`-fsize`) and resolution (`-res`). Usually it's best to use one or the other.

1. `-fsize 5e7 -res 1`
2. `-fsize 1e7 -res 1`
3. `-fsize 5e7 -res 10`
4. `-fsize 1e7 -res 10`



Command line options for makeUCSCfile

Usage: makeUCSCfile <tag directory> [options]

Creates a bedgraph file for visualization using the UCSC Genome Browser

General Options:

- fsize <#> (Size of file, when gzipped, default: 5e7)
- strand <both|separate|+|-> (control if reads are separated by strand, default: both)
- fragLength <# | auto | given> (Approximate fragment length, default: auto)
- adjust <#> (Adjust edge of tag 3' by # bp, negative for 5', default: none[good for dnase])
- tbp <#> (Maximum tags per bp to count, default: no limit)
- res <#> (Resolution, in bp, of file, default: 1)
- lastTag (To keep ucsc happy, last mapped tag is NOT extended by default
Using this option will allow extending of data past the last tag position)
- norm <#> (Total number of tags to normalize experiment to, default: 1e7)
- noadj (Do not normalize tag counts)
- neg (plot negative values, i.e. for - strand transcription)
- CpG (Show unmethylated CpG ratios)
- color <(0-255),(0-255),(0-255)> (no spaces, rgb color for UCSC track, default: random)
- bigWig <chrom.sizes> (creates a full resolution bigWig file and track line file
This requires bedGraphToBigWig to be available in your executable path
Also, because how bigWig files work, use "-strand -" and "-strand +"
in separate runs to make strand specific files: "-strand separate" will not work)
- o <filename|auto> (send output to this file - will be gzipped, default: prints to stdout)
auto: this will place an appropriately named file in the tag directory
- name <...> (Name of UCSC track, default: auto generated)
- style <option> (See options below:)
 - chipseq (standard, default)
 - rnaseq (strand specific)
 - tss (strand specific, single bp fragment length)

dnase (fragments centered on tag position instead of downstream)
methylated (single bp resolution of cytosine methylation)
unmethylated (single bp resolution of unmethylated cytosines)
-circos <chrN:XXX-YYY|genome> (output only a specific region for circos[no header])

Command line options for makeBigWig.pl

Script for automating the process of creating bigWigs

Usage: makeBigWig.pl <tag directory> <genome> [special options] [options]

Special Options for bigWigs [choose one, don't combine]:

- normal (ChIP-Seq style, default)
- strand (Strand specific, for RNA-Seq and GRO-Seq)
- dnase (Special options for Crawford-lab style DNase-Seq)
- cage (Special options for CAGE/TSS-Seq)
- cpg (Special options for mCpG/CpG)

Other options:

Whatever options you want to pass to makeUCSCfile
!!Warning!!: do not try to specify "-strand separate" - use the special option above.

File options:

- fsize <#> (Use to limit the size of the bigwig files)
- url <URL> (URL directory -no filename- to tell UCSC where to look)
- webdir <directory> (name of directory to place resulting bigWig file)
- update (overwrite bigwigs in the webDir directory, otherwise random numbers are added to make the file unique).

Current url target (-url): <http://biowhat.ucsd.edu/bigWig/>

Current web directory (-webDir): /data/www/bigWig/

You're going to want to modify the \$wwwDir and \$httpDir variables at the top of the makeBigWig.pl program file to accomodate your system so you don't have to specify -url and -webdir all the time.

Command line options for makeMultiWigHub.pl

Script for automating the process of creating multiWig tracks

Usage: makeMultiWigHub.pl <hubname> <genome> [options] -d <tag directory1> [tag directory2]...

Special Options for bigWigs [choose one, don't combine]:

- normal (ChIP-Seq style, default)
- strand (Strand specific, for RNA-Seq and GRO-Seq)
- dnase (Special options for Crawford-lab style DNase-Seq)
- cage (Special options for CAGE/TSS-Seq)
- cpg (Special options for mCpG/CpG)

Other options:

Whatever options you want to pass to makeUCSCfile
!!Warning!!: do not try to specify "-strand separate" - use the special option above.
Also, for the genome, do NOT use repeat version (mm9r) - use mm9 instead

File options:

- force (overwrite existing hub)
- fsize <#> (limit the file size of the bigwig files to this value)

- url <URL> (URL directory -no filename- to tell UCSC where to look)
- webdir <directory> (name of directory to place resulting hub directory)

Current url target (-url): http://biowhat.ucsd.edu/hubs/
Current web directory (-webDir): /data/www/hubs/

You're going to want to modify the \$wwwDir and \$httpDir variables at the top of the makeMultiWigHub.pl program file to accomidate your system so you don't have to specify -url and -webdir all the time.

Next: [Finding Peaks \(ChIP-enriched regions\) in the genome](#)



Can't figure something out? Questions, comments, concerns, or other feedback:
cbenner@ucsd.edu



HOMER

Software for motif discovery and next-gen sequencing analysis

Finding Enriched Peaks, Regions, and Transcripts

HOMER contains a program called **findPeaks** that performs all of the peak calling and transcript identification analysis. (Not to be confused with another peak finding program called [FindPeaks](#), which was also very creatively named). Finding peaks is one of the central goals of any ChIP-Seq experiment, and the same basic principles apply to other types of sequencing such as DNase-Seq. The basic idea is to identify regions in the genome where we find more sequencing reads than we would expect to see by chance. There are number of different approaches one can use to find peaks, and correspondingly there are many different methods for identifying peaks from ChIP-Seq experiments. It is not required that you use HOMER for peak finding to use the rest of the tools included in HOMER ([see below](#)).

findPeaks has 3 basic modes of operation:

factor

Peak finding for single contact or focal ChIP-Seq experiments or DNase-Seq. This type of analysis is useful for transcription factors, and aims to identify the precise location of DNA-protein contact. This type of peak finding uses a FIXED width peak size, which is automatically estimated from the Tag Autocorrelation.

histone

Peak finding for broad regions of enrichment found in ChIP-Seq experiments for various histone marks. This analysis finds variable-width peaks.

groseq

De novo transcript identification from strand specific GRO-Seq. This attempts to identify transcripts from nascent RNA sequencing reads.

HOMER does not perform de novo transcript isoform detection from spliced RNA-Seq. As we have just started analyzing RNA-Seq, and there is already a bunch of great work on this topic, there's no need to reinvent the wheel for this type of analysis. We recommend the [Tophat/Cufflinks](#) family of programs for RNA-Seq isoform detection.

Using findPeaks

To run **findPeaks**, you will normally type:

```
findPeaks <tag directory> -style <factor|histone|groseq> -o auto -i <control tag directory>
```

i.e. **findPeaks ERalpha-ChIP-Seq/ -style factor -o auto -i Control-ChIP-Seq/**

Where the first argument must be the tag directory (required). The other options are not required. The "**-style <...>**" option can be either "**factor**", "**histone**", or "**groseq**". Use the "**-i**" option to specify a control experiment tag directory (good idea when doing ChIP-Seq).

Output files

Use the "**-o <filename>**" to specify where to send the resulting peak file. If "**-o**" is not specified, the peak file will be written to **stdout**.

If "**-o auto**" is specified, the peaks will be written to:

```
"<tag directory>/peaks.txt" (-style factor)
"<tag directory>/regions.txt" (-style histone)
"<tag directory>/transcripts.txt" and "<tag directory>/transcripts.gtf" (-style groseq)
```

The top portion of the peak file will contain parameters and various analysis information. This output differs somewhat for GRO-Seq analysis, and is explained in more detail later. Some of the values are self explanatory. Others are explained below:

```
# HOMER Peaks
# Peak finding parameters:
# tag directory = Sox2-ChIP-Seq
#
# total peaks = 10280
# peak size = 137
```

```
# peaks found using tags on both strands
# minimum distance between peaks = 342
# fragment length = 132
# genome size = 4000000000
# Total tags = 9908245.0
# Total tags in peaks = 156820.0
# Approximate IP efficiency = 1.58%
# tags per bp = 0.001907
# expected tags per peak = 0.523
# maximum tags considered per bp = 1.0
# effective number of tags used for normalization = 10000000.0
# Peaks have been centered at maximum tag pile-up
# FDR rate threshold = 0.001000
# FDR effective poisson threshold = 0.000000
# FDR tag threshold = 8.0
# number of putative peaks = 10800
#
# size of region used for local filtering = 10000
# Fold over local region required = 4.00
# Poisson p-value over local region required = 1.00e-04
# Putative peaks filtered by local signal = 484
#
# Maximum fold under expected unique positions for tags = 2.00
# Putative peaks filtered for being too clonal = 36
#
# cmd = findPeaks Sox2-ChIP-Seq -style factor -o auto
#
# Column Headers:
```

Genome size represents the total effective number of mappable bases in the genome (remember each base could be mapped in each direction)

Approximate IP efficiency describes the fraction of tags found in peaks versus genomic background. This provides an estimate of how well the ChIP worked. Certain antibodies like H3K4me3, ERa, or PU.1 will yield very high IP efficiencies (>20%), while most rand in the 1-20% range. Once this number dips below 1% it's a good sign the ChIP didn't work very well and should probably be optimized.

Below the header information are the peaks, listed in each row. Columns contain information about each peak:

- Column 1: PeakID - a unique name for each peak (very important that peaks have unique names...)
- Column 2: chr - chromosome where peak is located
- Column 3: starting position of peak
- Column 4: ending position of peak
- Column 5: Strand (+/-)
- Column 6: Normalized Tag Counts - number of tags found at the peak, normalized to 10 million total mapped tags (or defined by the user)
- Column 7: (-style factor): Focus Ratio - fraction of tags found appropriately upstream and downstream of the peak center. (see below)
- (-style histone/-style groseq): Region Size - length of enriched region
- Column 8: Peak score (position adjusted reads from initial peak region - reads per position may be limited)
- Columns 9+: Statistics and Data from filtering

Two generic tools are available as part of HOMER to convert peak files to BED files and back. This will allow you to upload your peak files to the UCSC Genome Browser, or convert peak files in BED format from another program into a peak file that can be used by HOMER. These programs are named **pos2bed.pl** and **bed2pos.pl**, which can be used the following way:

```
pos2bed.pl peakfile.txt > peakfile.bed
bed2pos.pl peakfile.bed > peakfile.txt
```

Finding Transcription Factor Peaks with HOMER

To find peaks for a transcription factor use the **findPeaks** command:

```
findPeaks <tag directory> -style factor -o auto -i <input tag directory>
i.e. findPeaks ERalpha-ChIP-Seq/ -style factor -o auto -i MCF7-input-ChIP-Seq
```

Identification of Putative Peaks

If **findPeaks** is run in "factor" mode, a fixed peak size is selected based on estimates from the autocorrelation analysis performed during the **makeTagDirectory** command. This type of analysis maximizes sensitivity for identifying locations where the factor makes a single contact with the DNA. Peak size can be set manually with "-size <#>".

findPeaks loads tags from each chromosome, adjusting them to the center of their fragments, or by half of the estimated fragment length in the 3' direction (this value is also automatically estimated from the autocorrelation analysis). The fragment length can be specified manually using the "-fragLength <#>" option. It then scans the entire genome looking for fixed width clusters with the highest density of tags. As clusters are found, the regions immediately adjacent are excluded to ensure there are no "piggyback peaks" feed off the signal of large peaks. By default, peaks must be greater than 2x the peak width apart from on another (set manually with "-minDist <#>"). This continues until all tags have been assigned to clusters.

After all clusters have been found, a tag threshold is established to correct for the fact that we may expect to see clusters simply by random chance. Previously, to estimate the expected number of peaks for each tag threshold, HOMER would randomly assign tag positions and repeat the peak finding procedure. HOMER now assumes the local density of tags follows a Poisson distribution, and uses this to estimate the expected peak numbers given the input parameters much more quickly. Using the expected distribution of peaks, HOMER calculates the expected number of false positives in the data set for each tag threshold, setting the threshold that beats the desired False Discovery Rate specified by the user (default: 0.001, "-fdr <#>").

HOMER assumes the total number of mappable base pairs in the genome is 2,000,000,000 bp (** change from previous version. here 2e9 assumes the actual number of mappable positions is actually 2x [think + and - strand]) , which is "close enough" for human and mouse calculations. You can specify a different genome size using "-gsize <#>". HOMER also uses the reads themselves to estimate the size of the genome (i.e. that highest tag position on each chromosome). If this estimate is lower than the default, it will use that value to avoid using too large of a number on smaller genomes (For example, if you used findPeaks on *drosophila* data without specifying "-gsize").

It is important to note that this false discovery rate controls for the random distribution of tags along the genome, and not any other sources of experimental variation. Alternatively, users can specify the threshold using "-poisson <#>" to calculate the tag threshold that yields a cumulative poisson p-value less than provided or "-tagThreshold <#>" to specify a specific number tags to use as the threshold.

Filtering Peaks

The initial step of peak finding is to find non-random clusters of tags, but in many cases these clusters may not be representative of true transcription factor binding events. To increase the overall quality of peaks identified by HOMER, 3 separate filtering steps can be applied to the initial, putative peaks identified:

Using Input/IgG Sequencing as a Control

To use an Input or IgG sequencing run as a control (**HIGHLY RECOMMENDED**), you must first create a separate tag directory for the input experiment (see here). Additionally, you can use other cleaver experiments as a control, such as a ChIP-Seq experiment for the same factor in another cell or in a knockout. To find peaks using a control, type:

findPeaks <tag directory> -style factor -i <control tag directory> -o auto

i.e. **findPeaks ERalpha-ChIP-Seq/ -style factor -i Input-ChIP-Seq/ -o auto**

HOMER uses two parameters to filter peaks against a control experiment. First, it uses a fold change (which is sequencing depth-independent), requiring each putative peak to have 4-fold more normalized tags in the target experiment than the control (or specify a different fold change with "-F <#>"). In the case where there are no input tags near the putative peak, HOMER automatically sets these regions to be set to the average input tag coverage to avoid dividing by zero. HOMER also uses the poisson distribution to determine the chance that the differences in tag counts are statistically significant (sequencing-depth dependent), requiring a cumulative poisson p-value of 0.0001 (change with "-P <#>"). This effectively removes peaks with low tag counts for which there is a chance the differential enrichment is found simply due to sampling error.

One modification in recent versions of HOMER is the in the size of region used to compare experiment with control tags. Since control experiments are not always performed the same way, e.g. different fragment lengths, it helps to enlarge the peak size for the purposes of comparing experiments to ensure control reads found immediately outside of the peak region are still considered. By default, HOMER enlarges peaks by 2x to search the control experiment (change with "-inputSize <#>"). This may reduce specificity when trying to identify certain types of peaks.

Filtering Based on Local Signal

Our experience with peak finding is that often putative peaks are identified in regions of genomic duplication, or in regions where the reference genome likely differs from that of the genome being sequenced. This produces large regions of high tag counts, and if no Input/IgG sample is available, it can be hard to exclude these regions. Also, it may be advantageous to remove putative peaks that a spread out over larger regions as it may be difficult to pin-

point the important regulatory regions within them.

To deal with this, HOMER will filter peaks based on the local tag counts (similar in principle to MACS). By default, HOMER requires the tag density at peaks to be 4-fold greater than in the surrounding 10 kb region. This can be modified using `"-L <#>"` and `"-localSize <#>"` to change the fold threshold and size of the local region, respectively. As with input filtering, the comparison must also pass a poisson p-value threshold of 0.0001, which can be set using `"-LP <#>"` option.

Filtering Based on Clonal Signal

When we first sifted through peaks identified in ChIP-Seq experiments we noticed there are many peaks near repeat elements that contain odd tag distributions. These appear to arise from expanded repeats that result in peaks with high numbers of tags from only a small number of unique positions, even when many of the other positions within the region may be "mappable". To help remove these peaks, HOMER will compare the number of unique positions containing tags in a peak relative to the expected number of unique positions given the total number of tags in the peak. If the ratio between the later and the former number gets too high, the peak is discarded. The fold threshold can be set with the `"-C <#>"` option (default: `"-C 2"`). HOMER uses the **averageTagsPerPosition** parameter in the `tagInfo.txt` file to adjust this calculation as to not over-penalize ChIP-Seq experiments that are already highly "clonal". If analyzing MNase or other restriction enzyme digestion experiments turn this option off (`"-C 0"`);

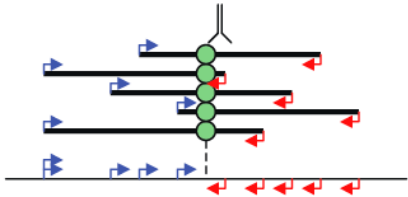
Disabling Filtering

To disable Input, Local, or Clonal filtering set any combination of `"-F 0 -L 0 -C 0"`.

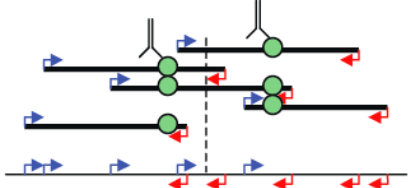
Peak Centering and Focus Ratios

If the option `"-style factor"` or `"-center"` is specified, **findPeaks** will calculate the position within the peak with the maximum ChIP-fragment overlap and calculate a **focusRatio** for the peak. This is not always desired (such as with histone modifications). The focus ratio is defined as the ratio of tags located 5' of the peak center on either strand relative to the total number of tags in the peak. Peaks that contain tags in the ideal positions are more likely to be centered on a single binding site, and these peaks can be used to help determine what sequences are directly bound by a transcription factor. Unfocused peaks, or peaks with low (i.e. <80%) focusRatios may be the result of several closely spaced binding sites or large complexes that cross-link to multiple positions along the DNA. Sometimes this means you have more than one binding site in close proximity (example below), but other times it means you cross-linked the \$#& out of your cells, or you have background in your sample.

Focused Peaks:



Unfocused Peaks:



To isolate focused peaks, you can use the `getFocalPeaks.pl` tool:

```
getFocalPeaks.pl <peak file> <focus % threshold> > focalPeaksOutput.txt
```

i.e. `getFocalPeaks.pl ERpeaks.txt 0.90 > ERfocalPeaks.txt`

Finding Enriched Regions of Variable Length

To find variable length peaks for histone marks, use the **findPeaks** command:

```
findPeaks <tag directory> -style histone -o auto -i <input tag directory>
```

i.e. `findPeaks H3K4me1-ChIP-Seq/ -style histone -o auto -i Input-ChIP-Seq`

If the option **"-style histone"** or **"-region"** is specified, **findPeaks** will stitch together enriched peaks into regions. Note that [local filtering](#) is turned off when finding regions. The most important parameters for region finding are the **"-size"** and **"-minDist"** (and of course the fragment length). First of all, **"-size"** specifies the width of peaks that will form the basic building blocks for extending peaks into regions. Smaller peak sizes offer better resolution, but larger peak sizes are usually more sensitive. By default, **"-style histone"** evokes a peak size of 500.

The second parameter, **"-minDist"**, is usually used to specify the minimum distance between adjacent peaks. If **"-region"** is used, this parameter then specifies the maximum distance between putative peaks that is allowed if they are to be stitched together to form a region. By default this is 2x the peak size. If you think about histone modifications, the signal is never continuous in enriched regions, with reduced signal due to non-unique sequences (that can't be mapped to) and nucleosome depleted regions. **"-minDist"** informs **findPeaks** how big of a gap in the signal will be tolerated for adjacent peaks to be considered part of the same region. (by default **"-style histone"** sets this to 1000).

One thing to note is that you may have to play around with these parameters to get the results you want. If you look at the examples below, you could make arguments for using each of the tracks given what you're interested in and how you would define a "region".

For example: (in the example below, the default size comes from the autocorrelation estimate for the Macrophage-H3K4me1 dataset)

```
Default Parameters:
findPeaks Macrophage-H3K4me1/ -i Macrophage-IgG -size 150 -minDist 370 > output.txt (i.e. defaults)
```

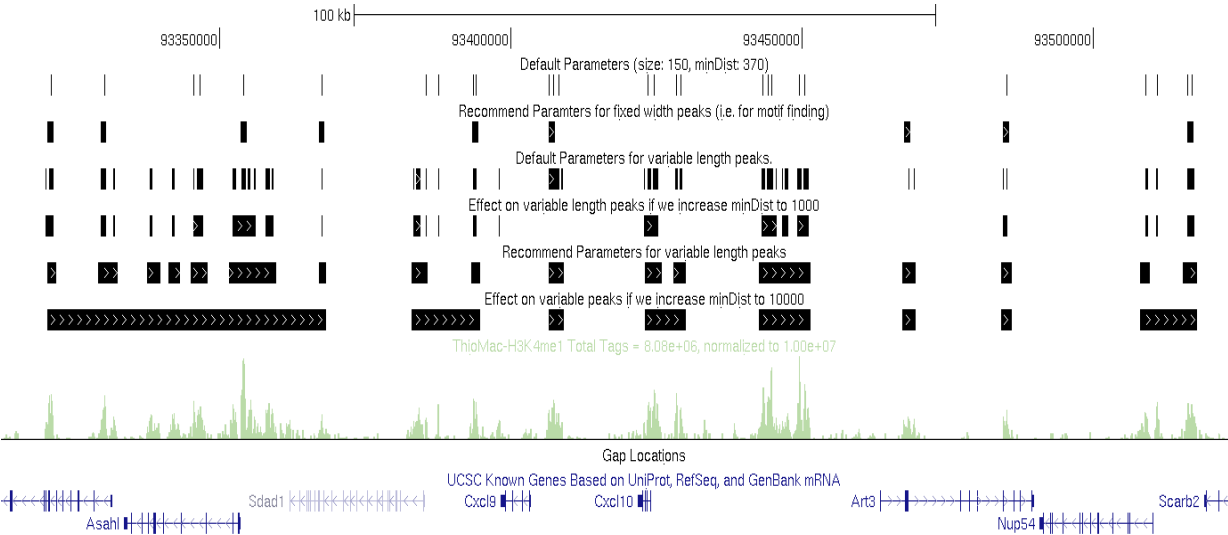
```
Recommend Parameters for fixed width peaks (i.e. for motif finding):
findPeaks Macrophage-H3K4me1/ -i Macrophage-IgG -size 1000 -minDist 2500 > output.txt
```

```
Default Parameters for variable length peaks.
findPeaks Macrophage-H3K4me1/ -i Macrophage-IgG -region -size 150 -minDist 370 > output.txt
```

```
Effect on variable length peaks if we increase minDist to 1000.
findPeaks Macrophage-H3K4me1/ -i Macrophage-IgG -region -size 150 -minDist 1000 > output.txt
```

```
Recommend Parameters for variable length peaks (H3K4me1 at least).
findPeaks Macrophage-H3K4me1/ -i Macrophage-IgG -region -size 1000 -minDist 2500 > output.txt
```

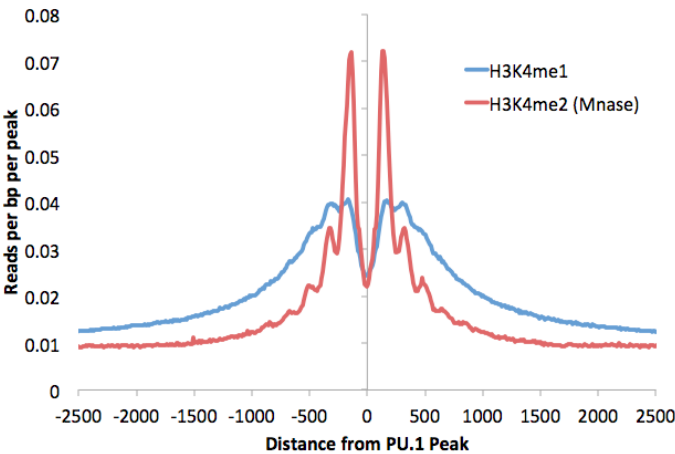
```
Effect on variable peaks if we increase minDist to 10000 (H3K4me1 at least).
findPeaks Macrophage-H3K4me1/ -i Macrophage-IgG -region -size 1000 -minDist 10000 > output.txt
```



Finding Histone Modification Peaks and Motif Finding

Finding peaks using histone modification data can be a little tricky - largely because we have very little idea what the histone marks actually do. If you want to find peaks in histone modification data with the purpose of analyzing them for enriched motifs, read this section. The problem with histone modification data (and some other types) is that the signal can spread over large distances. Trying to analyze large, variable length regions for motif enrichment is very difficult and not recommended. As such it is recommended sometimes a better idea to use fixed-size peak finding on histone marks (i.e. H3K4me1 enhancer mark) to improve motif analysis results. Using fixed size peaks helps make sure the assumptions needed for Motif Finding are, let's say, *less violated*, when dealing with regions of constant size.

There are two important differences between finding fixed width peaks for transcription factors and histone modifications. The first is the size of the peaks: Most histone modifications should be analyzed using a peak size in the range of 500-2000 usually (i.e. "-size 1000"). Below is an example of the histone mark distribution around transcription factor peaks (i.e. the things you're hoping Motif Finding will identify), which can be used to help estimate the parameters:



The other is that you should omit the "-center" option (i.e. do not specify "-style factor"). Since you are looking at a region, you do not necessarily want to center the peak on the specific position with the highest tag density, which may be at the edge of the region. Besides, in the case of histone modifications at enhancers, the highest signal will usually be found on nucleosomes surrounding the center of the enhancer, which is where the functional sequences and transcription factor binding sites reside. Consider H3K4me marks surrounding distal PU.1 transcription factor peaks. Typically, adding the -center option moves peaks further away from the functional sequence in these scenarios. An example for finding peaks:

```
findPeaks H3K4me1-ChIP-Seq/ -i Input-ChIP-Seq -size 1000 -o H3K4me1-ChIP-Seq/peaks.1kb.txt
```

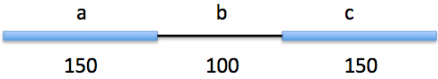
One issue with finding histone modification peaks using the defaults in HOMER is that the Local filtering step removes several of the peaks due to the "spreading" nature of many histone modifications. This can be good and bad. If you are looking for nice concentrated regions of modified histones, the resulting peaks will be a nice set for further analysis such as motif finding. However, if you are looking to identify every region in the cell that has an appreciable amount of modified histone, you may want to disable local filtering, or consider using the "-region" option below e.g.:

```
findPeaks H3K4me1-ChIP-Seq/ -i Input-ChIP-Seq -size 1000 -L 0 -o H3K4me1-ChIP-Seq/peaks.1kb.txt
```

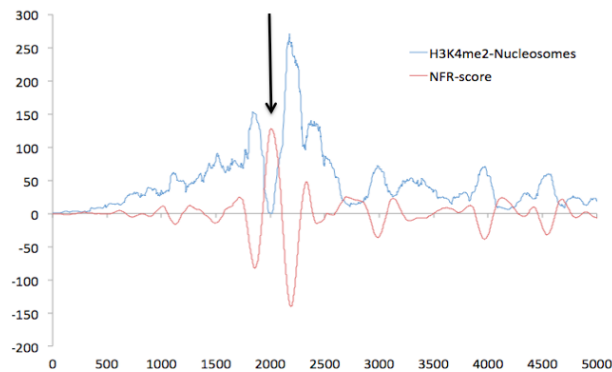
Also, if using MNase-treated chromatin (i.e. nucleosome mapping), you may want to use "-C 0" to avoid filtering peaks composed of highly clonal tag positions. These can arise from well positioned nucleosomes (or sites that are nicely digested by MNase at least).

Nucleosome Free Regions (NFR) -nfr

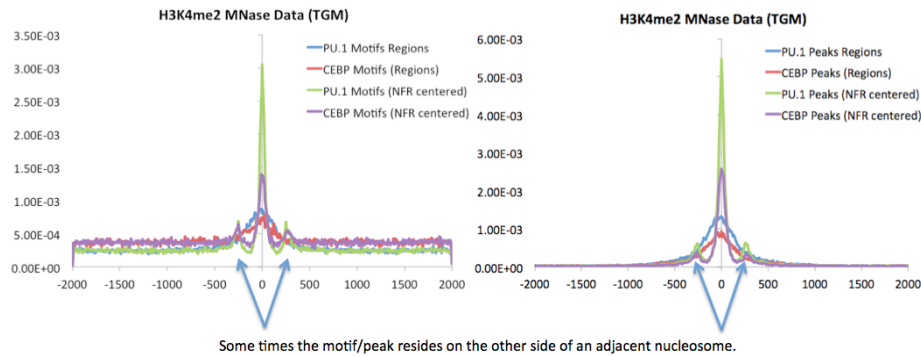
Just like peak centering for transcription factors, **findPeaks** has an option to look for large "dips" in the histone modification signal to infer where the primary nucleosome free region (NFR) is in the region. By adding "-nfr" to the command, HOMER will search for the location within the region that has the greatest differential in ChIP-signal and assign that location as the peak center.



NFR score = (a+c) – b
Where a, b, and c are normalized for their length



Works much better with MNase treated ChIP-Seq samples. The output file will then be centered on the NFR - this is useful for motif finding. By using NFRs instead of whole regions, you can narrow the search for regulatory elements down the +/- 100 bp instead of +/-500 or 1000 bp. Below is an example of ChIP-Seq peak locations with respect to center of H3K4me2-ChIP-Seq regions generated with and without the "-nfr" flag in Macrophages.



Peak finding and Sequencing Saturation

HOMER does not try to estimate sequencing saturation, which is the estimate of whether or not you have sequenced enough tags to identify all the peaks in a given experiment. Generally speaking, if you sequence more, you will get more peaks since your sensitivity will increase. The only real way to assess this is if you can somehow show that all of the "functional" or "real" peaks have high tag counts (i.e. are well above the threshold for identifying a peak), meaning that sequencing more is not likely to identify more "real" peaks. This generally cannot be determined by simply re-sampling the data and repeating the peak finding procedure - you need some sort of outside information to assess peak quality, such as motif enrichment or something else. Simply re-sampling will likely only tell you if you've gotten to the point where you are simply re-sequencing the same fragments again - i.e. becoming clonal.

Analyzing GRO-Seq: *de novo* transcript identification

To find transcripts directly from GRO-Seq, use the `findPeaks` command:

```
findPeaks <tag directory> -style groseq -o auto
```

```
i.e. findPeaks Macrophage-GroSeq -style groseq -o auto
```

GRO-Seq analysis does not make use of an control tag directory

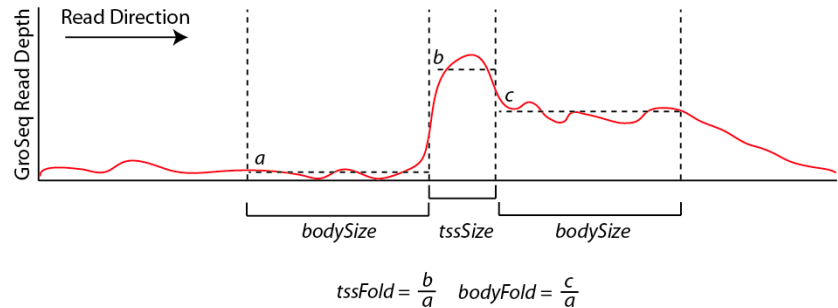
Basic Idea behind GRO-Seq Transcript identification

Finding transcripts using strand-specific GRO-Seq data is not trivial. GRO-Seq measures the production of nascent RNA, and is capable of revealing the loction of protein coding transcripts, promoter anti-sense transcripts, enhancer templated transcripts, long and short functional non-coding and miRNA transcripts, Pol III and Pol I transcripts, and whatever else is being transcribed in the cell nucleus. Identification and quantifiction of these transcripts is important for downstream analysis. Traditional RNA-Seq tools mainly focus on mRNA, which has different features than GRO-Seq, and are generally not useful for identifying GRO-Seq transcripts.

Important NOTE: Just as with ChIP-Seq, not all GRO-Seq data was created equally. Data created by different labs can

have features that make it difficult to have an single analysis technique that works perfectly for each one. As such, there are many parameters to play with the help get the desired results.

A large number of assumptions go into the analysis and are covered in more detail in the [GRO-Seq tutorial \(coming soon\)](#). In a nutshell, findPeaks tracks along each strand of each chromosome, searching for regions of continous GRO-Seq signal. Once it encounters high numbers of GRO-Seq reads, it starts a transcript. If the signal decreases significantly or disappears, the putative transcript is stopped. If the signal increases significantly (and sustainably), then a new transcript is considered from that point on. If the signal spikes, but overall does not increase over a large distance, it is considered an artifact or pause site and not considered in the analysis. Below is a chart that helps explain how the transcript detection works:



By default, new transcripts are created when the *tssFold* exceeds 4 and *bodyFold* exceed 3 ("**-tssFold <#>**", "**-bodyFold <#>**"). A small pseudo-count is added to the tag count from region *a* above to avoid dividing by zero and helps serve to set a minimum threshold for transcript detection ("**-pseudoCount <#>**", default: 1). Most transcripts show robust signal at the start of the transcript, and the *tssFold* helps select for these regions with high accuracy. The *bodyFold* is important for distinguishing between "spikes" in signal and real start sites; if a transcript is real, it's likely that increased levels of transcription follow behind the putative TSS. If the signal is roughly equal before and after the putative TSS, it is more likely to be an artifact.

To increase sensitivity, HOMER tries to adjust the size of the *bodySize* parameter above since it essentially defines the resolution of the detected transcript. If there are a large number of GRO-Seq tags in a region, the *bodySize* can be small since there is adequate data to estimate the location of the transcript. However, if the data is relatively sparse, the *bodySize* needs to be large to get a reliable estimate of the level of the transcript. The minimum and maximum *bodySizes* are 600 and 10000 bp ("**-minBodySize <#>**", "**-maxBodySize <#>**"). HOMER uses the smallest *bodySize* that contains at least *x* number of tags, where *x* is determined as the number of tags where the chance of detecting a *bodyFold* change is less than 0.00001 assuming the read depth varies according to the poisson distribution (adjustable with "**-confPvalue <#>**", or directly with "**-minReadDepth <#>**"). The basic idea is that the threshold for tag counts must be high enough that we don't expect it to vary too much by chance.

Using uniquely mappable regions to improve results

Since some transcripts cover very large regions, there are many places where genomic repeats interrupt the GRO-Seq signal of continous transcripts. To help deal with this problem, HOMER can take advantage of mappability information to help estimate transcript levels where uniquely mapping sequencing reads is not possible. In general this information is not really that helpful for ChIP-Seq analysis, but in this case it can make an important difference. For now, HOMER only take specially formatted binary files available below. To use them, download the appropriate version and unzip the archive:

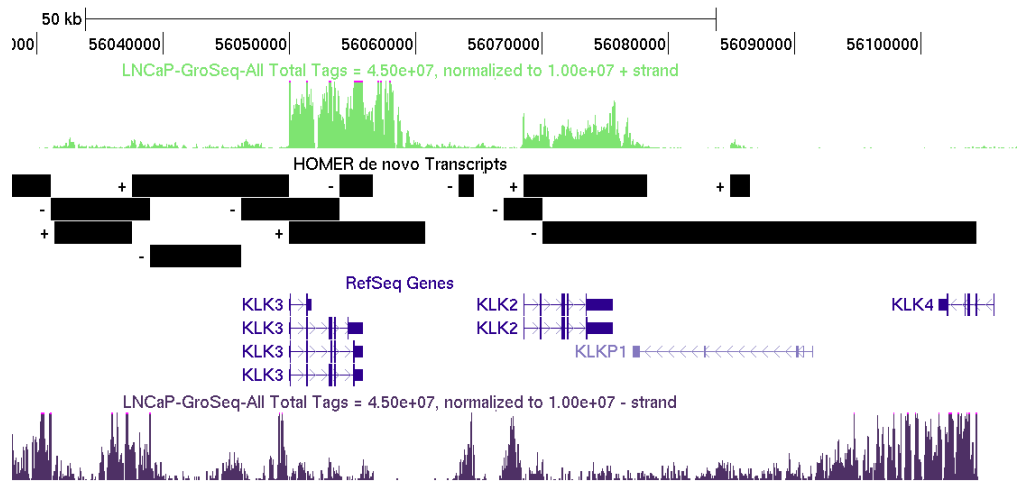
Human: [hg18 hg19](#)
Mouse: [mm8 mm9](#)
Fly: [dm3](#)

To use the uniq-map information, specify the location of the unzipped directory on the command line with "**-uniqmap <directory>**":

```
findPeak Macrophage-GroSeq/ -style groseq -o auto -uniqmap mm9-uniqmap/
```

GRO-Seq analysis output

Running findPeaks in groseq mode will produce a file much like the one produced for traditional peak finding, complete with a header section listing the parameters and statistics from the analysis. HOMER can also produce a GTF (gene transfer format) file for use with various programs. If "**-o auto**" is used to specify output, a "transcripts.gtf" file will be created in the tag directory. Otherwise, you can specify the name of the GTF output file by use "**-gtf <filename>**". The GTF file can also be easily uploaded to the UCSC Genome Browser to visualize your transcripts.



The GRO-Seq transcript detection works pretty well, but is likely to get some face-lifts in the near future.

Command line options for findPeaks

Usage: findPeaks <tag directory> [options]

Finds peaks in the provided tag directory. By default, peak list printed to stdout

General analysis options:

- o <filename|auto> (file name for to output peaks, default: stdout)
"-o auto" will send output to "<tag directory>/peaks.txt", ".../regions.txt",
or ".../transcripts.txt" depending on the "-style" option
- style <option> (Specialized options for specific analysis strategies)
factor (transcription factor ChIP-Seq, uses -center, default)
histone (histone modification ChIP-Seq, region based, uses -region -size 500 -L 0)
groseq (de novo transcript identification from GroSeq data)

chipseq/histone options:

- i <input tag directory> (Experiment to use as IgG/Input/Control)
- size <#> (Peak size, default: auto)
- minDist <#> (minimum distance between peaks, default: peak size x2)
- gsize <#> (Set effective mappable genome size, default: 4e9 [Think double stranded!])
- fragLength <#|auto> (Approximate fragment length, default: auto)
- inputFragLength <#|auto> (Approximate fragment length of input tags, default: auto)
- tbp <#> (Maximum tags per bp to count, 0 = no limit, default: auto)
- inputtbp <#> (Maximum tags per bp to count in input, 0 = no limit, default: auto)
- strand <both|separate> (find peaks using tags on both strands or separate, default:both)
- norm # (Tag count to normalize to, default 10000000)
- center (Centers peaks on maximum tag overlap and calculates focus ratios)
- region (extends start/stop coordinates to cover full region considered "enriched")

Peak Filtering options: (set -F/-L/-C to 0 to skip)

- F <#> (fold enrichment over input tag count, default: 4.0)
- P <#> (poisson p-value threshold relative to input tag count, default: 0.0001)
- L <#> (fold enrichment over local tag count, default: 4.0)
- LP <#> (poisson p-value threshold relative to local tag count, default: 0.0001)
- C <#> (fold enrichment limit of expected unique tag positions, default: 2.0)
- localSize <#> (region to check for local tag enrichment, default: 10000)
- inputSize <#> (Size of region to search for control tags, default: 2x peak size)
- fdr <#> (False discovery rate, default = 0.001)
- poisson <#> (Set poisson p-value cutoff, default: uses fdr)
- tagThreshold <#> (Set # of tags to define a peak, default: uses fdr)

GroSeq Options: (Need to specify "-style groseq"):

- tssSize <#> (size of region for initiation detection/artifact size, default: 300)
- minBodySize <#> (size of regoin for transcript body detection, default: 600)
- maxBodySize <#> (size of regoin for transcript body detection, default: 10000)
- tssFold <#> (fold enrichment for new initiation dectection, default: 4.0)
- bodyFold <#> (fold enrichment for new transcript dectection, default: 3.0)
- endFold <#> (end transcript when levels are this much less than the start, default: 25.0)

-fragLength <#> (Approximate fragment length, default: 150)
-uniqmap <directory> (directory of binary files specifying uniquely mappable locations)
Download from <http://biowhat.ucsd.edu/homer/groseq/>
-confPvalue <#> (confidence p-value: 0.00001)
-minReadDepth <#> (Minimum initial read depth for transcripts, default: auto)
-gtf <filename> (Output de novo transcripts in GTF format)
"-o auto" will produce <dir>/transcripts.txt and <dir>/transcripts.gtf

Links to alternative peak finding software

Lots of quality programs exist for finding peaks in ChIP-Seq data. Most use slightly different assumptions and peak definitions that result in slightly different sets of peaks. Many of these programs output peak files in BED format. To covert these to HOMER peak file format, use the bed2pos.pl program to convert the file (as of now, most HOMER programs work with BED files, so this isn't really necessary):

bed2pos.pl peaks.bed > peaks.txt

Peak finding software links:

- [MACS](#) (Liu Lab)
- [ChIPSeq Peak Finder](#) (Wold Lab)
- [CCAT](#) (Good for histone modifications)
- [FindPeaks](#)
- Probably hundreds of others: Google it!



Can't figure something out? Questions, comments, concerns, or other feedback:
cbenner@ucsd.edu



HOMER

Software for motif discovery and next-gen sequencing analysis

Finding Enriched Motifs in Genomic Regions (*findMotifsGenome.pl*)

HOMER was initially developed to automate the process of finding enriched motifs in ChIP-Seq peaks. More generally, HOMER analyzes genomic positions, not limited to only ChIP-Seq peaks, for enriched motifs. The main idea is that all the user really needs is a file containing genomic coordinates (i.e. a HOMER peak file or BED file), and HOMER will generally take care of the rest. To analyze a peak file for motifs, run the following command:

findMotifsGenome.pl <peak/BED file> <genome> <output directory> [options]

i.e. **findMotifsGenome.pl ERpeaks.txt hg18r ER_MotifOutput/**

A variety of output files will be placed in the <output directory>, including html pages showing the results.

The findMotifsGenome.pl program is a wrapper that helps set up the data for analysis using the HOMER motif discovery algorithm. By default this will perform *de novo* motif discovery as well as check the enrichment of known motifs. If you have not done so already, please look over [this page](#) describing how HOMER analyzes sequences for enriched motifs.

An important prerequisite for analyzing genomic motifs is that the appropriate genome [must be configured for use with HOMER](#). In version v3.1, HOMER now handles custom/arbitrary genomes. Instead of installing/configuring a genome, you can specify the path to a file or directory containing the genomic sequence in FASTA format. The genome can be in a single FASTA file, or you specify a directory where each chromosome can be in a separate file (named chrXXX.fa or chrXXX.fa.masked). In either case, the FASTA headers must contain the chromosome names followed by white space, i.e. ">chr blahblahblah", not ">chr1-blahblahblah", or preferably only ">chr1". (also note that homer will create a "prepared/" directory where the genome is, so make sure you have write permissions in the genomic directory.

Acceptable Input files

findMotifsGenome.pl accepts [HOMER peak files](#) or [BED files](#):

HOMER peak files should have at minimum 5 columns (separated by TABs, additional columns will be ignored):

- Column1: Unique Peak ID
- Column2: chromosome
- Column3: starting position
- Column4: ending position
- Column5: Strand (+/- or 0/1, where 0="+", 1="-")

BED files should have at minimum 6 columns (separated by TABs, additional columns will be

ignored)

- Column1: chromosome
- Column2: starting position
- Column3: ending position
- Column4: Unique Peak ID
- Column5: not used
- Column6: Strand (+/- or 0/1, where 0="+", 1="-")

In theory, HOMER will accept BED files with only 4 columns (+/- in the 4th column), and files without unique IDs, but this is NOT recommended. For one, if you don't have unique IDs for your regions, it's hard to go back and figure out which region contains which peak.

Mac Users: If using a EXCEL to prepare input files, make sure to save files as a "Text (Windows)" if running MacOS - saving as "Tab delimited text" in Mac produces problems for the software. Otherwise, you can run the script "**changeNewLine.pl <filename>**" to convert the Mac-formatted text file to a Windows/Dos/Unix formatted text file.

If errors occur, it is likely that the file is not in the correct format, or the first column is not actually populated with unique identifiers.

Custom Background Regions

Since HOMER uses a differential motif discovery algorithm, different types of background sequences can be chosen to produce different results. For example, you may want to compare the ChIP-Seq peaks specific in one cell type versus the peaks that are specific to another. To do this, create a second peak/BED file and use it with the argument "**-bg <peak/BED file>**". HOMER will still try to normalize the background to remove GC-bias and will also perform autonormalization (see below). You can turn off the normalization with ("**-noweight**" and/or "**-nlen 0**").

How findMotifsGenome.pl works

There are a series of steps that the program goes through to find quality motifs:

1. Verify peak/BED file

HOMER makes sure you have valid peaks, and checks to make sure you have unique peak identifiers. If there are replicates, it will inform you, and will add numbers to peak names to ensure they are unique for downstream analysis.

2. Extract sequences from the genome corresponding to the regions in the input file, filtering sequences that are >70% "N"

This step is pretty self explanatory. If you wish to extract sequences from a genome for any reason, check out [homerTools](#). HOMER will also trash sequences that are predominately "N". If you feel you are throwing away too many sequences, try running **findMotifsGenome.pl** on an unmasked genome.

3. Calculate GC/CpG content of peak sequences.

CpG Islands are the single biggest source of sequence content bias in mammalian genomes, and are unfortunately found near transcription start sites, where all the action is! By default, HOMER tracks GC% (use "**-cpg**" to use CpG%).

4. Prepare the genomic sequences of the selected size to serve as background sequences.

This step is only done the first time you find motifs from regions of a given size ("**-size <#>**"). HOMER takes regions near the TSS of genes (+/- 50kb) and splits them into regions of the indicated size. It then calculates their GC/CpG% and stores them for later use to speed up execution the next time you search for motifs from similar sized regions.

5. Randomly select background regions for motif discovery.

Since HOMER is a differential motif discovery algorithm, it must use background sequence regions as a control. By default, HOMER selects enough random background regions such that the total number of regions is 50000 or 2x the total number of peaks, whichever is larger (to change use "**-N <#>**"). The more total sequence that is used, the slower the program will run, but you want to make sure there is enough background regions to reliably estimate motif frequency. HOMER attempts to select background regions that match the GC-content distribution of the input sequences (in 5% increments). For example, if your input regions are extremely GC-rich, HOMER will select random regions from GC-rich regions of the genome as a control.

If custom background regions are provided ("**-bg <peak/BED file>**"), HOMER will automatically ensure that these regions do NOT overlap with the target regions (using **mergePeaks**). Custom regions will still be normalized for GC-content.

6. Autonormalization of sequence bias.

Autonormalization is a unique procedure provided by HOMER that attempts to remove bias introduced by lower-order oligo sequences. It works by assuming your targets regions and background regions should not have an imbalance in 1-mers, 2-mers, 3-mers, etc. The maximum length of oligo that is autonormalized is specified by "**-nlen <#>**" (default is 3, to disable use "**-nlen 0**"). For example, there should not be significantly more A's in the target sequences than in the background. After calculating the imbalances for each oligo, it adjusts the weights of each background sequence by a small amount to help normalize any imbalance. If target sequences are rich in A, then background sequences that contain many A's will be assigned higher weights while those with very few A's will be assigned lower weights. The weights are incremented by only small amounts and the procedure repeated many times in a hill climbing optimization. This procedure helps remove some of the sequence bias associated with certain genomic regions, or bias that may have been introduced by biased experimental results such as biased sequencing.

7. Check enrichment of known motifs

HOMER screens its library of reliable motifs against the target and background sequences for enrichment, returning motifs enriched with a p-value less than 0.05. The known motif enrichment is performed first since it is usually faster, and gives a faster look at what's enriched in your target regions. Known motif enrichment will be reported to the "knownResults.html" file in the output directory.

8. *de novo* motif finding

Best saved for last. By default, HOMER will search for motifs of len 8, 10, and 12 bp (change using "**-len <#,#,#>**" with no spaces between the numbers, i.e. "**-len 6,10,15,20**"). For a more detail description of the motif discovery algorithm, see [here](http://biowhat.ucsd.edu/homer/ngs/peakMotifs.html).

Output from the de novo motif finding will be displayed in the "homerResults.html" file.

findMotifsGenome.pl Output

A full description of motif finding output and the output can be found [here](#).

Several files are produced in the output directory:

homerMotifs.motifs<#> : these are the output files from the de novo motif finding, separated by motif length, and represent separate runs of the algorithm.

homerMotifs.all.motifs : Simply the concatenated file composed of all the homerMotifs.motifs<#> files.

motifFindingParameters.txt : command used to execute findMotifsGenome.pl

knownResults.txt : text file containing statistics about known motif enrichment (open in EXCEL).

seq.autonorm.tsv : autonormalization statistics for lower-order oligo normalization.

homerResults.html : formatted output of *de novo* motif finding.

Homer de novo Motif Results

[Known Motif Enrichment Results](#)
[Gene Ontology Enrichment Results](#)
If Homer is having trouble matching a motif to a known motif, try copy/pasting the matrix file into [STAMP](#)
More information on motif finding results: [HOMER](#) | [Description of Results](#) | [Tips](#)
Total target sequences = 37301
Total background sequences = 35962
* - possible false positive

Rank	Motif	P-value	log P-value	% of Targets	% of Background	STD(Bg STD)	Best Match/Details	Motif File
1	TGTTTACATA	1e-12661	-2.915e+04	70.91%	15.19%	40.5bp (65.1bp)	Foxa2(Forkhead)/Liver-Foxa2-ChIP-Seq/Homer More Information Similar Motifs Found	motif file (matrix)
2	CTTGGCAG	1e-578	-1.332e+03	27.14%	16.52%	54.0bp (65.5bp)	NF1-halfsite(CTF)/LNCaP-NF1-ChIP-Seq/Homer More Information Similar Motifs Found	motif file (matrix)
3	TTTTATTGGC	1e-384	-8.860e+02	17.77%	10.53%	53.9bp (62.1bp)	Unknown/Homeobox/Limb-p300-ChIP-Seq/Homer More Information Similar Motifs Found	motif file (matrix)
4	CCTGTAAAT	1e-164	-3.783e+02	3.17%	1.28%	52.2bp (62.9bp)	PH0048.1_Hoxa13 More Information Similar Motifs Found	motif file (matrix)
5	ATGACTCA	1e-151	-3.485e+02	3.38%	1.47%	50.2bp (65.4bp)	NF-E2(bZIP)/K562-NFE2-ChIP-Seq/Homer More Information Similar Motifs Found	motif file (matrix)
6	GCCATCTGGTGG	1e-107	-2.485e+02	1.21%	0.35%	56.3bp (69.7bp)	CTCF(Zf)/CD4+-CTCF-ChIP-Seq/Homer More Information Similar Motifs Found	motif file (matrix)
7	AGATAAGATC	1e-72	-1.671e+02	2.10%	1.02%	55.1bp (58.5bp)	MA0029.1_Evi1 More Information Similar Motifs Found	motif file (matrix)

homerResults/ directory: contains files for the homerResults.html webpage, including motif<#>.motif files for use in finding specific instance of each motif.

knownResults.html : formatted output of known motif finding.

knownResults/ directory: contains files for the knownResults.html webpage, including known<#>.motif files for use in finding specific instance of each motif.

Interpreting motif finding results

The format of the output files generated by **findMotifsGenome.pl** are identical to those generated by the promoter-based version **findMotifs.pl** (description).

In general, when analyzing ChIP-Seq / ChIP-Chip peaks you should expect to see strong enrichment for a motif resembling the site recognized by the DNA binding domain of the factor you are studying. Enrichment p-values reported by HOMER should be very very significant (i.e. $< 1e-50$). If this is not the case, there is a strong possibility that the experiment may have failed in one way or another. For example, the peaks could be of low quality because the factor is not expressed very high.

Practical Tips for Motif finding

Important motif finding parameters

Masked vs. Unmasked Genome (i.e. hg18 vs. hg18r)

Actually, this usually doesn't matter *that* much. Since HOMER is a differential motif discovery algorithm, common repeats are usually in both the target and background sequences. However, it is not uncommon that a transcription factor binds to a certain class of repeats, which may cause several large stretches of similar sequence to be processed, biasing the results. Usually it's safer to go with the masked version.

Region Size ("**-size <#>**", default: 200)

The size of the region used for motif finding is important. If analyzing ChIP-Seq peaks from a transcription factor, Chuck would recommend 50 bp for establishing the primary motif bound by a given transcription factor and 200 bp for finding both primary and "co-enriched" motifs for a transcription factor. When looking at histone marked regions, 500-1000 bp is probably a good idea (i.e. H3K4me or H3/H4 acetylated regions). In theory, HOMER can work with very large regions (i.e. 10kb), but with the larger the regions comes more sequence and longer execution time.

Motif length ("**-len <#>**" or "**-len <#>,<#>,...**", default 8,10,12)

Specifies the length of motifs to be found. HOMER will find motifs of each size separately and then combine the results at the end. The length of time it takes to find motifs increases greatly with increasing size. In general, it's best to try out enrichment with shorter lengths (i.e. less than 15) before trying longer lengths. Much longer motifs can be found with HOMER, but it's best to use smaller sets of sequence when trying to find long motifs (i.e. use "**-len 20 -size 50**"), otherwise it may take way too long (or take too much memory). The other trick to reduce the total resource consumption is to reduce the number of background sequences (**-N <#>**).

Mismatches allowed in global optimization phase ("**-mis <#>**", default: 2)

HOMER looks for promising candidates by initially checking ordinary oligos for enrichment, allowing mismatches. The more mismatches you allow, the more sensitive the algorithm, particularly for longer motifs. However, this also slows down the algorithm a bit. If searching for motifs longer than 12-15 bp, it's best to increase this value to at least 3 or even 4.

Number of CPUs to use ("**-p <#>**", default 1)

HOMER is now multicore compliant. It's not perfectly parallelized, however, certain types of analysis can benefit. In general, the longer the length of the motif, the better the speed-up you'll see.

- Number of motifs to find ("**-S <#>**", default 25)
Specifies the number of motifs of each length to find. 25 is already quite a bit. If anything, I'd recommend reducing this number, particularly for long motifs to reduce the total execution time.
- Normalize CpG% content instead of GC% content ("**-cpg**")
Consider trying if HOMER is stuck finding "CGCGCGCG"-like motifs. You can also play around with disabling GC/CpG normalization ("**-noweight**").
- Region level autonormalization ("**-nlen <#>**", default 3, "-nlen 0" to disable)
Motif level autonormalization ("**-olen <#>**", default 0 i.e. disabled)
Autonormalization attempts to remove sequence bias from lower order oligos (1-mers, 2-mers ... up to <#>). Region level autonormalization, which is for 1/2/3 mers by default, attempts to normalize background regions by adjusting their weights. If this isn't getting the job done (autonormalization is not guaranteed to remove all sequence bias), you can try the more aggressive motif level autonormalization ("**-olen <#>**"). This performs the autonormalization routine on the oligo table during de novo motif discovery. (see [here](#) for more info)
- User defined background regions ("**-bg <peak file of background regions>**")
Why let HOMER randomly pick you background regions when you can choose them yourself!! These will still be normalized for CpG% or GC% content just like randomly chosen sequences and autonormalized unless these options are turned off (i.e. "-nlen 0 -noweight"). This can be very useful since HOMER is a differential motif discovery algorithm. For example, you can give HOMER a set of peaks co-bound by another factor and compare them to the rest of the peaks. HOMER will automatically check if the background peaks overlap with the target peaks using **mergePeaks**, and discard overlapping regions.
- Hypergeometric enrichment scoring ("**-h**")
By default, **findMotifsGenome.pl** uses the binomial distribution to score motifs. This works well when the number of background sequences greatly outnumber the target sequences - however, if you are using "**-bg**" option above, and the number of background sequences is smaller than target sequences, it is a good idea to use the hypergeometric distribution instead ("**-h**"). FYI - The binomial is faster to compute, hence it's use for motif finding in large numbers of regions.
- Find enrichment of individual oligos ("**-oligo**").
This creates output files in the output directory named *oligo.length.txt*.
- Force **findMotifsGenome.pl** to re-prepare genome for the given region size ("**-preparse**").
In case there is a problem with the existing preparsed files, force them to be remade with "-preparse".
- Only search for motifs on + strand ("**-norevopp**")
By default, HOMER looks for transcription factor-like motifs on both strands. This will force it to only look at the + strand (relative to the peak, so - strand if the peak is on the - strand).
- Search for RNA motifs ("**-rna**")
If looking at RNA data (i.e. Clip-Seq or similar), this option will restrict HOMER to only search the + strand (relative to the peak), and will output RNA motif logos (i.e. U instead of T). It will also try to compare found motifs to an RNA motif database, which sadly, only contains miRNAs right now... I guess chuck roundhouse kicked all of the splicing and other RNA motifs into hard to find databases.

Mask motifs ("**-mask <motif file>**")

Mask the motif(s) in the supplied motif file before starting motif finding. Multiple motifs can be in the motif file.

Optimize motifs ("**-opt <motif file>**")

Instead of looking for novel de novo motifs, HOMER will instead try to optimize the motif supplied. This is cool when trying to change the length of a motif, or find a very long version of a given motif. For example, if you specify "-opt <file>" and "-len 50", it will try to expand the motif to 50bp and optimize it.

Dump FASTA files ("**-dumpFasta**")

Like the fact that HOMER organizes and extracts your sequence files, but don't care for HOMER as a motif finding algorithm? That's cool, just specify "-dumpFasta" and the files "target.fa" and "background.fa" will show up in your output directory. You can then use them with MEME or whatever. Just remember, Chuck knows where you live...

Finding Instance of Specific Motifs

By default, HOMER does not return the locations of each motif found in the motif discovery process. To recover the motif locations, you must first select the motifs you're interested in by getting the "motif file" output by HOMER. You can combine multiple motifs in single file if you like to form a "motif library". To identify motif locations, you have two options:

1. Run **findMotifsGenome.pl** with the "**-find <motif file>**" option. This will output a tab-delimited text file with each line containing an instance of the motif in the target peaks. The output is sent to *stdout*.

For example: **findMotifsGenome.pl ERalpha.peaks hg18 MotifOutputDirectory/ -find motif1.motif > outputfile.txt**

The output file will contain the columns:

1. Peak/Region ID
2. Offset from the center of the region
3. Sequence of the site
4. Name of the Motif
5. Strand
6. Motif Score (log odds score of the motif matrix, higher scores are better matches)

2. Run **annotatePeaks.pl** with the "**-m <motif file>**" option (see the [annotation section](#) for more info). Chuck prefers doing it this way. This will output a tab-delimited text file with each line containing a peak/region and a column containing instance of each motif separated by commas to stdout

For example: **annotatePeaks.pl ERalpha.peaks hg18 -m motif1.motif > outputfile.txt**

The output file will contain columns:

1. Peak/Region ID
2. Chromosome
3. Start

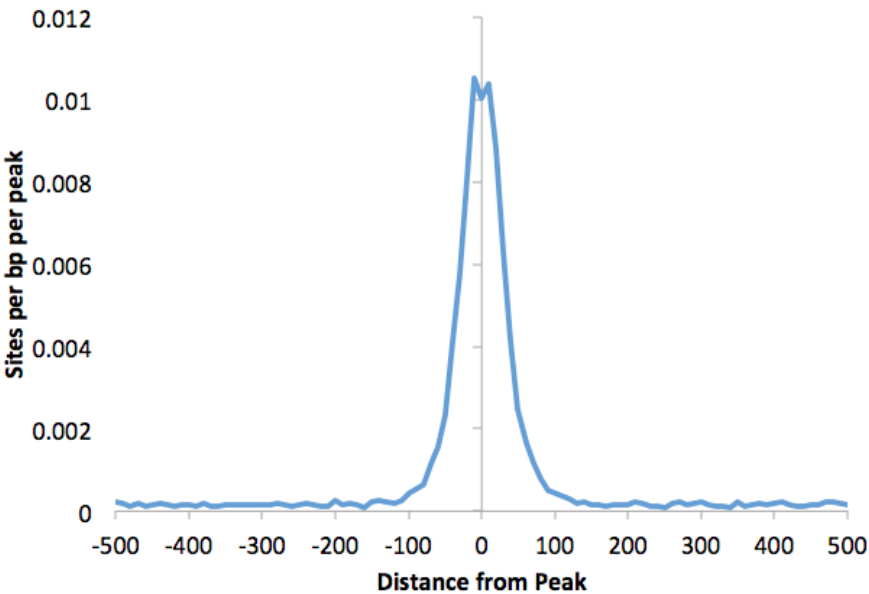
- 4. End
- 5. Strand of Peaks
- 6-18: annotation information
- 19. CpG%
- 20. GC%
- 21. Motif Instances
- ...

Motif Instances have the following format:
<distance from center of region>(<sequence>,<strand>,<conservation>)
i.e -29(TAAATCAACA,+,0.00)

You can also find histogram of motif density this way by adding "-hist <#>" to the command. For example:

```
annotatePeaks.pl ERalpha.peaks hg18 -m ere.motif foxa1.motif -size 1000 -hist 10 > outputfile.txt
```

Graphing the output with EXCEL:



Command-line options for findMotifsGenome.pl

Program will find de novo and known motifs in regions in the genome

Usage: findMotifsGenome.pl <pos file> <genome> <output directory> [additional options]
Example: findMotifsGenome.pl peaks.txt mm8r peakAnalysis -size 200 -len 8

Possible Genomes:

- ...
- Custom: provide the path to genome FASTA files (directory or single file)
- Heads up: will create the directory "preparsed/" in same location.

Basic options:

- bg <background position file> (genomic positions to be used as background, default=automatic)
 - removes background positions overlapping with target positions
- chopify (chop up large background regions to the avg size of target regions)
- len <#>[,<#>,<#>...] (motif length, default=8,10,12) [NOTE: values greater 12 may cause the program to run out of memory - in these cases decrease the number of sequences analyzed (-N), or try analyzing shorter sequence regions (i.e. -size 100)]
- size <#> (fragment size to use for motif finding, default=200)
 - size <#,#> (i.e. -size -100,50 will get sequences from -100 to +50 relative from center)
 - size given (uses the exact regions you give it)
- S <#> (Number of motifs to optimize, default: 25)
- mis <#> (global optimization: searches for strings with # mismatches, default: 2)
- norevopp (don't search reverse strand for motifs)
- nomotif (don't search for de novo motif enrichment)
- rna (output RNA motif logos and compare to RNA motif database, automatically sets -norevopp)

Scanning sequence for motifs

- find <motif file> (This will cause the program to only scan for motifs)

Known Motif Options/Visualization

- bits (scale sequence logos by information content, default: doesn't scale)
- nocheck (don't search for de novo vs. known motif similarity)
- mcheck <motif file> (known motifs to check against de novo motifs, default: /bioinformatics/homer/data/knownTFs/all.motifs)
- float (allow adjustment of the degeneracy threshold for known motifs to improve p-value[dangerous])
- noknown (don't search for known motif enrichment, default: -known)
- mknown <motif file> (known motifs to check for enrichment, default: /bioinformatics/homer/data/knownTFs/known.motifs)

Sequence normalization options:

- gc (use GC% for sequence content normalization, now the default)
- cpg (use CpG% instead of GC% for sequence content normalization)
- noweight (no CG correction)

Advanced options:

- h (use hypergeometric for p-values, binomial is default)
- N <#> (Number of sequences to use for motif finding, default=max(50k, 2x input))
- noforce (will attempt to reuse sequence files etc. that are already in output directory)
- local <#> (use local background, # of equal size regions around peaks to use i.e. 2)
- redundant <#> (Remove redundant sequences matching greater than # percent, i.e. -redundant 0.5)
- mask <motif file1> [motif file 2]... (motifs to mask before motif finding)
- opt <motif file1> [motif file 2]... (motifs to optimize or change length of)
- refine <motif file1> (motif to optimize)
- rand (randomize target and background sequences labels)
- ref <peak file> (use file for target and background - first argument is list of peak ids for targets)
- oligo (perform analysis of individual oligo enrichment)
- dumpFasta (Dump fasta files for target and background sequences for use with other programs)
- preparse (force new background files to be created)
- keepFiles (keep temporary files)

homer2 specific options:

- homer2 (use homer2 instead of original homer, default)
- nlen <#> (length of lower-order oligos to normalize in background, default: -nlen 3)
 - nmax <#> (Max normalization iterations, default: 160)
- olen <#> (lower-order oligo normalization for oligo table, use if -nlen isn't working well)
- p <#> (Number of processors to use, default: 1)
- e <#> (Maximum expected motif instance per bp in random sequence, default: 0.01)
- cache <#> (size in MB for statistics cache, default: 500)
- quickMask (skip full masking after finding motifs, similar to original homer)

Original homer specific options:

- homer1 (to force the use of the original homer)
- depth [low|med|high|allnight] (time spent on local optimization default: med)



Can't figure something out? Questions, comments, concerns, or other feedback:
cbenner@ucsd.edu

Software for motif discovery and next-gen sequencing analysis

Horner contains a useful, all-in-one program for performing peak annotation called **annotatePeaks.pl**. In addition to associating peaks with nearby genes, **annotatePeaks.pl** can perform Gene Ontology Analysis, genomic feature association analysis (Genome Ontology), associate peaks with gene expression data, calculate ChIP-Seq Tag densities from different experiments, and find motif occurrences in peaks. **annotatePeaks.pl** can also be used to create histograms and heatmaps. Description of the annotation functions are covered below, while quantification of tags, motifs, histograms, etc. are covered [here](#).

Basic usage:

```
i.e. annotatePeaks.pl ERpeaks.txt hg18 > outputfile.txt
```

The first two arguments, the <peak file> and <genome>, are required, and must be the first two arguments. Other optional command line arguments can be placed in any order after the first two. By default, **annotatePeaks.pl** prints the program output to *stdout*, which can be captured in a file by appending " > filename" to the command. With most uses of **annotatePeaks.pl**, the output is a data table that is meant to be opened with EXCEL or similar program. An example of the output can be seen below:

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R
1	PeakID	Chr	Start	End	Strand	Peak Score	Focus R2	Annotation	Detailed Anno	Distance to T	Nearest PromoterID	Nearest Uni	Nearest Refs	Nearest En	Gene Name	Gene Alias	Gene Description	
2	chr18-1	chr18	69007968	69008268	+	593	0.939	Intron (NR_03_03)	Intron (NR_03_03)	74595	NR_034133	400565	HS.579378	NR_034133	LOC400655		hypothetical	
3	chr9-1	chr9	88209966	88210266	+	5319	0.946	Intergenic	Intergenic	-50894	NM_011185	79670	HS.579057	NM_011185	ENSG000000200C46	DKFZp66661	skin finger, C	
4	chr14-1	chr14	62337073	62337373	+	505.4	0.918	Intron (NM_17_03)	Intron (NM_17_03)	244485	NM_172375	27133	HS.27043	NM_139318	ENSG0000011KNH5	EAG2/H-EAG	potassium voltage-gated channel subunit	
5	chr17-1	chr17	5076243	5076543	+	492.1	0.936	Intron (NR_03_03)	Intron (NR_03_03)	2414	NM_207103	388325	HS.462080	NM_207103	ENSG0000011C17orf87	CA32580	(M) chromosome	
6	chr17-2	chr17	47851214	47852014	+	476.2	0.824	Intergenic	Intergenic	-259488	NM_010182	56934	HS.463466	NM_010182	ENSG0000011CA10	FCR-1	(X) chromosome	
7	chr10-1	chr10	98420680	98420980	+	479.4	0.967	Intron (NM_15_03)	Intron (NM_15_03)	84939	NM_152309	118788	HS.310456	NM_152309	ENSG0000011PI3AP1	BCAP (RP11-1)	phosphoinositide 3-kinase	
8	chr9-2	chr9	81294389	81294689	+	456.9	0.957	Intergenic	Intergenic	492159	NM_007005	7091	HS.444213	NM_007005	ENSG0000011TK44	BCCE-1 (BCCE-1)	transducin-like	
9	chr14-2	chr14	36817736	36818036	+	452.7	0.757	Intron (NM_13_03)	Intron (NM_13_03)	81017	NM_010195	145282	HS.660396	NM_010195	ENSG0000011MISPL1	DKFZp313M7	myosin-III	
10	chr18-2	chr18	20049825	20050125	+	449.7	0.853	Intron (NM_06_03)	Intron (NM_06_03)	56219	NM_018030	114876	HS.370725	NM_018030	ENSG0000011OSBP1A	FLJ10217 (F)	oxysterol-binding protein	
11	chr7-1	chr7	12226829	12227129	+	445.7	0.901	Intron (NM_01_03)	Intron (NM_01_03)	9606	NM_010134	54664	HS.396358	NM_010134	ENSG0000011TME0168	FLJ11273 (M)	transmembrane protein	
12	chr14-3	chr14	88712188	88712488	+	443.1	0.844	Intron (NM_0C_03)	Intron (NM_0C_03)	240869	NM_005197	1112	HS.621371	NM_010185	ENSG0000001FOXN3	C14orf116 (C)	forkhead box protein	
13	chr18-3	chr18	62951924	62952224	+	443.1	0.947	Intergenic	Intergenic	-382689	NR_033921	643542	HS.652901	NR_033921	LOC643542		hypothetical	
14	chr13-1	chr13	32196769	32197069	+	443.1	0.87	Intergenic	Intergenic	-58256	NM_178868	152189	HS.154986	NM_178868	ENSG0000011CMT48	CKLF5F8/CKLF	like MAFK	
15	chr11-1	chr11	110685448	110685748	+	425.8	0.907	Intergenic	Intergenic	-9849	NR_034154	399948	HS.729225	NR_034154	C11orf92	DKFZp781P1	chromosome	
16	chr4-1	chr4	81755366	81755666	+	423.2	0.908	Intron (NM_15_03)	Intron (NM_15_03)	279618	NM_152770	255119	HS.527104	NM_152770	ENSG0000011C4orf22	MGC35043	chromosome	

annotatePeaks.pl accepts HOMER peak files or BED files:

HOMER peak files should have at minimum 5 columns (separated by TABs, additional columns will be ignored):

- Column1: Unique Peak ID
- Column2: chromosome
- Column3: starting position
- Column4: ending position
- Column5: Strand (+/- or 0/1, where 0="+", 1="-")

BED files should have at minimum 6 columns (separated by TABs, additional columns will be ignored)

- Column1: chromosome
- Column2: starting position
- Column3: ending position
- Column4: Unique Peak ID
- Column5: not used

- Column6: Strand (+/- or 0/1, where 0="+", 1="-")

In theory, HOMER will accept BED files with only 4 columns (+/- in the 4th column), and files without unique IDs, but this is NOT recommended.

Mac Users: If using a EXCEL to prepare input files, make sure to save files as a "Text (Windows)" if running MacOS - saving as "Tab delimited text" in Mac produces problems for the software. Otherwise, you can run the script "**changeNewLine.pl** <filename>" to convert the Mac-formatted text file to a Windows/Dos/Unix formatted text file.

If errors occur, it is likely that the file is not in the correct format, or the first column is not actually populated with unique identifiers.

How Basic Annotation Works

The process of annotating peaks/regions is divided into two primary parts. The first determines the distance to the nearest TSS and assigns the peak to that gene. The second determines the genomic annotation of the region occupied by the center of the peak/region.

Distance to the nearest TSS

By default, `annotatePeaks.pl` loads a file in the `"/path-to-homer/data/genomes/<genome>/<genome>.tss"` that contains the positions of RefSeq transcription start sites. It uses these positions to determine the closest TSS, reporting the distance (negative values mean upstream of the TSS, positive values mean downstream), and various annotation information linked to locus including alternative identifiers (unigene, entrez gene, ensembl, gene symbol etc.). This information is also used to link gene-specific information (see below) to a peak/region, such as gene expression.

Genomic Annotation

To annotate the location of a given peak in terms of important genomic features, `annotatePeaks.pl` calls a separate program (`assignGenomeAnnotation`) to efficiently assign peaks to one of millions of possible annotations genome wide. Two types of output are provided. The first is "Basic Annotation" that includes whether a peak is in the TSS (transcription start site), TTS (transcription termination site), Exon (Coding), 5' UTR Exon, 3' UTR Exon, Intronic, or Intergenic, which are common annotations that many researchers are interested in. A second round of "Detailed Annotation" also includes more detailed annotation, also considering repeat elements and CpG islands. Since some annotation overlap, a priority is assign based on the following (in case of ties it's random [i.e. if there are two overlapping repeat element annotations]):

1. TSS (by default defined from -1kb to +100bp)
2. TTS (by default defined from -100 bp to +1kb)
3. CDS Exons
4. 5' UTR Exons
5. 3' UTR Exons
6. **CpG Islands
7. **Repeats
8. Introns
9. Intergenic

** Only applicable for the "Detailed Annotation".

Although HOMER doesn't allow you to explicitly change the definition of the region that is the TSS (-1kb to +100bp), you can "do it yourself" by sorting the annotation output in EXCEL by the "Distance to nearest TSS" column, and selecting those within the range you are interested in.

Using Custom Annotations

Custom Gene Annotation Definition using GTF files

RefSeq doesn't do it for everyone. There are many other quality gene annotations out there, including UCSC genes, Ensembl, and Gencode to name a couple. Even more important, as RNA-Seq methods develop, the locations of exons etc. can be defined based on your own experimental RNA data rather than using a static database to define transcripts. You can download GTF files for various gene definitions from the UCSC Genome Browser in the "[Table Browser](#)" section. Custom GTF files can be created from RNA-Seq data using

tools like [Cufflinks](#).

HOMER can process GTF (Gene Transfer Format) files and use them for annotation purposes ("**-gtf <gtf filename>**"). If a GTF file is specified, HOMER will parse it and use the TSS from the GTF file for determining the distance to the nearest TSS. It will also use the GTF file's definition of TSS/TTS/exons/Introns for Basic Genome Annotation. The original HOMER annotation files are still used for the "Detailed Annotation" since the repeats will not be define in the GTF files.

i.e. **annotatePeaks.pl ERpeaks.txt hg18 -gtf gencode.gtf > outputfile.txt**

HOMER will try it's best to take the "transcript_id" from the GTF definition and translate it into a known gene identifier. If it can't match it to a known gene, many of the annotation columns corresponding to Unigene etc. will be empty.

HOMER can also work with GFF and GFF3 files - to a degree. The problem with these is that the formats do not have the same strict naming convention enforced in GTF files. To use them, substitute "**-gtf <GTF file>**" with "**-gff <GFF file>**" or "**-gff3 <GFF3 file>**"

Custom Promoter Locations

By default, **annotatePeaks.pl** assigns peaks to the nearest TSS. If you have a custom peak/pos file of these locations, you can supply that with the option "**-cTSS <peak/pos file>**". This will override the default, which is RefSeq TSS.

Using Custom Genomes/Annotations

For organisms with relatively incomplete genomes, **annotatePeaks.pl** can still provide some functionality. If the genome is not available as a pre-configured genome in HOMER, then you can supply the path to the full genome FASTA file or path to directory containing chromosome FASTA files as the 2nd argument. For example, lets say you were sequencing the [banana slug](#) genome, and had downloaded the current draft of the genome into the file "bananaSlug.fa". You could then run **annotatePeaks.pl** like this:

annotatePeaks.pl chip-seq-peaks.txt bananaSlug.fa > output.txt

If no genome sequence is available, you can also specify "**none**":

annotatePeaks.pl chip-seq-peaks.txt none > output.txt

You may also find a custom annotation file for the organism, such as **banana_slug_genes.gtf**, or **banana_slug_genes.gff** from the community website. This can then be used to help annotate your data:

annotatePeaks.pl chip-seq-peaks.txt bananaSlug.fa -gtf banana_slug_genes.gtf > output.txt
or
annotatePeaks.pl chip-seq-peaks.txt bananaSlug.fa -gff banana_slug_genes.gff > output.txt

In the case of custom genomes, HOMER will not be able to convert gene IDs to names - you may have to do this yourself, but many of the other features, including motif finding etc., are available as long as you have the sequence.

Basic Annotation File Output

Description of Columns:

1. Peak ID
2. Chromosome
3. Peak start position
4. Peak end position
5. Strand
6. Peak Score
7. FDR/Peak Focus Ratio/Region Size
8. Annotation (i.e. Exon, Intron, ...)
9. Detailed Annotation (Exon, Intron etc. + CpG Islands, repeats, etc.)

10. Distance to nearest RefSeq TSS
11. Nearest TSS: Native ID of annotation file
12. Nearest TSS: Entrez Gene ID
13. Nearest TSS: Unigene ID
14. Nearest TSS: RefSeq ID
15. Nearest TSS: Ensembl ID
16. Nearest TSS: Gene Symbol
17. Nearest TSS: Gene Aliases
18. Nearest TSS: Gene description
19. *Additional columns depend on options selected when running the program.*

As of now, basic annotation is based on alignments of RefSeq transcripts to the UCSC hosted genomes.

Adding Gene Expression Data

annotatePeaks.pl can add gene-specific information to peaks based on each peak's nearest annotated TSS. To add gene expression or other data types, first create a gene data file (tab delimited text file) where the first column contains gene identifiers, and the first row is a header describing the contents of each column. In principle, the contents of these columns doesn't matter. To add this information to the annotation result, use the "**-gene <gene data file>**".

```
annotation.pl <peak file> <genome> -gene <gene data file> > output.txt
```

For peaks that are near genes with associated data in the "gene data file", this data will be appended to the end of the row for each peak.

Peak Annotation Enrichment

Gene Ontology Analysis of Associated Genes

annotatePeaks.pl offers two types of annotation enrichment analysis. The first is based on Gene Ontology classifications for genes. The idea is that your regions (i.e. ChIP-Seq peaks) might be preferentially found near genes with specific biological functions. By specifying "**-go <GO output directory>**", HOMER will take the list of genes associated with your regions and search for enriched functional categories, placing the output in the indicated directory. This is just like looking for GO enrichment in a set of regulated genes ([covered in greater detail here](#)).

There are some caveats to this type of analysis - for example, some genes are simply more likely to be considered in the analysis if there are isolated along the genome where they are likely to be the "closest gene" for peaks in the region. But... since gene density roughly correlates with transcription factor density, it works itself out to some degree. Other tools, such as [GREAT](#) attempt to deal with this problem to some degree, and are worth trying. My experience is that the results are not that different once you factor in the fact that most other tools use "GO slims".

Genome Ontology: Looking for Enriched Genomic Annotations

The **Genome Ontology** looks for enrichment of various genomic annotations in your list of peaks/regions. Just as the Gene Ontology contains groups of genes sharing a biological function, the Genome Ontology contains groups of genomic regions sharing an annotation, such as TSS, or LINE repeats, coding exons, Transcription factor peaks, etc. To run the Genome Ontology, add the option "**-genomeOntology <output directory>**". A much more detailed description of the Genome Ontology will be available [here](#) (coming soon).

Command line options for annotatePeaks.pl

Usage: `annotatePeaks.pl <peak file | tss> <genome version> [additional options...]`

Available Genomes (required argument): (name,org,directory,default promoter set)

-- or --

Custom: provide the path to genome FASTA files (directory or single file)

User defined annotation files (default is UCSC refGene annotation):

annotatePeaks.pl accepts GTF (gene transfer formatted) files to annotate positions relative to custom annotations, such as those from de novo transcript discovery or Gencode.
-gtf <gtf format file> (-gff and -gff3 can work for those files, but GTF is better)

Peak vs. tss/tts/rna mode (works with custom GTF file):

If the first argument is "tss" (i.e. annotatePeaks.pl tss hg18 ...) then a TSS centric analysis will be carried out. Tag counts and motifs will be found relative to the TSS. (no position file needed) ["tts" now works too - e.g. 3' end of gene]
["rna" specifies gene bodies, will automaticall set "-size given"]
NOTE: The default TSS peak size is 4000 bp, i.e. +/- 2kb (change with -size option)
-list <gene id list> (subset of genes to perform analysis [unigene, gene id, accession, probe, etc.], default = all promoters)
-cTSS <promoter position file i.e. peak file> (should be centered on TSS)

Primary Annotation Options:

- m <motif file 1> [motif file 2] ... (list of motifs to find in peaks)
- mscore (reports the highest log-odds score within the peak)
- nmotifs (reports the number of motifs per peak)
- mdist (reports distance to closest motif)
- mfasta <filename> (reports sites in a fasta file - for building new motifs)
- fm <motif file 1> [motif file 2] (list of motifs to filter from above)
- rmrevopp <#> (only count sites found within <#> on both strands once, i.e. palindromic)
- matrix <prefix> (outputs a motif co-occurrence files:
 - prefix.count.matrix.txt - number of peaks with motif co-occurrence
 - prefix.ratio.matrix.txt - ratio of observed vs. expected co-occurrence
 - prefix.logPvalue.matrix.txt - co-occurrence enrichment
 - prefix.stats.txt - table of pair-wise motif co-occurrence statisticsadditional options:
 - matrixMinDist <#> (minimum distance between motif pairs - to avoid overlap)
 - matrixMaxDist <#> (maximum distance between motif pairs)
- mbed <filename> (Output motif positions to a BED file to load at UCSC (or -mpeak))
- d <tag directory 1> [tag directory 2] ... (list of experiment directories to show tag counts for) NOTE: -dfile <file> where file is a list of directories in first column
- bedGraph <bedGraph file 1> [bedGraph file 2] ... (read coverage counts from bedGraph files)
- p <peak file> [peak file 2] ... (to find nearest peaks)
 - pdist to report only distance (-pdist2 gives directional distance)
 - pcount to report number of peaks within region
- vcf <VCF file> (annotate peaks with genetic variation information, one col per individual)
 - editDistance (Computes the # bp changes relative to reference)
 - individuals <name1> [name2] ... (restrict analysis to these individuals)
- gene <data file> ... (Adds additional data to result based on the closest gene. This is useful for adding gene expression data. The file must have a header, and the first column must be a GeneID, Accession number, etc. If the peak cannot be mapped to data in the file then the entry will be left empty.
- go <output directory> (perform GO analysis using genes near peaks)
- genomeOntology <output directory> (perform genomeOntology analysis on peaks)
 - gsize <#> (Genome size for genomeOntology analysis, default: 2e9)

Annotation vs. Histogram mode:

- hist <bin size in bp> (i.e 1, 2, 5, 10, 20, 50, 100 etc.)

The -hist option can be used to generate histograms of position dependent features relative to the center of peaks. This is primarily meant to be used with -d and -m options to map distribution of motifs and ChIP-Seq tags. For ChIP-Seq peaks for a Transcription factor you might want to use the -center option (below) to center peaks on the known motif
** If using "-size given", histogram will be scaled to each region (i.e. 0-100%), with the -hist parameter being the number of bins to divide each region into.

Histogram Mode specific Options:

- nuc (calculated mononucleotide frequencies at each position,
 - Will report by default if extracting sequence for other purposes like motifs)
- di (calculated dinucleotide frequencies at each position)
- histNorm <#> (normalize the total tag count for each region to 1, where <#> is the minimum tag total per region - use to avoid tag spikes from low coverage)
- ghist (outputs profiles for each gene, for peak shape clustering)

-rm <#> (remove occurrences of same motif that occur within # bp)

Peak Centering: (other options are ignored)

- center <motif file> (This will re-center peaks on the specified motif, or remove peak if there is no motif in the peak. ONLY recentering will be performed, and all other options will be ignored. This will output a new peak file that can then be reanalyzed to reveal fine-grain structure in peaks (It is advised to use -size < 200) with this to keep peaks from moving too far (-mirror flips the position)
- multi (returns genomic positions of all sites instead of just the closest to center)

Advanced Options:

- len <#> / -fragLength <#> (Fragment length, default=auto, might want to set to 0 for RNA)
- size <#> (Peak size[from center of peak], default=inferred from peak file)
 - size #,# (i.e. -size -10,50 count tags from -10 bp to +50 bp from center)
 - size "given" (count tags etc. using the actual regions - for variable length regions)
- log (output tag counts as log2(x+1+rand) values - for scatter plots)
- sqrt (output tag counts as sqrt(x+rand) values - for scatter plots)
- strand <+|-|both> (Count tags on specific strands relative to peak, default: both)
- pc <#> (maximum number of tags to count per bp, default=0 [no maximum])
- cons (Retrieve conservation information for peaks/sites)
- CpG (Calculate CpG/GC content)
- ratio (process tag values as ratios - i.e. chip-seq, or mCpG/CpG)
- nfr (report nucleosome free region scores instead of tag counts, also -nfrSize <#>)
- norevopp (do not search for motifs on the opposite strand [works with -center too])
- noadj (do not adjust the tag counts based on total tags sequenced)
- norm <#> (normalize tags to this tag count, default=1e7, 0=average tag count in all directories)
- pdist (only report distance to nearest peak using -p, not peak name)
- map <mapping file> (mapping between peak IDs and promoter IDs, overrides closest assignment)
- noann, -nogene (skip genome annotation step, skip TSS annotation)
- homer1/-homer2 (by default, the new version of homer [-homer2] is used for finding motifs)



Can't figure something out? Questions, comments, concerns, or other feedback:
cbenner@ucsd.edu



HOMER

Software for motif discovery and next-gen sequencing analysis

Quantifying Data and Motifs and Comparing Peaks/Regions in the Genome

Homer contains a useful, all-in-one program for performing peak annotation called **annotatePeaks.pl**. In addition to associating peaks with nearby genes, **annotatePeaks.pl** can perform Gene Ontology Analysis, genomic feature association analysis (Genome Ontology), associate peaks with gene expression data, calculate ChIP-Seq Tag densities from different experiments, and find motif occurrences in peaks. **annotatePeaks.pl** can also be used to create histograms and heatmaps. Description of the annotation functions are covered [here](#), while quantification of tags, motifs, histograms, etc. are covered below.

Basic usage (see [Annotation](#)):

```
annotatePeaks.pl <peak/BED file> <genome> [options] > <output file>
```

i.e. `annotatePeaks.pl ERpeaks.txt hg18 > outputfile.txt`

Everything packed into one program

The great thing about `annotatePeaks.pl` is that it combines many features into a single location. It is the primary program to investigate how sequencing reads, sequence motifs, and other information and annotations interact.

Three primary options are available to specify types of data that can be processed by `annotatePeaks.pl`:

```
-d <tag directory1> [tag directory 2] ...
-m <motif file 1> [motif file 2] ...
-p <peak/BED file 1> [peak/BED file 2] ...
```

By default, these data types are processed relative to each peak/region provided in the primary input file. There are a bunch of options that help fine tune how each type of data is considered by the program covered below.

However, `annotatePeaks.pl` can take the same input data and do other things, such as make histograms and heatmaps, allowing you to explore the data in a different way.

```
-hist <# bin size>
```

Acceptable Input files

annotatePeaks.pl accepts [HOMER peak files](#) or [BED files](#):

HOMER peak files should have at minimum 5 columns (separated by TABs, additional columns will be ignored):

- Column1: Unique Peak ID
- Column2: chromosome
- Column3: starting position
- Column4: ending position
- Column5: Strand (+/- or 0/1, where 0="+", 1="-")

BED files should have at minimum 6 columns (separated by TABs, additional columns will be ignored)

- Column1: chromosome
- Column2: starting position
- Column3: ending position
- Column4: Unique Peak ID
- Column5: not used
- Column6: Strand (+/- or 0/1, where 0="+", 1="-")

In theory, HOMER will accept BED files with only 4 columns (+/- in the 4th column), and files without unique IDs, but this is NOT recommended. For one, if you don't have unique IDs for your regions, it's hard to go back and figure out which region contains which peak.

Mac Users: If using a EXCEL to prepare input files, make sure to save files as a "Text (Windows)" if running MacOS - saving as "Tab delimited text" in Mac produces problems for the software. Otherwise, you can run the script "**changeNewLine.pl** **<filename>**" to convert the Mac-formatted text file to a Windows/Dos/Unix formatted text file.

If errors occur, it is likely that the file is not in the correct format, or the first column is not actually populated with unique identifiers.

TSS Mode

Instead of supplying a peak file, HOMER makes it easy to do analysis of features near TSS. To analyze transcription start sites as if they were peaks, use **"tss"** as the first argument.

```
annotatePeaks.pl tss hg18 ...
```

To restrict the analysis to a subset of TSS promoters, add the option **"-list <file>"** where the file is a Tab-delimited text file with the first column containing gene identifiers. TSS mode also works with custom gene definitions specified with **"-gtf <GTF file>"**.

Specifying the Peak Size - the most important parameter

Pretty much everything explained in the following sections depends heavily on the **"-size <#>"** parameter. A couple of quick notes:

- size <#> : will perform analysis on the # bp surrounding the peak centers [example: -size 1000]
- size <#,#> : will perform analysis from # to # relative to peak center [example: -size -200,50]
- size given : will perform analysis on different sized peaks - size given by actual coordinates in peak/BED file [example: -size given]

For example, if you peaks are actually transcription start sites, you might want to specify **"-size -500,100"** to perform the analysis upstream -500 bp to +100 bp downstream. If your peaks/regions are actually "transcript" regions, specifying **"-size given"** will count reads along the entire transcript. If it doesn't make sense, watch Delta Force I and II back to back. That should numb the brain enough to get it.

Annotating Individual Peaks

Calculating ChIP-Seq Tag Densities across different experiments

annotatePeaks.pl is useful program for cross-referencing data from multiple experiments. In order to count the number of tags from different sequencing experiments, you must first [create tag directories](#) for each of these experiments. Once created, tag counts from these directories in the vicinity of your peaks can be added by specifying **"-d <tag directory 1> <tag directory 2> ..."**. You can specify as many tag directories as you like. Tag totals for each directory will be placed in new columns starting on column 18. For example:

```
annotatePeaks.pl pu1peaks.txt mm8 -size 400 -d Macrophage-PU.1/ Bcell-PU.1/ > output.txt
```

output.txt, when opened in EXCEL, will look like this:

N	O	P	Q	R	S	T	U	V	W	X	Y
est Refs	Nearest Ense	Gene Name	Gene Alias	Gene Descrip	Macrophage-PU.1/	Tag C	Bcell-PU.1/	Tag Count in 400 bp	(8861792 Total, normalization factor = 1.13, effe		
01010133	ENSMUSG0000	Gm11487	OTTMUSG000	predicted ger	3268.14		8209.4				
01010455	ENSMUSG0000	Zfp868	AI449175	AV zinc finger pr	4919.21		11197.51				
025423	ENSMUSG0000	1110059E24f	-	RIKEN cDNA 1	3.78		1005.44				
010924	ENSMUSG0000	Nnmt	-	nicotinamide	935.1		1751.34				
011412	ENSMUSG0000	Slit3	Slit2	slit homolog	3.78		804.58				
077662	ENSMUSG0000	Ctso	A330105D01	cathepsin O	5.67		920.81				
028810	ENSMUSG0000	Rnd3	2610017M01	Rho family G	846.32		1663.32				
011518	ENSMUSG0000	Syk	-	spleen tyrosi	627.18		1649.78				

HOMER automatically normalizes each directory by the total number of mapped tags such that each directory contains **10 million tags**. This total can be changed by specifying **"-norm <#>"** or by specifying **"-noadj"**, which will skip this normalization step.

The other important parameter when counting tags is to specify the size of the region you would like to count tags in with **"-size <#>"**. For example, **"-size 1000"** will count tags in the 1kb region centered on each peak, while **"-size 50"** will count tags in the 50 bp region centered on the peak (default is 200). The number of tags is not normalized by the size of the region.

One last thing to keep in mind is that in order to fairly count tags, HOMER will automatically center tags based on their estimated ChIP-fragment lengths. This is can be overridden by specifying a fixed ChIP-fragment length using **"-len <#>"** or **"-fragLength <#>"**. This is important to consider when trying to count RNA tags, or things such as 5' RNA CAGE/TSS-Seq, where you may want to specify **"-len 0"** so that HOMER doesn't try to move the tags before counting them.

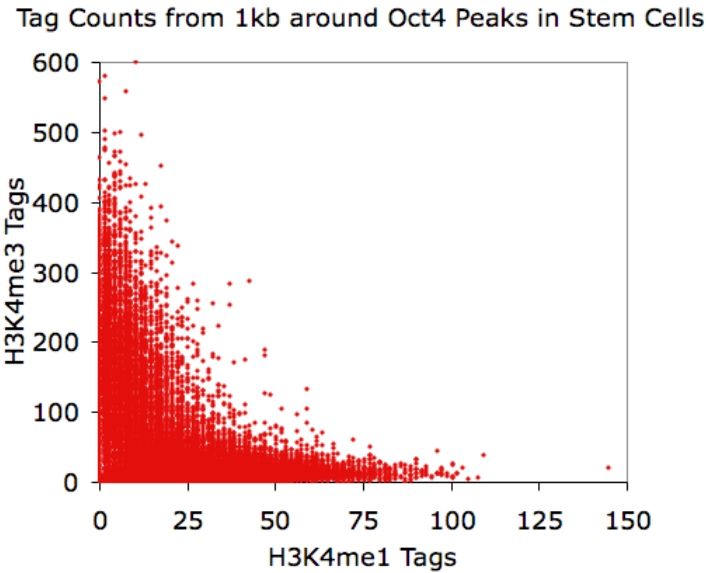
Making Scatter Plots

X-Y scatter plots are a great way to present information, and by counting tag densities from different tag directories, you can visualize the relative levels of different sequencing experiments. Because of the large range of values, it can be difficult to appreciate the relationship between data sets without log transforming the data (or sqrt to stay Poisson friendly). Also, due to the digital nature of tag counting, it can be hard to properly assess the data from a X-Y scatter plot since may of the data points will have the same values and overlap. To assist with these issues, you can specify **"-log"** or **"-sqrt"** to transform the data. These functions will actually report "log(value+1+rand)" and "sqrt(value+rand)", respectively, where rand is a random

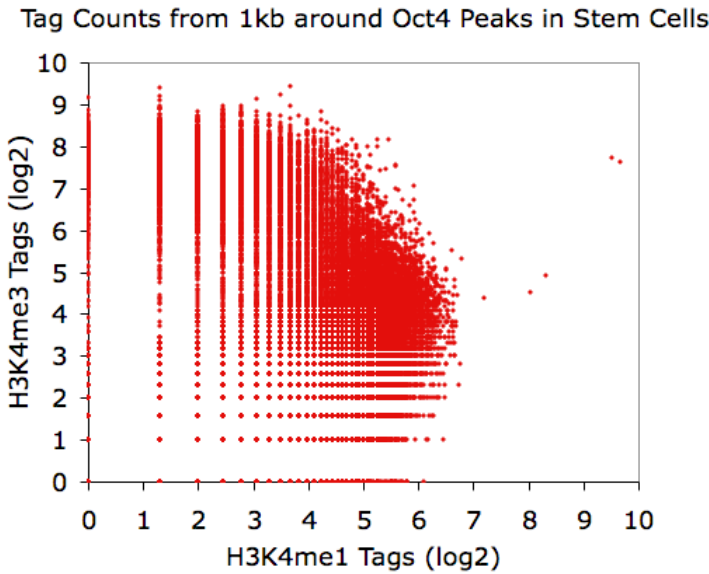
"fraction of a tag" that adds *jitter* to your data so that data points with low tag counts will not have exactly the same value. For example, lets look at the distribution of H3K4me1 and H3K4me3 near Oct4 peaks in mouse embryonic stem cells:

```
annotatePeaks.pl Oct4.peaks.txt mm8 -size 1000 -d H3K4me1-ChIP-Seq/ H3K4me3-ChIP-Seq/ > output.txt
```

Opening output.txt with EXCEL and plotting the last two columns:



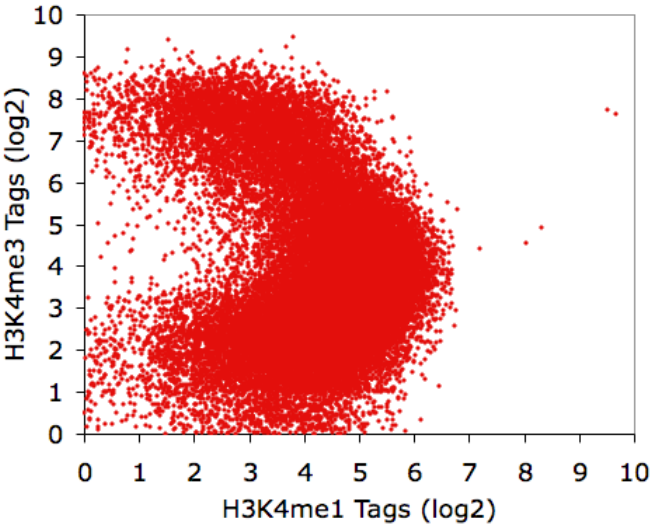
Using EXCEL to take the log(base 2) of the data:



Now using the "-log" option:

```
annotatePeaks.pl Oct4.peaks.txt mm8 -size 1000 -log -d H3K4me1-ChIP-Seq/ H3K4me3-ChIP-Seq/ > output.txt
```

Tag Counts from 1kb around Oct4 Peaks in Stem Cells



Believe it or not, all of these X-Y plots show the same data. Interesting, eh?

Finding instances of motifs near peaks

Figuring out which peaks have instances of [motifs found with findMotifsGenome.pl](#) is very easy. Simply use **"-m <motif file1> <motif file 2>..."** with **annotatePeaks.pl**. (Motif files can be concatenated into a single file for ease of use) This will search for each of these motifs near each peak in your peak file. Use **"-size <#>"** to specify the size of the region around the peak center you wish to search. Found instances of each motif will be reported in additional columns of the output file. For example:

```
annotatePeaks.pl pu1peaks.txt mm8 -size 200 -m pu1.motif cebp.motif > output.txt
```

Opening output.txt with EXCEL:

O	P	Q	R	S	T	U	V
arrest	Ense	Gene Name	Gene Alias	Gene Descrip	CpG%	GC%	PU.1/ThioMac-PU.1-ChIP-Seq/Homer Distance From Peak(sequence,strand,conservation)
3MUSG000	Dsm1	1700022L091	DSN1, MIND 1	0.015	0.621891	-55(GGAGGAAAGTG,+0.00),-26(TGGGGAAAGTG,+0.00),35(GCAGGAAAGTG,+0.00)	CEBP/CEBPb-ChIP-Seq/Homer Dist
3MUSG000	Cd53	AI323659	CD53 antigen	0.01	0.432836	34(TGAGGAAAGTG,+0.00)	
3MUSG000	Atg4c	App4-C	App4 autophagy-re	0.01	0.378109	8(AGAGGAAAGTG,+0.00),54(CACTTCCTCT,-0.00)	
3MUSG000	Tax1bp1	1200003J11F	Tax1 (human	0.019608	0.307692		-28(ATTTCATAAC,+0.00)
3MUSG000	Plekho2	AI840980	plekstrin ho	0.005	0.532338	-11(TGGGGAAAGTG,+0.00),33(TGAGGAAAGTG,+0.00)	
3MUSG000	Cttnbp2nl	AA552995	CTTNBP2 N-b	0.01	0.462687	-97(GGAGGAAAGTG,+0.00),48(ACAGGAAAGTG,+0.00)	
3MUSG000	Pitp	OD107	phospholipid	0.005	0.467662	29(CAGTTCTCTT,-0.00)	
3MUSG000	Ifngr2	Ifgr2	interferon ga	0.01	0.462687	-24(CACTTCCTCA,-0.00),49(CACTTCCTCA,-0.00)	
3MUSG000	Aff2	Fmr2	Ox19(AFA/FMR2 fai	0	0.512438		
3MUSG000	Syk	-	spleen tyrosi	0.01	0.497512	40(AGGGGAAAGTG,+0.00),60(GGAGGAAAGTG,+0.00)	
3MUSG000	Exoc6	4833405E05	exocyst comp	0.015	0.432836	-41(CAGTTCTCTCT,-0.00)	
3MUSG000	Ccdc109b	9030408N13	coiled-coil do	0.005	0.482587	71(GGAGGAAAGTG,+0.00)	53(6TTGTGTAAG,-0.00)
3MUSG000	Ctnd2	Catnd2	Npraj catenin (cad	0.021858	0.472826	-54(AGCGGAAAGTG,+0.00),18(CAGTTCTCTGT,-0.00),34(CACTTCCTCT,-0.00)	64(6TTTCATAAT,-0.00)
3MUSG000	Slc7a1	Glut1	Glut1	0.02	0.547564	-28(AGAGGAAAGTG,+0.00),-16(CACTTCCTCT,-0.00)	

Each instance of the motif is specified in the following format (separated by commas):

Distance from Peak Center(Sequence Matching Motif,Strand,Average Conservation)

The average conservation will not be reported unless you specify **"-cons"**. Also, when finding motifs, the average CpG/GC content will automatically be reported since it has to extract peak sequences from the genome anyway.

There are also a bunch of motif specific options for specialized analysis:

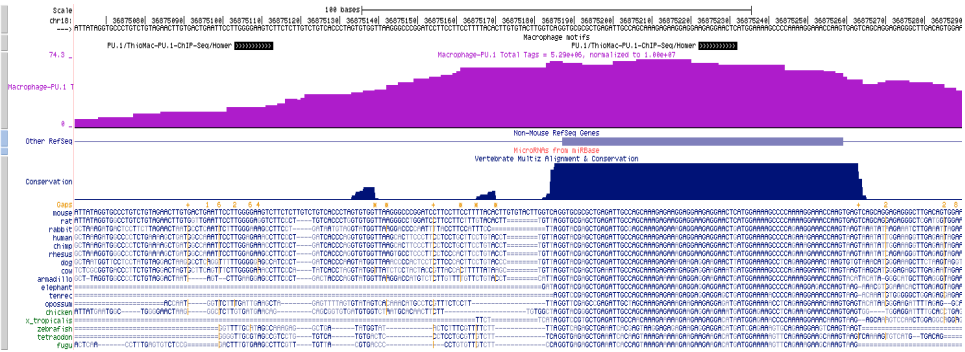
- "-norevopp" (only search + strand relative to peak strand for motifs)
- "-nmotifs" (just report the total number of motifs per peak)
- "-mscore" (report the maximum log-odds score of the motif in each peak)
- "-rmrevopp <#>" (tries to avoid double counting reverse opposites within # bp)
- "-mdist" (reports distance to closest motif)
- "-fm <motif file 1> [motif file 2]" (list of motifs to filter out of results if found)
- "-mfasta <filename>" (reports sites in a fasta file - for building new motifs)
- "-mbed <filename>" (Output motif positions to a BED file to load at UCSC - see below)
- "-matrix <filename>" (outputs a motif co-occurrence matrix with the p-value of co-occurrence assuming instance of each motif are independently distributed amongst the peaks)

Visualizing Motif positions in the UCSC Genome Browser

This feature may seem slightly out of place, but since **annotatePeaks.pl** is the workhorse of HOMER, you can add **"-mbed <filename>"** in conjunction with **"-m <motif file 1> [motif file 2] ..."** to produce a BED file describing motif positions near

peaks that can be loaded as a custom track in the UCSC genome browser. In the example below, you would load `motif.bed` as a custom track:

```
annotatePeaks.pl pu1peaks.txt mm8 -size 200 -m pu1.motif cebp.motif -mbed motif.bed > output.txt
```



Finding the distance to other sets of Peaks

In order to find the nearest peak from another set of peaks, use `-p <peak file 1> [peak file 2] ...`. This will add columns to the output spreadsheet that will specify the nearest peak ID and the distance to that peak. If all you want is the distance (so you can sort this column), add the option `-pdist` to the command. Otherwise, if you prefer to count the number of peaks in the peak file found within the indicated regions (i.e. with `-size <#>`), add `-pcount`.

Creating Histograms from High-throughput Sequencing data and Motifs

HOMER can be used to make histograms that document sequencing library and motif densities relative to specific positions in the genome. This can be done near peaks, subsets of peaks, or near promoters, exon junctions or anywhere else you find interesting. To make histograms, use the `annotatePeaks.pl` program but add the parameters `"-hist <#>"` to produce a tab delimited text file that can then be visualized using EXCEL or other data visualization software.

Basic usage:

```
annotatePeaks.pl <peak file> <genome> -size <#> -hist <#> -d <tag directory 1> [tag directory2] ... -m <motif 1> <motif 2> ... > <output matrix file>
```

i.e. `annotatePeaks.pl ERpeaks.txt hg18 -size 6000 -hist 25 -d MCF7-H3K4me1/ MCF7-H3K4me2/ MCF7-H3K4me3/ > outfile.txt`

Running this command is very similar to creating annotated peak files - in fact, most of the data can be used to make both types of files - hence the reason for combining this functionality in the same command. Be default, HOMER normalizes the output histogram such that the resulting units are **per bp per peak**, on top of the standard total mapped tag normalization of **10 million tags**.

Histograms of Tag Directories:

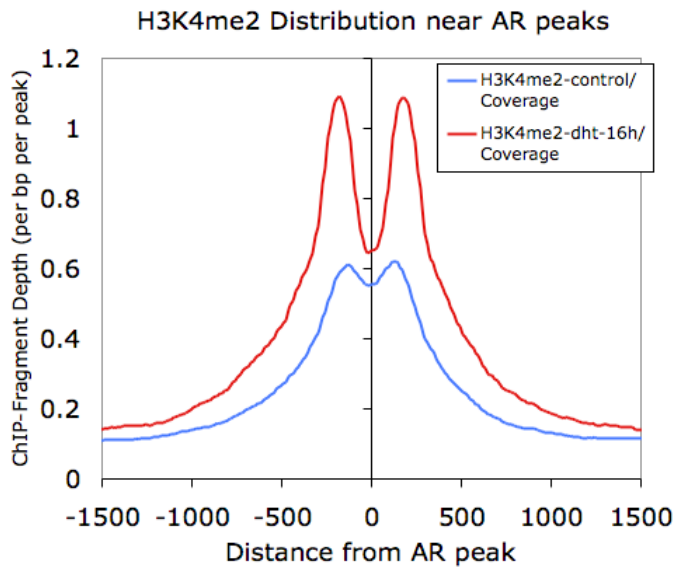
For each tag directory or motif, HOMER will output 3 columns in the histogram. In the case of tag directories, the first column will indicate **ChIP-Fragment Coverage**, which is calculated by extending tags by their estimated ChIP-fragment length, and is analogous to the profiles made for the UCSC Genome Browser. The 2nd and 3rd columns report the density of 5' and 3' independent tags, and are independent of fragment length. For example, lets look at H3K4me2 distribution near Androgen Receptor (AR) peaks before and after 16 hours of treatment with testosterone (dht):

```
annotatePeaks.pl ARpeaks.txt hg18 -size 4000 -hist 10 -d H3K4me2-control/ H3K4me2-dht-16h/ > outfile.txt
```

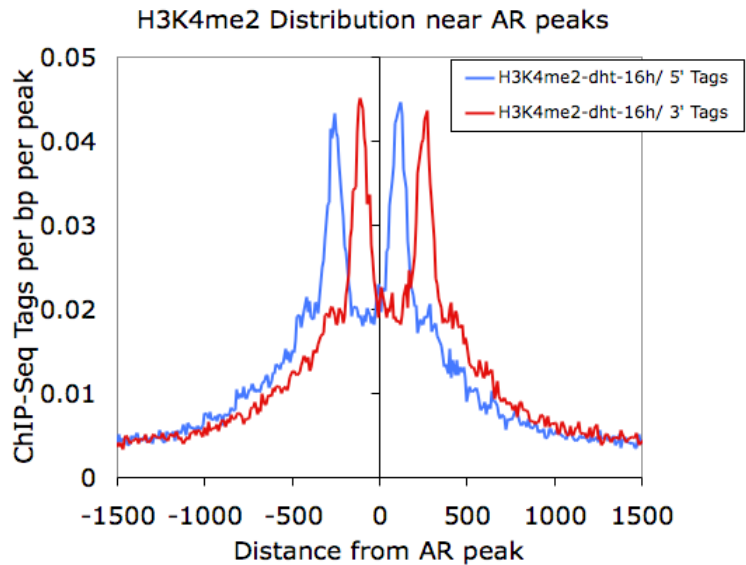
Opening outfile.txt with EXCEL, we see:

	A	B	C	D	E	F	G
1	Distance from Center H3K4me2-control/	Coverage H3K4me2-control/	5' Tags H3K4me2-control/	3' Tags H3K4me2-dht-16h/	Coverage H3K4me2-dht-16h/	5' Tags H3K4me2-dht-16h/	3' Tags
2	-2000	0.046971	0.001564	0.001734	0.059736	0.001824	0.002136
3	-1990	0.052394	0.003621	0.003179	0.06552	0.004344	0.002928
4	-1980	0.05661	0.003689	0.003298	0.070872	0.004008	0.003528
5	-1970	0.059721	0.003026	0.003128	0.075048	0.003816	0.003912
6	-1960	0.063614	0.003638	0.002975	0.079128	0.00396	0.00372
7	-1950	0.066742	0.002958	0.003298	0.084048	0.004392	0.00372
8	-1940	0.069751	0.003315	0.003145	0.087792	0.004128	0.003432
9	-1930	0.072403	0.003281	0.00323	0.091416	0.003312	0.004632
10	-1920	0.075259	0.002924	0.002958	0.0936	0.003504	0.004224

Graphing columns B and E while using column A for the x-coordinates, we get the following:



However, if we graph only the 5' and 3' tags that come from the H3K4me2-dht-16h directory (columns F and G):



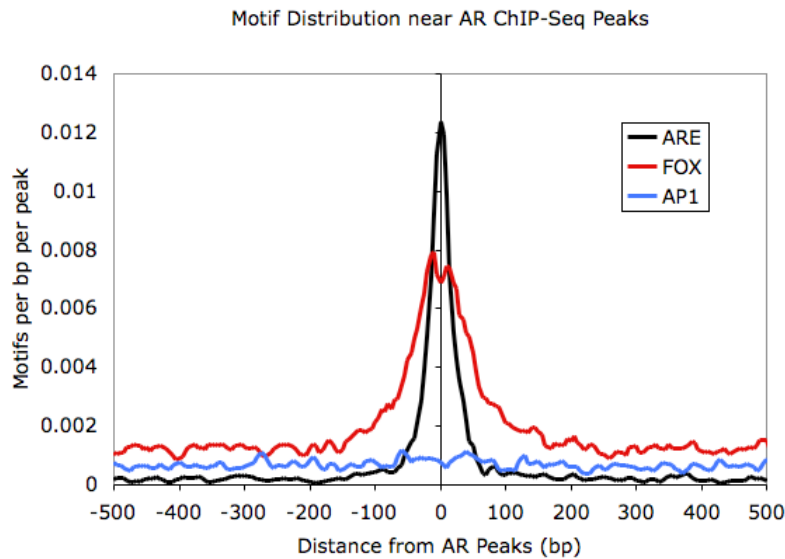
Here we can see how the 5' and 3' reads from the H3K4me2 marked nucleosomes are distributed near the AR sites.

Histograms of Motif Densities:

Making histograms out of motif occurrences is very similar to sequencing tag distributions. Run the `annotatePeaks.pl` program with `"-hist <#>"` and `"-m <motif file>"` (you can also find motif densities and tag densities at the same time):

```
annotatePeaks.pl ARpeaks.txt hg18 -size 1000 -hist 5 -m are.motif fox.motif ap1.motif > outputfile.txt
```

Graphing `outputfile.txt` with EXCEL:



Centering Peaks on Motifs

One cool analysis strategy is to center peaks on a specific motif. For example, by centering peak for the Androgen Receptor Transcription Factor on the ARE motif (GNACANNNTGTNC), you can map the spacial relationship between the motif and other sequence features and sequencing reads.

To center peaks on a motif, run **annotatePeaks.pl** with the following options:

```
annotatePeaks.pl <peak file> <genome> -size <#> -center <motif file> > newpeakfile.txt
```

i.e. **annotatePeaks.pl** ARpeaks.txt hg18r -size 200 -center are.motif > areCenteredPeaks.txt

Now the idea is to use the new peak file to perform analysis:

```
annotatePeaks.pl areCenteredPeaks.txt hg18 -size 6000 -hist 10 -d H3K4me2-notx/ ... > output.txt
```

Creating Heatmaps from High-throughput Sequencing data

HOMER is not capable of generating actual Heatmaps *per se*, but it will generate the data matrix (similar to a gene expression matrix) than can then be visualized using standard gene expression heatmap tools. For example, I will generate a heatmap data matrix file using HOMER, and then open it with [Cluster 3.0 \(Micheal Eisen/de Hoon\)](#) to cluster it and/or visualize it with [Java Tree View \(by Alok J. Saldanha\)](#). In reality, you can use any clustering and/or heatmap visualization software (i.e. R).

Basic usage (add **"-ghist"** when making a histogram):

```
annotatePeaks.pl <peak file> <genome> -size <#> -hist <#> -ghist -d <tag directory 1> [tag directory2] ... > <output matrix file>
```

i.e. **annotatePeaks.pl** ARpeaks.txt hg18 -size 6000 -hist 25 -ghist -d H3K4me2-control/ H3K4me2-dht-16h/ > outputfile.txt

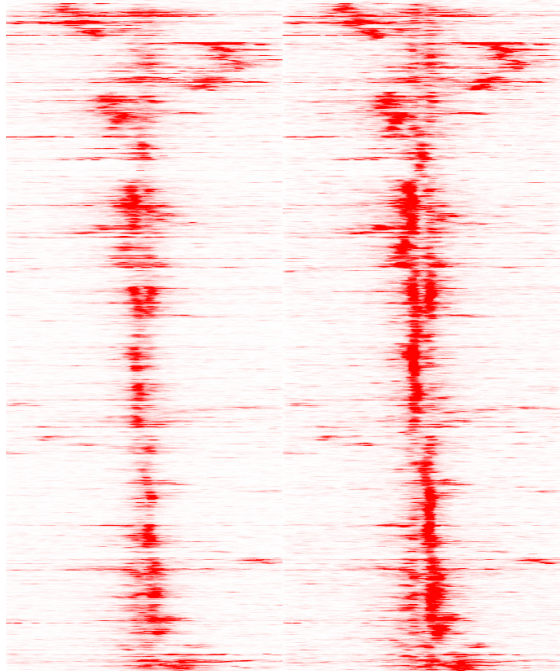
Running this command is very similar to making histograms with **annotatePeaks.pl**. In fact, a heatmap isn't really all that different from a histogram - basically, instead of averaging all of the data from each peak, we keep data from each peak separate and visualize it all together in a heatmap. The key difference when making a heat map or a histogram is that you must add **"-ghist"** when making a heatmap.

Format of Data Matrix Output File

The resulting file is a tab-delimited text file where the first row is a header file and the remaining rows represent each peak from the input peak file. The first set of columns will describe the read densities from the **first tag directory** as a function of distance from the center of the peak, with bin sizes corresponding to the parameter used with **"-hist <#>"**. After the first block of columns, a second block will start over with read densities from the **2nd tag directory**, and so on. If you would like to cluster this file to help organize the patterns, make sure you only cluster the "genes" (i.e. rows).

Example: **annotatePeaks.pl** ARpeaks.txt hg18 -size 6000 -hist 25 -ghist -d H3K4me2-control/ H3K4me2-dht-16h/ > outputfile.txt

After creating the file, I loaded into Cluster 3.0 to cluster and clustered the genes using "centered correlation" as the distance metric, then loaded that output into Java Tree View.



General Options to Control Data Analysis Behavior

- strand** <+|-|both> (only look for motif etc. on + or - strand, default both)
- fragLength** <#> (Fragment length, default=auto, might want to set to 0 for RNA)
- pc** <#> (maximum number of tags to count per bp, default=0 [no maximum])
- cons** (Retrieve conservation information for peaks/sites - creates new column for this information)
- CpG** (Calculate CpG/GC content)
- norevopp** (do not search for motifs on the opposite strand [works with -center too])
- norm** <#> (normalize tags to this tag count, default=1e7, 0=average tag count in all directories)
Use **-noadj** to disable tag normalization for sequencing depth

Command line options for annotatePeaks.pl

Usage: `annotatePeaks.pl <peak file | tss> <genome version> [additional options...]`

Available Genomes (required argument): (name,org,directory,default promoter set)

User defined annotation files (default is UCSC refGene annotation):

`annotatePeaks.pl` accepts GTF (gene transfer formatted) files to annotate positions relative to custom annotations, such as those from de novo transcript discovery or Gencode.

`-gtf <gtf format file>`

Peak vs. tss / tts mode (works with custom GTF file):

If the first argument is "tss" (i.e. `annotatePeaks.pl tss hg18 ...`) then a TSS centric analysis will be carried out. Tag counts and motifs will be found relative to the TSS. (no position file needed) ["tts" now works too - e.g. 3' end of gene]

NOTE: The default TSS peak size is 4000 bp, i.e. +/- 2kb (change with `-size` option)

`-list <gene id list>` (subset of genes to perform analysis [unigene, gene id, accession, probe, etc.], default = all promoters)

Primary Annotation Options:

- m** <motif file 1> [motif file 2] ... (list of motifs to find in peaks)
- m**score (reports the highest log-odds score within the peak)
- n**motifs (reports the number of motifs per peak)
- m**dist (reports distance to closest motif)
- m**fasta <filename> (reports sites in a fasta file - for building new motifs)
- f**m <motif file 1> [motif file 2] (list of motifs to filter from above)
- r**mrevopp <#> (only count sites found within <#> on both strands once, i.e. palindromic)
- m**atrix <filename> (outputs a motif co-occurrence matrix)
- m**bed <filename> (Output motif positions to a BED file to load at UCSC (or `-mpeak`))

- d <tag directory 1> [tag directory 2] ... (list of experiment directories to show tag counts for) NOTE: -dfile <file> where file is a list of directories in first column
- p <peak file> [peak file 2] ... (to find nearest peaks)
 - pdist to report only distance (-pdist2 gives directional distance)
 - pcount to report number of peaks within region
- gene <data file> ... (Adds additional data to result based on the closest gene. This is useful for adding gene expression data. The file must have a header, and the first column must be a GeneID, Accession number, etc. If the peak cannot be mapped to data in the file then the entry will be left empty.
- go <output directory> (perform GO analysis using genes near peaks)
- genomeOntology <output directory> (perform genomeOntology analysis on peaks)
- gsiz <#> (Genome size for genomeOntology analysis, default: 2e9)

Annotation vs. Histogram mode:

- hist <bin size in bp> (i.e 1, 2, 5, 10, 20, 50, 100 etc.)
- The -hist option can be used to generate histograms of position dependent features relative to the center of peaks. This is primarily meant to be used with -d and -m options to map distribution of motifs and ChIP-Seq tags. For ChIP-Seq peaks for a Transcription factor you might want to use the -center option (below) to center peaks on the known motif
- ** If using "-size given", histogram will be scaled to each region (i.e. 0-100%), with the -hist parameter being the number of bins to divide each region into.
- Histogram Mode specific Options:
- nuc (calculated mononucleotide frequencies at each position, Will report by default if extracting sequence for other purposes like motifs)
 - di (calculated dinucleotide frequencies at each position)
 - histNorm <#> (normalize the total tag count for each region to 1, where <#> is the minimum tag total per region - use to avoid tag spikes from low coverage)
 - ghist (outputs profiles for each gene, for peak shape clustering)
 - rm <#> (remove occurrences of same motif that occur within # bp)

Peak Centering: (other options are ignored)

- center <motif file> (This will re-center peaks on the specified motif, or remove peak if there is no motif in the peak. ONLY recentering will be performed, and all other options will be ignored. This will output a new peak file that can then be reanalyzed to reveal fine-grain structure in peaks (It is advised to use -size < 200) with this to keep peaks from moving too far (-mirror flips the position))
- multi (returns genomic positions of all sites instead of just the closest to center)

Advanced Options:

- len <#> / -fragLength <#> (Fragment length, default=auto, might want to set to 0 for RNA)
- size <#> (Peak size[from center of peak], default=inferred from peak file)
 - size #,# (i.e. -size -10,50 count tags from -10 bp to +50 bp from center)
 - size "given" (count tags etc. using the actual regions - for variable length regions)
- log (output tag counts as log2(x+1+rand) values - for scatter plots)
- sqrt (output tag counts as sqrt(x+rand) values - for scatter plots)
- strand <+|-|both> (Count tags on specific strands relative to peak, default: both)
- pc <#> (maximum number of tags to count per bp, default=0 [no maximum])
- cons (Retrieve conservation information for peaks/sites)
- CpG (Calculate CpG/GC content)
- ratio (process tag values as ratios - i.e. chip-seq, or mCpG/CpG)
- norevopp (do not search for motifs on the opposite strand [works with -center too])
- noadj (do not adjust the tag counts based on total tags sequenced)
- norm <#> (normalize tags to this tag count, default=1e7, 0=average tag count in all directories)
- pdist (only report distance to nearest peak using -p, not peak name)
- noann, -nogene (skip genome annotation step, skip TSS annotation)
- homer1/-homer2 (by default, the new version of homer [-homer2] is used for finding motifs)



Can't figure something out? Questions, comments, concerns, or other feedback:
cbenner@ucsd.edu



HOMER

Software for motif discovery and next-gen sequencing analysis

Quantifying RNA with analyzeRNA.pl

Homer contains a program (**analyzeRNA.pl**) to quantify RNA reads in genes. Of all the tools in HOMER, this one is probably the least efficient and resource hungry program in terms of memory. It needs to be rewritten but I have not found the time or pressing need to do this yet. However, it does perform useful calculations, particularly for GRO-Seq applications, that are not available or not automated for other programs.

analyzeRNA.pl DOES produces a gene expression matrix from "**Tag Directories**", and works in a similar manner to how annotatedPeaks.pl works for **ChIP-Seq** data. It has options to count reads in "genic" vs. "exon" regions, and attempts to automate the analysis of promoter proximal pausing seen in GRO-Seq data.

analyzeRNA.pl DOES NOT produce differential expression or differential splicing calls. It would be great if Chuck had the time to help me with that, but for now I recommend sending the output from analyzeRNA.pl to other utilities such as [edgeR](#) or [DEseq](#) or whatever your favorite next-gen differential signal finder is.

Basic usage:

```
analyzeRNA.pl <rna|repeats|gtf file> <genome> [options] -count [genes|exons|introns] -d <Tag Directory> [Tag Directory 2] ... > <output file>
```

i.e. `analyzeRNA.pl rna hg18 -count genes -d IMR90-GroSeq > outputfile.txt`

Specifying the Genes/Transcripts to Analyze

The first argument to analyzeRNA.pl specifies which "gene definition" to use for analysis. There are three options:

1. **rna** - This will direct HOMER to analyze the default RefSeq annotation for that genome.
2. **repeats** - HOMER will load repeat definitions from UCSC and assess the expression levels of different repeat classes. This can be resource hungry and is not recommended unless your computer has a bunch of memory.
3. **<GTF file>** - Specify your own custom genes. These can be an alternative annotation (i.e. Ensembl genes, UCSC genes) or it can be custom, de novo genes you found analyzing GRO-Seq or mRNAs defined using "cufflinks". If you are looking for a GTF file for your favorite annotation, check out the [UCSC Table Browser](#). You can use this tool to download GTF formatted files (with the correct genome version) for many popular annotations.

Examples include "analyzeRNA.pl rna mm9 ..." or "analyzeRNA.pl myGenes.gtf mm9 ...".

Choosing Experiments to Analyze

As with all HOMER programs, you need to create "Tag Directories" out of each experiment you want to quantify first. To quantify them with **analyzeRNA.pl** simply add one or more directory after "-d". For example:

```
analyzeRNA.pl rna hg18 -d LNCaP-RNA-Seq-notx/ LNCaP-RNA-Seq-Dht/ > outputfile.txt
```

This will produce an output file containing genes expression values for two experiments, "LNCaP-RNA-Seq-notx" and "LNCaP-RNA-Seq-Dht".

Measuring Gene Expression in Exons vs. Gene Bodies.

Depending on the type of sequencing you are analyzing, you will want to quantify RNA from different parts of the gene. The "-count [...]" option controls which regions of the gene are used for analysis (use like **"-count exons"** or **"-count genes"**)

- **exons** (default) Counts tags in exons only. Use this for most applications of RNA-Seq, such as polyA-RNA-seq or other techniques that aim to measure mRNA.
- **cds** - Counts tags in coding regions only. This could be useful for quantifying ribosome coverage on coding sequences with techniques such as Ribo-Seq
- **introns** - Counts tags on introns only.
- **5utr** or **3utr** - Count tags on 5' UTR and 3' UTR regions, respectively
- **genes** - Counts tags on the full gene body (TSS to TTS). This is useful for GRO-Seq where we expect coverage across the entire transcript. Can also be used to quantify H3K36me3 or PolII ChIP-Seq.

By default, analyzeRNA.pl assumes your RNA is strand-specific. If it is not, or you are using analyzeRNA.pl for other types of data such as ChIP-Seq, you can specify:

- "-strand +"** - default, only measure + strand (relative to gene orientation), for strand specific RNA
- "-strand -"** - only measure - strand (relative to gene orientation)
- "-strand both"** - count reads on both strands, for non-strand specific RNA or ChIP-Seq

Also, there are a couple options to slightly modify how tags are counted to avoid TSS/TTS features. By specifying **"-start <#>"**, you can adjust where (relative to the TSS) HOMER starts counting reads. This is useful for GRO-Seq when you might want to avoid the promoter-proximal paused region by add **"-start 500"** to start counting 500 bp downstream from the TSS. Negative values would start counting 500 bp upstream. The option **"-end <#>"** is similar, but applies to the TTS. Negative values will stop the tag counting upstream of the TTS (**"-end -500"**), while positive values will count tags further downstream.

Normalization of Gene Expression Values

Normalization is probably the trickiest part about RNA-Seq. There are lots of papers on it. HOMER offers a couple different normalization options depending on what the experiment is and what your needs are.

By default, HOMER normalizes each experiment to 10 million mapped reads, which is the same normalization strategy used in **annotatePeaks.pl** for ChIP-Seq data. However, RNA has the potential to contain much more "contaminates" than ChIP-Seq. In ChIP-Seq, the background is the genome, with random fragments coming down in the immunoprecipitation step. With RNA, instead of a two copies of the genome per cell, you have 99% ribosomal RNA, along with a host of other very very common short RNA species such as tRNAs, snoRNAs, and other problematic things, and then a small fraction of what you're actually interested in. Most RNA-Seq protocols contain enrichment steps, such as polyA selection, to isolate mRNAs from the rest of the crap (my apologies to those studying rRNA and tRNA). These enrichment steps can have different efficiencies from sample to sample, with some samples containing more rRNA, tRNA etc. than others. When mapping to the genome, many of these RNA species will be discarded if you only keep reads that map uniquely since the genomic elements that create them are often repeated in the genome. However, you're bound to still map many of these reads. Differential contamination of common RNA species can throw off the default normalization - if 50% of your "mapped" reads are rRNA in one sample and only 25% in another, you're likely going to have problems.

There are several things you can do to combat these problems. One is to "pre-clear" common RNA species computationally, i.e. map your reads to rRNA first to remove them, then map the rest to the genome. Another strategy is to normalize strictly with mRNA species only instead of considering the total number of mapped reads. In general, I would recommend normalizing in this strategy (**"-normMatrix <#>"**) - it's probably the safest in most situations due to the contamination problem. Below are the various normalization options in **analyzeRNA.pl**.

- **-norm <#>** - Normalize the total number of mapped reads per experiment to #, this is the default **"-norm 1e7"**.
- **-normMatrix <#>** - Normalize the total of number of reads found in the gene expression matrix to # (i.e. normalize total reads in mRNAs)
- **-noadj** - Don't perform any normalization, just report the exact number of reads found. This is useful when sending the output of HOMER to another differential expression program such as edgeR or similar that requires the raw read counts.

The options above perform the actual normalization. "Dress up" normalization/transformations options follow:

- **-rpkm** - Report normalized values as reads per kilobase per million mapped reads

- **-log** - Report log normalized values with jitter = $\log_2(x+1+\text{rand}(0-1))$
- **-sqrt** - Report square root of the normalized value with jitter = $\sqrt{x+\text{rand}(0-1)}$

Limiting the number of Tags per Position ("-pc <#>")

Another important parameter (that I almost made mandatory) is **"-pc <#>"**, which controls the maximum number of reads to consider per position. In the case of some genes, there may be high-abundance RNA species in introns, where large spikes in RNA reads may occur. If quantifying expression from gene bodies (i.e. **"-count genes"**, GRO-Seq), these reads will be counted toward the gene's total abundance. Ideally these are removed. However, as a quick and dirty solution to this problem, you can limit the number of reads that HOMER counts from each position. In many cases rRNA loci will have tens of thousands of reads per bp. On the flip side, limiting the number of reads per bp can kill your dynamic range. Usually the middle ground such as **"-pc 3"** is a good way to go - you effectively remove the super spikes but still allow genes with high RPKM to "express themselves" in your analysis.

How analyzeRNA.pl Works

The following are the steps HOMER takes to produce a Gene Expression Matrix:

1. Parses the RNA definition file to figure out where all of the genes are in the genome.
2. Queries each of the "Tag Directories" and determine where each of the tags are within gene bodies.
3. At that point it will rummage through the tag positions and count only the tags found within the desired regions (i.e. exons). Incidentally, this is also the part that makes it less efficient at the moment - it's done with perl and keeps way too much extra information around.
4. Expression values are calculated and normalized
5. Results are formatted and set to *stdout*

Description of Output Files

The output gene expression matrix with the following columns:

1. Transcript ID (RefSeq accession is the default with "rna" option, custom name is used otherwise)
2. chromosome
3. start
4. end
5. strand
6. mRNA length (e.g. exons only)
7. gene length (TSS to TTS)
8. Copies in genome (some genes map to the genome more than once...)
9. Symbol (e.g Gene Name)
10. Alias (alternative gene names)
11. Description
12. Unigene
13. Entrez Gene ID
14. Ensembl
15. Data: First Tag Directory Gene Expression information
16. [Data: Second Tag Directory Gene Expression information]
17. ...

A separate column is generated for each tag directory. The head of these columns contains the name, the region of the gene used for expression (i.e. gene, exon, 5utr...), the total number of mapped tags in the directory, and the normalization factor.

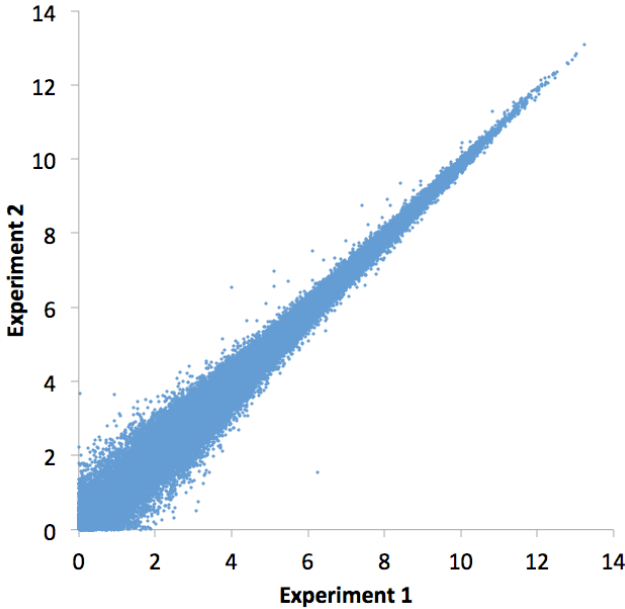
Due to the fact that some genes map to multiple places in the genome (<1% of RefSeq), only one of their locations will be reported. However, the gene length and mRNA lengths will be averaged, and their read totals will be averaged in the output.

If custom IDs are used, HOMER will attempt to link them to known gene identifiers (As of now it does not annotate their position, just treats their ID as an accession number and tries to match it with known genes). If you give analyzeRNA.pl a custom GTF file from Cufflinks, where the IDs are all "CUFF12345.1", most of the annotation columns will be blank.

Below is an example of the output opened with EXCEL:

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	
1	Genes	chr	start	end	stran	mRNA Length	Gene Length	Copies in	Symbol	Alias	Description	Unigene	GeneID	Ensembl	Macrophage-RNA-nc	Macrophage-RN	
2	NM_001001130	chr13	67848736	67856071	-	2218	7335	1	Zfp85-rs1	KRAB19	Rslc zinc finger pr	Mm.288396	22746	ENSMUSG	25.49128477	26.38801643	
3	NM_001001144	chr9	110235796	110287450	+	4286	51654	1	Scap	9530044G19	SREBF chape	Mm.288741	235623	ENSMUSG	82.8466755	160.357946	
4	NM_001001152	chr13	67355853	67370004	-	3488	14151	1	Zfp458	BC062958	R zinc finger pr	Mm.306358	238690	ENSMUSG	11.15243709	12.85570031	
5	NM_001001160	chr6	85419571	85452880	-	6562	33309	1	Fbxo41	9630017H13	F-box proteir	Mm.38777	330369	ENSMUSG	0	0	
6	NM_001001176	chrX	103402212	103415181	-	2583	12969	1	Taf9b	BC066223	TAF9B RNA p	Mm.19440	407786	ENSMUSG	0	0	
7	NM_001001177	chr17	34535764	34597679	+	1900	61915	1	BC051142	NG8	TSBP	T cDNA sequer	Mm.73205	407788	ENSMUSG	0.796602649	0
8	NM_001001178	chr2	58674108	58998684	-	3920	324576	1	Ccdc148	5830402J09	coiled-coil dc	Mm.432280	227933	ENSMUSG	0	0	
9	NM_001001179	chr6	128489839	128531624	-	4698	41785	1	BC048546	MGC58520	c cDNA sequer	Mm.259234	232400	ENSMUSG	0	0	
10	NM_001001180	chr7	147995575	148008077	-	3909	12502	1	Zfp941	-	zinc finger pr	Mm.359154	407812	ENSMUSG	0	0.676615806	
11	NM_001001181	chr18	75165553	75169587	+	1047	4034	1	BC031181	-	cDNA sequer	Mm.29866	407819	ENSMUSG	157.7273245	150.2087089	
12	NM_001001182	chr2	59737419	59963797	-	7980	226378	1	Baz2b	5830435C13	bromodomai	Mm.436730	407823	ENSMUSG	215.8793179	158.3280986	
13	NM_001001183	chr17	25194646	25218059	-	1779	23413	1	Tmem204	-	transmembri	Mm.34379	407831	ENSMUSG	1.593205298	0.676615806	
14	NM_001001184	chr8	47660947	47702554	-	3797	41607	1	Ccdc111	BC065112	N coiled-coil dc	Mm.217385	408022	ENSMUSG	145.7782848	63.60188576	
15	NM_001001185	chr13	67964273	67964854	+	581	581	1	BC048507	-	cDNA sequer	Mm.177840	408058	ENSMUSG	24.69468212	33.8307903	
16	NM_001001186	chr13	67464519	67476699	-	4079	12180	1	Zfp456	BC065418	N zinc finger pr	Mm.461583	408065	ENSMUSG	47.79615894	33.15417449	
17	NM_001001187	chr13	67768377	67784449	-	4504	16072	1	Zfp738	3830402I07R	zinc finger pr	Mm.435551	408068	ENSMUSG	218.2691258	174.5668779	
18	NM_001001189	chr9	64154562	64189064	-	3575	34502	1	Dis3l	AV340375	DIS3 mitotic	Mm.268341	213550	ENSMUSG	45.40635099	64.27850156	
19	NM_001001297	chr2	21127350	21136636	+	2327	9286	1	Thnsl1	AW413632	threonine syi	Mm.268841	208967	ENSMUSG	23.10147682	12.17908451	
20	NM_001001309	chr2	12028286	12223547	-	5782	195261	1	Itga8	AI447669	integrin alph	Mm.329997	241226	ENSMUSG	0	0	
21	NM_001001319	chr4	143649028	143659221	+	2399	10193	1	Pramel4	C79430	D4E1 preferentialh	Mm.271697	347710	ENSMUSG	0	0	
22	NM_001001320	chr19	3992751	3999512	+	1778	6761	1	Tbx10	Dc MGC129	T-box 10	Mm.246555	109575	ENSMUSG	0	0	
23	NM_001001321	chr13	64197617	64230638	-	2276	33021	1	Slc35d2	5730408I21R	solute carrier	Mm.133731	70484	ENSMUSG	42.2199404	37.21386932	
24	NM_001001322	chr2	26828935	26865145	+	4580	36210	1	Adamts13	Gm710	VWF a disintegrin	Mm.330084	279028	ENSMUSG	0	1.353231612	
25	NM_001001326	chr7	116667424	116760661	-	4255	93237	1	St5	2010004M01	suppression	Mm.252009	76954	ENSMUSG	140.9986689	59.54219092	

If you specify the **"-log"** or **"-sqrt"** options, you can make a nice X-Y scatter plot of the data columns. Here is an example:



Quantifying Promoter-proximal RNA Polymerase Pausing

One of the neat observations from GRO-Seq experiments is that an actively engaged RNA Polymerases are present at the promoters of many genes, but they do not seem to elongate down the body of the gene. The idea is that they are "paused", waiting for elongation signals to give them the green light to make the gene.



HOMER can calculate the "pausing" ratio, which is the ratio of tag density at the promoter relative to the body of the gene. To use this option, specify **"-pausing <#>"**, where # the is the distance downstream of the TSS to stop counting promoter tags and start counting gene body tags. A safe value is probably 250-500.

This will que the program to produce 3 columns of output per experiment. First, it will report the pausing ratio, and then it will report the promoter and gene body read densities.

Analyzing Repeat and non-mRNA Expression

mRNA is not the only RNA species present in the cell. In fact, it makes up a very small fraction of the total. **analyzeRNA.pl** offers an option to help quantify repeat RNAs, rRNA, etc. By specifying "**repeats**", HOMER will load the definition of all repeats in the genome and quantify tags on them. Within the repeat definition are tRNAs, rRNAs, etc. so you get more than just transposable elements. Use it like so:

analyzeRNA.pl repeats mm9 -d Macrophage-RNAseq > outputfile.txt

By default, all the repeats are treated like "exons". Repeats from similar classes are combined to give a single expression value. Only one of the positions are reported (randomly). Be careful! It uses a lot of memory! If you're running out of memory, try using fewer tag directories (i.e. analyze one at a time).

To do repeat expression justice, you should carefully consider how the data is mapped to the genome. Normally for ChIP-Seq (or even RNA-Seq), you do not want to consider reads that map to multiple locations in the genome. However, in the case of RNA repeats, this means that you will be discarding many of the reads mapping to repeat regions. One trick that works fairly well is to keep one random position from the mapping, regardless if it maps uniquely to the genome (but only do it for this type of analysis). Typically, if a read maps to multiple locations, those multiple locations are probably all the same type of repeat element, so it will be added to the expression of that repeat class regardless of where it is specifically placed. This is "approximate", so use with care.

The thing to note is that repeat/rRNA/tRNA expression is compounded by the fact that most protocols try to get rid of it, so depending on the efficiency of these clearing steps experiment-to-experiment, you may be measuring the difference in clearing efficiency rather than the actual expression of the RNA.

Command line options for analyzeRNA.pl

Usage: analyzeRNA.pl <rna | repeats | custom RNA/GTF file> <genome version> [additional options...]

Program for quantifying RNA tag counts. The first argument can be "rna" (refseq genes), "repeats" (repeat classes), or a custom RNA definition file. (see website for format)

!!! Right now "repeats" is not memory efficient, need to optimize - i.e. 20Gb !!!

!!! Only run "repeats" with a single tag directory for now !!!

Available Genomes (required argument): (name,org,directory,default promoter set)

Primary Annotation Options:

- d <tag directory 1> [tag directory 2] ... (list of experiment directories to show tag counts for) NOTE: -dfile <file> where file is a list of directories in first column
- rpkm (Report results as reads per kb per million mapped)
- norm <#> (Normalize to total mapped tags: default 1e7)
- normMatrix <#> (Normalize to total tags in gene expression matrix: not used)
- noadj (Don't normalize)
- count <exons|introns|genes|5utr|3utr|cds> (Count tags in introns, exons, etc., default: exons)
- noCondensing (do not condense counts from entries with same ID, default: do condense)
- pc <#> (maximum tags to count per position, default: 0=no limit)
- strand <+|-|both> (count tags on indicated strand, default: +)
- gene <data file> ... (Adds additional data to result based on the closest gene. This is useful for adding gene expression data. The file must have a header, and the first column must be a GeneID, Accession number, etc. If the peak cannot be mapped to data in the file then the entry will be left empty.)
- log (output tag counts as randomized log2 values - for scatter plots)
- sqrt (output tag counts as randomized sqrt values - for scatter plots)
- start <#> (start counting tags relative # offset of beginning of gene)
- end <#> (finish counting tags relative # offset to end of the gene)
- pausing <#> (calculate ratio of pausing first [# bp of transcript] to gene body) Produces 3 columns - promoter rpk, body rpk, and ratio (add -log for log versions) Also sets "-count genes". Use "-strand both" when analyzing Pol II ChIP-Seq

rpk is reads per kb - set -norm 1e6 or -normMatrix 1e6 to get rpkm



Can't figure something out? Questions, comments, concerns, or other feedback:
cbenner@ucsd.edu



HOMER

Software for motif discovery and next-gen sequencing analysis

homerTools - General sequence manipulation

homerTools is a utility program Chuck uses for basic sequence manipulation of FASTQ files, extracting sequences from genome FASTA files, and calculating nucleotide frequencies. To run **homerTools** do the following:

homerTools [command] [command specific options]

i.e. **homerTools trim -3 AAAAAAAA s_1_sequence.txt**

The following commands are available in **homerTools**:

barcodes - for separating and removing 5' barcodes from FASTQ/FASTA files

trim - for trimming by adapter sequence, specific lengths, etc. from FASTQ/FASTA files

freq - for calculating nucleotide frequencies in FASTQ/FASTA/txt sequence files

extract - for extracting specific regions of sequence from genomic FASTA files

Separating 5' Barcodes:

To separate and remove 5' barcodes from sequencing data (where the first "x" base pairs of the read are the barcode):

homerTools barcodes <# length of barcode> [options] <sequence file1> [sequence file2] ...

i.e. **homerTools barcodes 3 s_1_sequence.txt** (removes first 3bp as the barcode and sorts the reads by barcode)

The 3rd argument must be the length of the 5' barcode, which will be the first base pairs in the sequence. By default, this command creates files named "filename.barcode", such as s_1_sequence.txt.AAA, s_1_sequence.txt.AAC, s_1_sequence.txt.AAG etc. The parameter "**-min <#>**" specifies the minimum barcode frequency to keep (default is 0.02 [2%]). The frequency of each barcode is recorded in the output file "filename.freq.txt". If important barcodes were deleted, rerun the command with a smaller value for "**-min <#>**".

Trimming Sequence Files

With all the fancy types of sequencing being done, it is getting common to find adapters as part of the sequences that are analyzed. The trim command allows users to trim sequences from the 3' and 5' ends by either a specific number of nucleotides or remove a specific adapter sequence. The basic command is executed like this:

homerTools trim [options] <sequence file1> [sequence file2]

The output will be placed in files "filename.trimmed" and the distribution of sequence lengths after trimming will be in "filename.lengths" for each of the input files. The following options control how homerTools trims the sequences:

- len <#>** (trim sequences to this length)
- min <#>** (remove sequence that are shorter than this after trimming)
- 3 <#>** (trim this many bp off the 3' end of the sequence)
- 5 <#>** (trim this many bp off the 5' end of the sequence)
- 3 <ACGT>** (trim adapter sequence (i.e. "-3 GGAGGATTT") from the 3' end of the sequence)
- 5 <ACGT>** (trim adapter sequence (i.e. "-5 GGAGGATTT") from the 5' end of the sequence)

For adapter sequence trimming, it will search for the first full match to the sequence and delete the rest of the sequence. For example if you specify "-3 AA", it will search for the first instance of "AA" and delete everything after it. It will also delete partial matches if they are at the end of the sequence (or beginning for 5'). As another example, our lab uses an amplification strategy for RNA that results in the ligation of a polyA tail to the RNA sequence. If the reads are long enough, the read will be just As.

i.e. GAGATTATCTACGTACCGAAAAAAAAAAAAAAAAAAAAA

Trimming with "-3 AAAAAAAAAA" will cleave the complete polyA stretch.

In this example: GAGATTATCTACGTACCGTACTGCATGACGGGAAAA, only the final 4 As would be trimmed.

Extracting Genomic Sequences From FASTA Files

The **extract** command can be used to extract large numbers of specific genomic sequence. The first input file you need is a HOMER style peak file or a BED file with genomic locations. Next, you must have the genomic DNA sequences in one of two formats: (1) a directory of chr1.fa, chr2.fa FASTA files (can be masked file like *.fa.masked), or (2) a single file FASTA file with all of the chromosomes concatenated in one file. The sequences are sent to *stdout* as a tab-delimited file, or as a FASTA formatted file if "-fa" is added to the end of the command. Save the output to a file by adding "> outputfile.txt" to the end of the command. The program is run like this:

homerTools extract <peak/BED file> <FASTA directory or file location> [-fa]

```
i.e. homerTools extract peaks.bed
/home/chucknorris/homer/data/genomes/mm9/ > outputSequences.txt
Or, to get FASTA files back, i.e. homerTools extract peaks.bed
/home/chucknorris/homer/data/genomes/mm9/ -fa >
outputSequence.fa
```

Calculating Nucleotide Frequencies

The **freq** command will calculate nucleotide frequencies from FASTQ, FASTA, or tab-delimited text sequence files. The program tries to auto detect the format, but it may help to specify the format directly ("**-format fastq**", "**-format fasta**", "**-format tsv**"). The program outputs a position-dependent nucleotide/dinucleotide frequency file as a function of the distance from the start of the sequencing reads. The output is sent to *stdout*, unless you specify "**-o <outputfile.txt>**". If you specify "**-gc <outputfile2.txt>**", the program will also create a file that specifies the cumulative frequency of CpG, total G+C, total A+G, and total A+C in each individual sequence.

```
homerTools freq -format fastq s_1_sequence.txt > s_1.frequency.txt
homerTools freq -format fastq s_1_sequence.txt -gc
GCdistribution.txt -o positionFrequency.txt
```

homerTools Command Line options:

Usage: homerTools <command> [--help | options]

Collection of tools for sequence manipulation

Commands: [type "homerTools <command>" to see individual command options]

- barcodes - separate FASTQ file by barcodes
- trim - trim adapter sequences or fixed sizes from FASTQ files(also splits)
- freq - calculate position-dependent nucleotide/dinucleotide frequencies
- extract - extract specific sequences from FASTA file(s)
- decontaminate - remove bad tags from a contaminated tag directory
- cluster - hierarchical clustering of a NxN distance matrix
- special - specialized routines (i.e. only really useful for chuck)

Options for command: barcode

- min <#> (Minimum frequency of barcodes to keep: default=0.020)
- freq <filename> (output file for barcode frequencies, default=file.freq.txt)
- qual <#> (Minimum quality score for barcode nucleotides, default=not used)
- qualBase <character> (Minimum quality character in FASTQ file, default=B)

Options for command: trim

- 3 <#>|[ACGT]> (trim # bp or adapter sequence from 3' end of sequences)
- 5 <#>|[ACGT]> (trim # bp or adapter sequence from 5' end of sequences)
- mis <#> (Maximum allowed mismatches in adapter sequence, default: 0)
- minMatchLength <#> (minimum adapter sequence at edge to match, default: half adapter length)
- len <#> (Keep first # bp of sequence - i.e. make them the same length)
- stats <filename> (Output trimming statistics to filename, default: sent to stdout)
- min <#> (Minimum size of trimmed sequence to keep, default: 1)
- max <#> (Maximum read length, default: 100000)
- suffix <filename suffix> (output is sent to InuptFileName.suffix, default: trimmed)
- lenSuffix <filename suffix> (length distribution is sent to InuptFileName.suffix, default: lengths)
- split <#> (Split reads into two reads at bp #, output to trimmed1 and trimmed2)
- revopp <#> (Return reverse opposite of read [if used with -split, only the 2nd half of the read will be retuned as reverse opposite])

Options for command: freq

- format <tsv|fasta|fastq> (sequence file format, default: auto detect)
- offset <#> (offset of first base in output file, default: 0)
- maxlen <#> (Maximum length of sequences to consider, default: length of 1st seq)
- o <filename> (Output filename, default: output sent to stdout)
- gc <filename> (calculate CpG/GC content per sequence output to "filename")

OutputFormat: name<tab>CpG<tab>GC<tab>AG<tab>AC<tab>Length

Options for command: extract

- fa (output sequences in FASTA format - default is tab-delimited format)

Alternate Usage: homerTools extract stats <Directory of FASTA files>

Displays stats about the genome files (such as length)

Options for command: decontaminate

- frac <#> (Estimate fraction of sample that is contaminated, default: auto)
- estimateOnly (Only estimate the contamination, do not decontaminate)
- o <output tag directory> (default: overrides contaminated tag directory)
- size <#> (Peak size for estimating contamination/Max distance from contaminant reads to remove contaminated reads, default: 250)
- min <#> (Minimum tag count to consider when estimating contamination, default: 20)



Can't figure something out? Questions, comments, concerns, or other feedback:
cbenner@ucsd.edu



HOMER

Software for motif discovery and next-gen sequencing analysis

Miscellaneous Tools for Sharing Data

HOMER contains several utility programs for changing file formats and performing tasks that you may find useful from time to time. Below are a list of programs that may come in handy.

Outputing Tag Directory as a BED file (tagDir2bed.pl):

This is useful when you want to share your sequencing with others as BED format is about the most general format there is out there.

```
tagDir2bed.pl <tag directory> > output.alignment.file.bed
```

```
i.e. tagDir2bed.pl Macrophage-PU.1-ChipSeq/ > mac.pu1.bed
```

This will produce a large BED file that can be used to import the data to other programs.

Covertng between HOMER peak and BED file formats (pos2bed.pl / bed2pos.pl):

Want to load HOMER peaks into the genome browser? Or use them with other software?

Covert a HOMER peak/position file to a BED file:

```
pos2bed.pl <peak file> > output.bed
```

Covert a BED file to a HOMER peak/position file:

```
bed2pos.pl <BED file> > output.peakfile.txt
```



Can't figure something out? Questions, comments, concerns, or other feedback:
cbenner@ucsd.edu



HOMER

Software for motif discovery and ChIP-Seq analysis

Finding Overlapping and Differentially Bound Peaks

HOMER provides a utility for comparing sets of peaks called **mergePeaks**. It's default behavior is to take two or more peak files and return a single peak file containing the unique peak positions from the original files. For example:

```
mergePeaks -d <maximum distance to merge> <peak file1> <peak file2>  
[peak file3] ... > newPeakFile.txt
```

The program will output a new peak file containing the merged peaks to *stdout*. Peaks within the distance in bp specified by "**-d <#>**" will be reported as the average position between the peaks found within the common region (default=100 bp, good for transcription factors). The origin of the peaks is specified in the 7th column of the new peak file. Alternatively you can specify "**-d given**" to require a specific overlap between the start and end coordinates of the peaks. This is more useful if comparing large regions as opposed to peaks. The program will also output the numbers for creating a venn diagram, and these can be directed to a specific file by specifying "**-venn <filename>**".

Separating Peaks into Unique and Overlapping sets

Merging peaks together into a single file is very useful for certain types of analysis, such as making scatter plots that compare the tag-densities between peaks from separate experiments - in this case you want to count tags at specific and common regions. Alternatively, you may be interested in separating the peaks into common and specific sets for focused analysis. To do this use the "**-prefix <filename>**" option - this will create separate files based on overlapping peaks for each set of peaks. For example:

```
mergePeaks -d 100 pu1.peaks cebp.peaks -prefix mmm
```

This will create files named "**mmm_pu1.peaks**", "**mmm_cebp.peaks**", and "**mmm_pu1.peaks_cebp.peaks**".

The output file will contain the following columns:

1. Merged Peak name (will start with "Merged-")
2. chromosome
3. start (average from merged peaks)
4. end (average from merged peaks)

5. strand
6. Average peak score (actually, the average of the original values in column 6 of the peak files - or column 5 of BED files)
7. Original peak files contributing to the merged peak
8. Total number of peaks merged (occasionally more than one peak from a single file will be merged if the peaks are within the specify distance or two or more peaks from one file overlap with the same single peak(s) from another file)

Peak Co-Occurrence Statistics

The **mergePeaks** program will also find calculate the statistics of co-occurrence between peaks in a pairwise fashion. If "**-matrix <filename>**" is specified, HOMER will calculate statistics about the pairwise overlap of peaks. Three separate pairwise matrix files will be produced using the supplied <filename> as a prefix:

filename.logPvalue.matrix.txt (natural log p-values for overlap using the hypergeometric distribution, positive values signify divergence)

filename.logRatio.matrix.txt (natural log of the ratio of observed overlapping peaks to the expected number of overlapping peaks)

filename.count.matrix.txt (raw counts of overlapping peaks)

The statistics are dependent on the effective size of the genome, which can be specified using "**-gsize <#>**" (default: 2,000,000,000)

Co-Bound Peaks

Sometimes you just want to know how many other peaks bind a set of reference peaks. If "**-cobound <#>**", mergePeaks counts how many of the other peak files contain overlapping peaks with the peaks found in the first peak file. It then outputs peak files named "coboundBy0.txt", "coboundBy1.txt", etc. up to the number specified.

Differentially Bound Peaks

To find peaks that are differentially enriched between two experiments, there are two basic options. First, you could run **findPeaks** ([info here](#)) using the 2nd experiment as the control sample. Alternatively, you can use **getDifferentialPeaks**, which will take a given list of peaks and quickly identify which peaks contain significantly more tags in the target experiment relative to the background experiment. To use it, follow this syntax:

**getDifferentialPeaks <peak/BED file> <target Tag directory>
<background Tag directory> [options]**

By default it looks for peaks that have 4-fold more tags (sequencing-depth

independent) and a cumulative Poisson p-value less than 0.0001 (sequencing-depth dependent). These parameters are adjustable with ("**-F <#>**", and "**-P <#>**"). By specifying "**-same**", peaks that are similar between the two tag directories will be returned instead of differential peaks. One caveat is that it is a good idea to set the size of the region used to search for reads to be larger than the actual peaks (i.e. +100 bp relative to the peak size) to avoid problems that arise from experiments with different fragment lengths, etc.

Command Line options for mergePeaks

Usage: mergePeaks [options] <primary peak file> [additional peak/annotation files...]

Merges and/or compares peak/position files (peak files listed twice are only considered once)

General Options:

- strand (Only merge/consider peaks on the same strand, default: either strand)
- d <#|given> (Maximum distance between peak centers to merge, default: 100)
 - Using "-d given" looks for literal overlaps in peak regions
 - Use "-d given" when features have vastly different sizes (i.e. peaks vs. introns)
- file <filename> (file listing peak files to compare - for lots of peak files)
- gsize <#> (Genome size for significance calculations, default: 2e9)

Merging Peaks Options (default):

- prefix <filename> (Generates separate files for overlapping and unique peaks)
 - By default all peaks are sent to stdout
- matrix <filename> (Generates files with pairwise comparison statistics)
 - filename.logPvalue.matrix.txt - ln p-values for overlap, +values for divergence
 - filename.logRatio.matrix.txt - ln ratio of observed/expected overlaps
 - filename.count.matrix.txt - peak overlap counts
- venn <filename> (output venn diagram numbers to file, default: to stderr)
- code (report peak membership as binary instead of by file names)

Classify peaks by how many are co-bound by other peak files vs. reference(1st file)

- cobound <#> (Maximum number of co-bound peaks to consider)
 - Will output sets of peaks that are co-bound by various numbers of factors
 - to files coBoundBy0.txt, coBoundBy1.txt, coboundBy2.txt, ...
 - Or <prefix>.coBoundBy0.txt, <prefix>.coBoundBy1.txt, ...
- matrix <filename> (generates similar files to above with pairwise overlap statistics)

Single peak file:

(If a single peak file is given, peaks within the maximum distance will be merged)
-filter chrN:XXX-YYY (only analyze peaks within range)
-coverage <output file> (returns the total bp covered by each peak file - use "-d given")

Command Line options for getDifferentialPeaks

Usage: getDifferentialPeaks <peak file> <target tag directory> <background tag directory> [options]

Extracts tags near each peak from the tag directories and counts them, outputting peaks with significantly different tag densities

General Options:

-F <#> (fold enrichment over background tag count, default: 4.0)
-P <#> (poisson enrichment p-value over background tag count, default: 0.0001)
-same (return similar peaks instead of different peaks)
-rev (return peaks with higher tag counts in background instead of target library)
-size <#> (size of region around peak to count tags, default: -fixed)
-fixed (Count tags relative to actual peak start and stop, default)

Output Options:

-strand <both|+|-> (Strand [relative to peak] to count tags from, default: both)
-tagAdjust <#> (bp to shift tag positions to estimate fragment centers, default: auto)
'-tagAdjust auto' uses half of the estimated tag fragment length
-tagAdjustBg <#> (bp to shift background tag positions to estimate fragment centers, default: auto)
'-tagAdjustBg auto' uses half of the estimated tag fragment length
-tbp <#> (Maximum tags per bp to count, 0 = no limit, default: 0)
-tbpBg <#> (Maximum background tags per bp to count, 0 = no limit, default: 0)



Can't figure something out? Questions, comments, concerns, or other feedback:
cbenner@ucsd.edu



HOMER

Software for motif discovery and next-gen sequencing analysis

ChIP-Seq Analysis: Analyzing a ChIP-Seq experiment with one command

Even I don't like typing the same commands over and over again. The following command performs the standard set of analysis commands so that you can do better things while your data is processed.

analyzeChIP-Seq.pl <Tag Directory> <genome> [general options] [-A | B | C | D sub-program options]

i.e. a common use: **analyzeChIP-Seq.pl** Factor-ChIP-Seq/ hg18r -i Input-ChIP-Seq -focus -A factor_alignment_file.bed factor_alignment_file2.bed

This command performs 4 separate tasks labeled as A,B,C & D:

- A. Runs **makeTagDirectory** to parse alignment files, set up the tag directory, and performs basic QC such as tag auto correlation and checks for sequence bias.
- B. Runs **makeUCSCfile** and **findPeaks** to generate UCSC Genome Browser files and peak files for the experiment.
- C. Runs **findMotifsGenome.pl** to determine enriched motifs in your ChIP-Seq peaks.
- D. Runs **annotatePeaks.pl** to generate an annotated peak file and performs GO analysis on genes found near the peaks.

As output, this program will create standard files in the "Tag Directory" including an "index.html" file that links you to each of the output files.

There are a couple of general options that can be used with **analyzeChIP-Seq.pl**:

- i <input tag directory> : "Tag directory" to use as a control for peak finding.
- size <#> : Force this peak size for analysis (default is "auto" for peak finding, 200 bp for motif analysis and 50 bp for focused peak analysis)
- focus : This will find enriched motifs in 85% focused peaks using only +/- 25 bp of sequence, useful for identifying the primary motif bound by the factor.
- enhancer : This will set the peak size to "-size 1000" and only perform motif analysis on peaks > 3kb from the TSS.

To specifically tailor the options used by the sub-programs, first enter the "general options" you want above, then enter "-A" followed by the options you want passed to the sub-programs. For example, the most used sub-program option I use is the

following:

"-A s_1_eland_result.txt" - this passes the alignment file to the **makeTagDirectory** program so that it will be used to make the Tag Directory.

If you've already made a tag directory, no need to use the "-A blah blah" option. In similar fashion, to tell the motif finding to check motifs of length 10, 11, and 12, add "**C -len 10,11,12**" to the end of the command.

[Back to ChIP-Seq Analysis](#)



Can't figure something out? Questions, comments, concerns, or other feedback:
cbenner@ucsd.edu



HOMER

Software for motif discovery and ChIP-Seq analysis

File Formats

List of files used by HOMER - might be helpful when encountering problems.

[Another good resource on file formats: UCSC Genome Browser File Formats](#)

Peak/Positions files

These files specify genomic locations similar to BED files. They are tab-delimited text files with a minimum of 5 columns (additional columns are ignored). They are 1-indexed and inclusive, meaning the first nucleotide of a chromosome is referenced as position 1. They are inclusive in the sense that a line with a start of 100 and end of 200 indicates of region of size 101. Columns are as followed:

1. peak name (should be unique)
2. chromosome
3. starting position [integer] (1-indexed)
4. end position [integer]
5. strand [either 0/1 or +/-] (in HOMER strand of 0 is +, 1 is -)
6. Optional/ignored ...

Peak/Position files are very similar to BED files - to convert them use **pos2bed.pl** or **bed2pos.pl**.

BED files

These are essentially the same as Peak/Position files, except that they have a stricter [definition](#) but greater portability. They are also tab-delimited text files - the important difference is that they are 0-indexed, meaning the first nucleotide of the chromosome is referenced as position 0.

1. chromosome
2. starting position [integer] (0-indexed)
3. ending position [integer]
4. peak name
5. value (usually ignored)
6. strand [+/-]

BED files also come in a short form:

1. chromosome

2. starting position [integer] (0-indexed)
3. ending position [integer]
4. strand [+/-]

Peak/Position files are very similar to BED files - to convert them use **pos2bed.pl** or **bed2pos.pl**.

Motif files

These are files for specifying motifs, and are created by HOMER during motif discovery. They are tab-delimited text files. A more elaborate description of the format and how to tinker with it is [here](#). Basically, each motif within the file contains a header row starting with a ">", followed by several rows with 4 columns, specifying the probabilities of each nucleotide at each position.

```
>ASTTCCTCTT 1-ASTTCCTCTT 8.059752 -23791.535714 0
T:17311.0(44 ...
0.726 0.002 0.170 0.103
0.002 0.494 0.354 0.151
0.016 0.017 0.014 0.954
0.005 0.006 0.027 0.963
0.002 0.995 0.002 0.002
0.002 0.989 0.008 0.002
0.004 0.311 0.148 0.538
0.002 0.757 0.233 0.009
0.276 0.153 0.030 0.542
0.189 0.214 0.055 0.543
```

The first row starts with a ">" followed by various information, and the other rows are the positions specific probabilities for each nucleotide (A/C/G/T). These values do not need to be between 0-1. HOMER will automatically normalize whatever values are there, so interger counts are ok. The header row is actually TAB delimited, and contains the following information:

1. ">" + Consensus sequence (not actually used for anything, can be blank) example: >ASTTCCTCTT
2. Motif name (should be unique if several motifs are in the same file) example: 1-ASTTCCTCTT or NFkB
3. Log odds detection threshold, used to determine bound vs. unbound sites (**mandatory**) example: 8.059752
4. (optional) log P-value of enrichment, example: -23791.535714
5. (optional) 0 (A place holder for backward compatibility, used to describe "gapped" motifs in old version, turns out it wasn't very useful :)
6. (optional) Occurrence Information separated by commas, example:
T:17311.0(44.36%),B:2181.5(5.80%),P:1e-10317
 1. T: # (%) - number of target sequences with motif, % of total of total targets
 2. B: # (%) - number of background sequences with motif, % of total background
 3. P: # - final enrichment p-value
7. (optional) Motif statistics separated by commas, example:

Tpos:100.7,Tstd:32.6,Bpos:100.1,Bstd:64.6,StrandBias:0.0,Multiplicity:1.13

1. Tpos: average position of motif in target sequences (0 = start of sequences)
2. Tstd: standard deviation of position in target sequences
3. Bpos: average position of motif in background sequences (0 = start of sequences)
4. Bstd: standard deviation of position in background sequences
5. StrandBias: log ratio of + strand occurrences to - strand occurrences.
6. Multiplicity: The average number of occurrences per sequence in sequences with 1 or more binding site.

Only the first 3 columns are needed. In fact, the rest of the columns are really just statistics from motif finding and aren't important when searching for instances of a motif.

The MOST IMPORTANT value is the 3rd column - this sets the detection threshold, which specifies whether a given sequence is enough of a "match" to be considered recognized by the motif. More on that below.

Internal File Formats:

These are files that you normally won't modify or play with, but in case your interested...

***.tags.tsv files**

These are files used to store sequencing data in HOMER tag directories. They are tab-delimited text files that are sorted to allow for relatively quick access and processing.

1. blank (can be used for a name)
2. chromosome
3. position (1-indexed)
4. strand (0 or 1, +/- not allowed here)
5. Number of reads (can be fractional)
6. length of the read (optional)



Can't figure something out? Questions, comments, concerns, or other feedback:
cbenner@ucsd.edu



HOMER

Software for motif discovery and ChIP-Seq analysis

This is the old version of the documentation: [New Version](#)

ChIP-Seq Analysis

ChIP-Seq is the best thing that happened to ChIP since the antibody. It is 100x better than ChIP-Chip since it escapes most of the problems of microarray probe hybridization. Plus it is cheaper, and genome wide. Chuck would be pleased if he came up with the idea.

HOMER offers solid tools and methods for interpreting ChIP-Seq experiments. In addition to UCSC visualization support and peak finding [and motif finding of course], HOMER can help assemble data across multiple experiments and look at positional specific relationships between sequencing tags, motifs, and other features. You do not need to use the peak finding methods in this package to use motif finding. (Use the bed2pos.pl program to create peak files from BED files).

NOTE: The current implementation is geared for single tag sequencing.

Background:

- [Introduction to ChIP-Seq](#)
- [Aligning ChIP-Seq tags](#)

Standard ChIP-Seq analysis with HOMER:

1. [Creating a "Tag Directory" from aligned sequences](#)
2. [Basic quality control \(sequence bias, fragment length estimation\)](#)
3. [Creating files to view your data in the UCSC Genome Browser](#)
4. [Finding Peaks \(ChIP-enriched regions\) in the genome](#)
5. [Finding enriched motifs in ChIP-Seq peaks](#)
6. [Annotating Peaks \(and cross referencing other experiments and motifs\)](#)

[Automating standard ChIP-Seq analysis with analyzeChIP-Seq.pl](#)

Advanced ChIP-Seq Analysis with HOMER:

- [Finding overlapping or differentially bound peaks](#)
- [Creating histograms with sequencing data](#)
- [Creating heatmaps with sequencing data](#)
- [Re-centering peaks on motifs](#)



Can't figure something out? Questions, comments, concerns, or other feedback:
cbenner@ucsd.edu



HOMER

Software for motif discovery and next-gen sequencing analysis

HOMER Motif Analysis

HOMER contains a novel motif discovery algorithm that was designed for regulatory element analysis in genomics applications (DNA only, no protein). It is a differential motif discovery algorithm, which means that it takes two sets of sequences and tries to identify the regulatory elements that are specifically enriched in one set relative to the other. It uses ZOOPS scoring (zero or one occurrence per sequence) coupled with the hypergeometric enrichment calculations (or binomial) to determine motif enrichment. HOMER also tries its best to account for sequenced bias in the dataset. It was designed with ChIP-Seq and promoter analysis in mind, but can be applied to pretty much any nucleic acids motif finding problem.

There are several ways to perform motif analysis with HOMER. The links below introduce the various workflows for running motif analysis. In a nutshell, HOMER contains two tools, **findMotifs.pl** and **findMotifsGenome.pl**, that manage all the steps for discovering motifs in promoter and genomic regions, respectively. These scripts attempt to make it easy for the user to analyze a list of genes or genomic positions for enriched motifs. However, if you already have the sequence files that you want to analyze (i.e. FASTA files), **findMotifs.pl** (and **homer2**) can process these directly.

[Analyzing lists of genes with promoter motif analysis](#) (**findMotifs.pl**)

[Analyzing genomic positions](#) (**findMotifsGenome.pl**)

[Analyzing custom FASTA files](#) (**findMotifs.pl**, **homer2**)

[Analyzing data for RNA motifs](#) (**findMotifs.pl/findMotifsGenome.pl**)

[Tips for motif finding](#)

[Creating custom motif files](#)

Regardless of how you invoke HOMER, the same basic steps are executed to discover regulatory elements:

Preprocessing:

1. Extraction of Sequences (**findMotifs.pl/findMotifsGenome.pl**)

If genomic regions are provided as input, the appropriate genomic DNA is extracted. If gene accession numbers are provided, the appropriate promoter regions are selected.

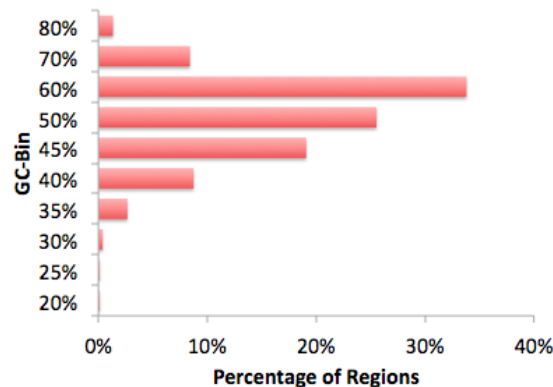
2. Background Selection (**findMotifs.pl/findMotifsGenome.pl**)

If the background sequences were not explicitly defined, HOMER will automatically select them for you. If you are using genomic positions, sequences will be randomly selected from the genome, matched for GC% content (to make GC normalization easier in the next step). If you are using promoter based analysis, all promoters (except those chosen for analysis) will be used as background. Custom backgrounds can be specified with "**-bg <file>**".

3. GC Normalization (**findMotifs.pl/findMotifsGenome.pl**)

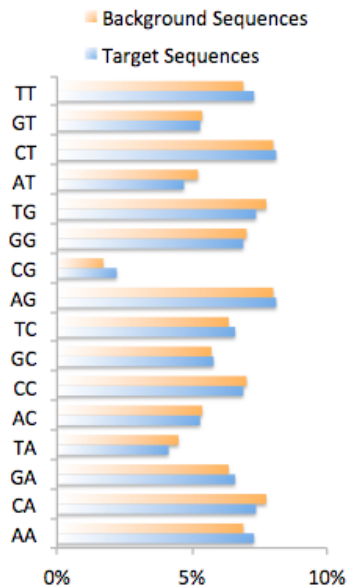
Sequences in the target and background sets are then binned based on their GC-content (5% intervals). Background sequences are weighted to resemble the same GC-content distribution observed in the target sequences. This helps avoid HOMER avoid simply finding motifs that are GC-rich when analyzing sequences from CpG Islands. To perform CpG% normalization instead of GC%(G+C) normalization, use "**-cpg**". An example of the GC%-distribution of regions from a

ChIP-Seq experiment:



4. Autonormalization (New with v3.0, homer2/findMotifs.pl/findMotifsGenome.pl)

Often the target sequences have an imbalance in the sequence content other than GC%. This can be caused by biological phenomenon, such as codon-bias in exons, or experimental bias caused by preferential sequencing of A-rich stretches etc. If these sources of bias are strong enough, HOMER will lock on to them as features that significantly differentiate the target and background sequences. HOMER now offers autonormalization as a technique to remove (or partially remove) imbalances in short oligo sequences (i.e. AA) by assigning weights to background sequences. The procedure attempts to minimize the difference in short oligo frequency (summed over all oligos) between target and background data sets. It calculates the desired weights for each background sequence to help minimize the error. Due to the complexity of the problem, HOMER uses a simple hill-climbing approach by making small adjustment in background weight at a time. It also penalizes large changes in background weight to avoid trivial solutions that a increase or decrease the weights of outlier sequences to extreme values. The length of short oligos is controlled by the "-nlen <#>" option.



Discovering Motifs *de novo* (homer2)

By default, HOMER uses the new homer2 version of the program for motif finding. If you wish to use the old version when running any of the HOMER family of programs, add "-homer1" to the command line.

5. Parsing input sequences into an Oligo Table

Input sequences parsed in to oligos of desired motif length, and read into an Oligo Table. The Oligo Table hold each unique oligo in the data set, remembering how many times it occurs in the target and background sequences. This is done to make searching for motif (which are essentially collections of oligos) much more efficient. However, this also destroys the relationship between individual oligos and their sequence of origin.

6. Oligo Autonormalization (optional)

While the Autonormalization described in step 4 above is applied to full sequences (i.e. ~200 bp), you can also apply the autonormalization concept to the Oligo Table. The idea is still to equalize the smaller oligos (i.e. 1,2,3 bp) within the larger motif lengthed oligos (i.e. 10,12,14 bp etc.). This is a little more dangerous since the total number of motif lengthed oligos can be very large (i.e. 500k for 10 bp, much more for longer motifs), meaning there are a lot of weights to "adjust". However, this can help if there is an extreme sequence bias that you might be having trouble scrubbing out of the data set (the **"-olen <#>"** option).

7. Global Search phase

After creating (and possibly normalizing) the Oligo Table, HOMER conducts a global search for enriched "oligos". The basic idea is that if a "Motif" is going to be enriched, then the oligos considered part of the motif should also be enriched. First, HOMER screens each possible oligo for enrichment. To increase sensitivity, HOMER then allows mismatches in the oligo when searching for enrichment. To speed up this process, which can be very resource consuming for longer oligos with a large number of possible mismatches, HOMER will skip oligos when allowing multiple mismatches if they were not promising, for example if they had more background instances than target instances, or if allowing more mismatches results in a lower enrichment value. The **"-mis <#>"** controls how many mismatches will be allowed.

Calculating Motif Enrichment:

Motif enrichment is calculated using either the cumulative hypergeometric or cumulative binomial distributions. These two statistics assume that the classification of input sequences (i.e. target vs. background) is independent of the occurrence of motifs within them. The statistics consider the total number of target sequences, background sequences and how many of each type contains the motif that is being checked for enrichment. From these numbers we can calculate the probability of observing the given number (or more) of target sequences with the motif by chance if we assume there is no relationship between the target sequences and the motif. The hypergeometric and binomial distributions are similar, except that the hypergeometric assumes sampling without replacement, while the binomial assumes sampling with replacement. The motif enrichment problem is more accurately described by the hypergeometric, however, the binomial has advantages. The difference between them is usually minor if there are a large number of sequences and the background sequences >> target sequences. In these cases, the binomial is preferred since it is faster to calculate. As a result it is the default statistic for **findMotifsGenome.pl** where the number of sequences is typically higher. However, if you use your own background that has a limited number of sequences, it might be a good idea to switch to the hypergeometric (use **"-h"** to force use of the hypergeometric). **findMotifs.pl** expects smaller number for promoter analysis and uses the hypergeometric by default.

One important note: Since HOMER uses an Oligo Table for much of the internal calculations of motif enrichment, where it does not explicitly know how many of the original sequences contain the motif, it approximates this number using the total number of observed motif occurrences in background and target sequences. It assumes the occurrences were equally distributed among the target or background sequences with replacement, were some of the sequences are likely to have more than one occurrence. It uses the expected number sequences to calculate the enrichment statistic (the final output reflects the actual enrichment based on the original sequences).

8. Matrix Optimization

HOMER takes the most enriched oligos from the global optimization step, transforms them into simple position specific probability matrices, and further optimizes them with a sensitive local optimization algorithm. This step is performed separately for each oligo, and will create the "motif probability matrix" as well as determine the optimal detection threshold to maximize the enrichment of the motif in the target vs. background sequences. The detection threshold is simply done by scoring each oligo in the data to the probability matrix, and then sorting the oligos by their similarity to the matrix. HOMER then steps down the list, effectively decreasing the detection threshold, including more and more oligos until an optimal enrichment is found. After this step, HOMER will create several new probability matrices based on the oligos found in different detection thresholds and check which one has the highest enrichment. This process is repeated until the enrichment can no longer be improved, producing a final motif.

9. Mask and Repeat

After the first "promising oligo" is optimized into a motif, the sequences bound by the motif to are removed from the analysis and the next promising oligo is optimized for the 2nd motif, and so on. This is repeated until the desired number of motifs are found ("**-S <#>**", default: 25). This is where there is an important difference between the old (homer) and new (homer2) versions. The old version of homer would simply mask the oligos bound by the motif from the Oligo Table. For example if the motif was GAGGAW then GAGGAA and GAGGAT would be removed from the Oligo Table to avoid having the next motif find the same sequences. However, if GAGGAW was enriched in the data, there is a good chance that any 6-mer oligo like nGAGGA or AGGAWn would also be somewhat enriched in the data. This would cause homer to find multiple versions of the same motif and provide a little bit of confusion in the results.

To avoid this problem in the new version of HOMER (homer2), once a motif is optimized, HOMER revisits the original sequences and masks out the oligos making up the instance of the motif as well as oligos immediately adjacent to the site that overlap with at least one nucleotide. This helps provide much cleaner results, and allows greater sensitivity when co-enriched motifs. To make revert back to the old way of motif masking with homer2, specify "**-quickMask**" at the command line. You can also run the old version with "**-homer1**".

Screening for Enrichment of Known Motifs (homer2):

10. Load Motif Library

In order to search for Known Motifs in your data, HOMER loads a list of previously determined motifs from previous data. You can also add your own motifs by specifying them at the command line ("**-mknown <file>**") or by editing the primary file ("data/knownTFs/known.motifs"). HOMER doesn't screen all of TRANSFAC - partially due to motif quality (which can be low), and partially due to the fact that we need a detection threshold.

11. Screen Each Motif

To find the enrichment for each motif, HOMER scans each sequence for instances of the motif and calculates the final enrichment by considering how many target vs. background sequences are considered "bound". ZOOPS (zero or one occurrence per sequence) counting is used and the hypergeometric or binomial is used to calculate the significance.

Motif Analysis Output:

12. Motif Files (homer2, findMotifs.pl, findMotifsGenome.pl)

The true output of HOMER are "*.motif" files which contain the information necessary to identify future instance of motifs. They are reported in the output directories from findMotifs.pl and findMotifsGenome.pl. A typical motif file will look something like:

```
>ASTTCCTCTT 1-ASTTCCTCTT 8.059752 -23791.535714 0 T:17311.0(44 ...
```


0.726	0.002	0.170	0.103
0.002	0.494	0.354	0.151
0.016	0.017	0.014	0.954
0.005	0.006	0.027	0.963
0.002	0.995	0.002	0.002
0.002	0.989	0.008	0.002
0.004	0.311	0.148	0.538
0.002	0.757	0.233	0.009
0.276	0.153	0.030	0.542
0.189	0.214	0.055	0.543

The first row starts with a ">" followed by various information, and the other rows are the positions specific probabilities for each nucleotide (A/C/G/T). The header row is actually TAB delimited, and contains the following information:

1. ">" + Consensus sequence (not actually used for anything, can be blank) example:
>ASTTCCTCTT
2. Motif name (should be unique if several motifs are in the same file) example: 1-
ASTTCCTCTT or NFkB
3. Log odds detection threshold, used to determine bound vs. unbound sites (**mandatory**)
example: 8.059752
4. log P-value of enrichment, example: -23791.535714
5. 0 (A place holder for backward compatibility, used to describe "gapped" motifs in old version, turns out it wasn't very useful :)
6. Occurrence Information separated by commas, example:
T:17311.0(44.36%),B:2181.5(5.80%),P:1e-10317
 1. T: # (%) - number of target sequences with motif, % of total of total targets
 2. B: # (%) - number of background sequences with motif, % of total background
 3. P: # - final enrichment p-value
7. Motif statistics separated by commas, example:
Tpos:100.7,Tstd:32.6,Bpos:100.1,Bstd:64.6,StrandBias:0.0,Multiplicity:1.13
 1. Tpos: average position of motif in target sequences (0 = start of sequences)
 2. Tstd: standard deviation of position in target sequences
 3. Bpos: average position of motif in background sequences (0 = start of sequences)
 4. Bstd: standard deviation of position in background sequences
 5. StrandBias: log ratio of + strand occurrences to - strand occurrences.
 6. Multiplicity: The average number of occurrences per sequence in sequences with 1 or more binding site.

You can [easily create your own motif files](#), just remember that the **first 3 columns are required!!!**

13. De novo motif output (findMotifs.pl/findMotifsGenome.pl/compareMotifs.pl)

HOMER takes the motifs identified from *de novo* motif discovery step and tries to process and present them in a useful manner. An HTML page is created in the output directory named homerResults.html along with a directory named "homerResults/" that contains all of the image and other support files to create the page. These pages are explicitly created by running a subprogram called "**compareMotifs.pl**".

Comparison of Motif Matrices:

Motifs are first checked for redundancy to avoid presenting the same motifs over and over again. This is done by aligning each pair of motifs at each position (and their reverse opposites) and scoring their similarity to determine their best alignment. Starting with HOMER v3.3, matrices are compared using Pearson's correlation coefficient by converting each matrix into a vector of values. Neutral frequencies (0.25) are used in where the motif matrices do not overlap.

The old comparison was done by comparing the probability matrices using the formula below which manages the expectations of the calculations by scrambling the nucleotide identities as a

control. (freq1 and freq2 are the matrices for motif1 and motif2)

$$SimilarityScore = \frac{1}{MotifLength} \sum_i^{MotifLength} - \frac{(Observed_i - Expect_i)}{Expect_i}$$
$$Observed_i = \sum_j^{A,C,G,T} - (freq1_{ij} - freq2_{ij})^2$$
$$Expect_i = \sum_j^{A,C,G,T} \sum_k^{A,C,G,T} \frac{-(freq1_{ij} - freq2_{ik})^2}{4}$$

The output will be a score ranging from some lower bound (depending on the matrix frequencies) to 1, where 1 is complete similarity. By default the threshold for assigning similar motifs is 0.6, which is a reasonable cutoff in practice. This can be changed if you run **compareMotifs.pl** and change the **"-reduceThresh <#>"** parameter.

Motifs are next compared against a library of known motifs. For this step, all motifs in JASPAR and the "known" motifs are used for comparison. You can specify a custom motif library using **"-mcheck <motif library file>"** when using **findMotifs[Genome].pl** or **"-known <motif library file>"** when calling **compareMotifs.pl** directly.

By default, it looks for the file **"/path-to-homer/data/knownTFs/all.motifs"** to find the motif to compare with the de novo motifs. If **"-rna"** is specified, it will load the file **"/path-to-homer/data/knownTFs/all.rna.motifs"**.

An example of the output HTML is show below:

Homer de novo Motif Results

Known Motif Enrichment Results

Gene Ontology Enrichment Results

If Homer is having trouble matching a motif to a known motif, try copy/pasting the matrix file into [STAMP](#)

More information on motif finding results: [HOMER](#) | [Description of Results](#) | [Tips](#)

Total target sequences = 15212

Total background sequences = 34047

* - possible false positive

Rank	Motif	P-value	log P-value	% of Targets	% of Background	STD(Bg STD)	Best Match/Details	Motif File
1	GGAAATTCCTCC	1e-1835	-4.228e+03	28.11%	5.16%	37.7bp (63.1bp)	NFkB-p65(RHD)/GMI12787-p65-ChIP-Seq/Homer More Information Similar Motifs Found	motif file (matrix)
2	AAAGAGGAAGTG	1e-1716	-3.953e+03	34.50%	8.65%	47.8bp (62.6bp)	PB0058.1_Stp11_1 More Information Similar Motifs Found	motif file (matrix)
3	TTCCGCAATCA	1e-1585	-3.651e+03	28.85%	6.39%	41.8bp (62.8bp)	MA0102.1_Cebpa More Information Similar Motifs Found	motif file (matrix)
4	TGAGTTCAT	1e-1004	-2.314e+03	25.07%	7.22%	49.2bp (61.0bp)	NF-E2(bZIP)/K562-NFE2-ChIP-Seq/Homer More Information Similar Motifs Found	motif file (matrix)
5	TCAATATGCAAA	1e-262	-6.039e+02	5.52%	1.27%	47.8bp (59.7bp)	Oct2(POU/Homeobox)/Bcell-Oct2-ChIP-Seq/Homer More Information Similar Motifs Found	motif file (matrix)
6	ATGACGTCACTCT	1e-198	-4.565e+02	6.42%	2.09%	49.9bp (53.2bp)	c-Jun-CRE(bZIP)/K562-cJun-ChIP-Seq/Homer More Information Similar Motifs Found	motif file (matrix)
7	AAACCACA	1e-146	-3.370e+02	7.66%	3.30%	51.7bp (58.9bp)	RUNX1(Runt)/Jurkat-RUNX1-ChIP-Seq/Homer More Information Similar Motifs Found	motif file (matrix)

Depending on how the **findMotifs[Genome].pl** program that was executed, the "Known Motif Enrichment Results" and "Gene Ontology Enrichment Results" may or may not link to anything. Motifs are sorted based on p-value, and basic statistics about the motif (present in the motif files) is displayed.

The final column contains a link to the "motif file", which is important if you want to search for the motif in other sequences.

In the Best Match/Details column, HOMER will display the known motif which most closely


matched with the *de novo* motif. It is very important that you **TAKE THIS ASSIGNMENT WITH A GRAIN OF SALT!!!!** Unfortunately, sometimes the best match still isn't any good. Also, it is common that the "known" motif isn't any good to begin with. To investigate the assignment further, click on the "More Information" link which provides a page that looks like this:

Basic Information: The section contains basic information, including links to the motif file (normal and reverse opposite) and the pdf version of the motif logo.


Homer de novo Motif Results

motif1

Information for motif1



Reverse Opposite:



p-value:	1e-1835
log p-value:	-4.228e+03
Number of Target Sequences with motif	4276.0
Percentage of Target Sequences with motif	28.11%
Number of Background Sequences with motif	1756.8
Percentage of Background Sequences with motif	5.16%
Average Position of motif in Targets	99.3 +/- 37.7bp
Average Position of motif in Background	99.7 +/- 63.1bp
Strand Bias (log2 ratio + to - strand density)	-0.0
Multiplicity (# of sites on avg that occur together)	1.12
Motif File:	file (matrix) reverse opposite
PDF Format Logos:	forward logo reverse opposite

Followed by matches to known motifs. This section shows the alignments between the *de novo* motif and known motifs. It's important to check and see if these alignments look reasonable:

Matches to Known Motifs

NFkB-p65(RHD)/GM12787-p65-ChIP-Seq/Homer	
Match Rank: 1	
Score: 0.93	
Offset: -2	
Orientation: forward strand	
Alignment: --GGAATTYCCC NGGGGATTCCCC	
MA0061.1_NF-kappaB	
Match Rank: 2	
Score: 0.87	
Offset: -1	
Orientation: forward strand	
Alignment: -GGAATTYCCC GGGAATTCC-	
MA0105.1_NFKB1	
Match Rank: 3	
Score: 0.84	
Offset: -1	
Orientation: forward strand	
Alignment: -GGAATTYCCC GGGGATTCCCC	

Clicking on the "similar motifs" will show the other de novo motifs found during motif finding that resemble the motif but had a lower enrichment value. It contains a similar "header" as the "More Information" link, but below it shows the motifs that were considered similar. It is usually a good idea to check this list over - sometimes a distinct motif will be grouped incorrectly in the list because it shares a couple residues.

Similar de novo motifs found

Rank	Match Score	Redundant Motif	P-value	log P-value	% of Targets	% of Background	Motif file
1	0.918		1e-1776	-4089.766852	26.30%	4.60%	motif file (matrix)
2	0.873		1e-1711	-3941.421170	25.85%	4.62%	motif file (matrix)
3	0.844		1e-968	-2231.146991	25.56%	7.71%	motif file (matrix)
4	0.616		1e-259	-597.025749	12.81%	5.44%	motif file (matrix)
5	0.662		1e-233	-537.315538	13.40%	6.12%	motif file (matrix)
6	0.795		1e-222	-512.488031	22.69%	13.20%	motif file (matrix)
7	0.874		1e-148	-341.450152	20.88%	13.25%	motif file (matrix)







To rerun this part of the analysis on any arbitrary set of motifs, simply run the "**compareMotifs.pl**" command (use without any command line parameters to get the usage options).

14. Known motif output

Known motif enrichment is displayed as a HTML file (knownResults.html). The page sorts the results based on enrichment and displays basic information:

Homer Known Motif Enrichment Results

[Homer de novo Motif Results](#)
[Gene Ontology Enrichment Results](#)
[Known Motif Enrichment Results \(txt file\)](#)
Total Target Sequences = 15213, Total Background Sequences = 34081

Rank	Motif	Name	P-value	log P-value	# Target Sequences with Motif	% of Targets Sequences with Motif	# Background Sequences with Motif	% of Background Sequences with Motif	Motif File	PDF
1		NFkB-p65(RHD)/GM12787-p65-ChIP-Seq/Homer	1e-1707	-3.931e+03	3855.0	25.34%	1506.4	4.42%	motif file (matrix)	pdf
2		CEBP(bZIP)/CEBPb-ChIP-Seq/Homer	1e-1310	-3.018e+03	3423.0	22.50%	1551.0	4.55%	motif file (matrix)	pdf
3		PU.1(ETS)/ThioMac-PU.1-ChIP-Seq/Homer	1e-1288	-2.967e+03	3413.0	22.44%	1569.7	4.61%	motif file (matrix)	pdf
4		AP-1(bZIP)/ThioMac-PU.1-ChIP-Seq/Homer	1e-947	-2.183e+03	3251.0	21.37%	1900.8	5.58%	motif file (matrix)	pdf
5		NFkB-p65-Rel(RHD)/LPS-exp/Homer	1e-936	-2.157e+03	1163.0	7.65%	166.2	0.49%	motif file (matrix)	pdf
6		ETS1(ETS)/Jurkat-ETS1-ChIP-Seq/Homer	1e-885	-2.039e+03	4447.0	29.23%	3568.0	10.47%	motif file (matrix)	pdf



Can't figure something out? Questions, comments, concerns, or other feedback:
cbenner@ucsd.edu



HOMER

Software for motif discovery and next-gen sequencing analysis

HOMER Motif Analysis

HOMER contains a novel motif discovery algorithm that was designed for regulatory element analysis in genomics applications (DNA only, no protein). It is a differential motif discovery algorithm, which means that it takes two sets of sequences and tries to identify the regulatory elements that are specifically enriched in one set relative to the other. It uses ZOOPS scoring (zero or one occurrence per sequence) coupled with the hypergeometric enrichment calculations (or binomial) to determine motif enrichment. HOMER also tries its best to account for sequenced bias in the dataset. It was designed with ChIP-Seq and promoter analysis in mind, but can be applied to pretty much any nucleic acids motif finding problem.

There are several ways to perform motif analysis with HOMER. The links below introduce the various workflows for running motif analysis. In a nutshell, HOMER contains two tools, **findMotifs.pl** and **findMotifsGenome.pl**, that manage all the steps for discovering motifs in promoter and genomic regions, respectively. These scripts attempt to make it easy for the user to analyze a list of genes or genomic positions for enriched motifs. However, if you already have the sequence files that you want to analyze (i.e. FASTA files), **findMotifs.pl** (and **homer2**) can process these directly.

[Analyzing lists of genes with promoter motif analysis](#) (**findMotifs.pl**)

[Analyzing genomic positions](#) (**findMotifsGenome.pl**)

[Analyzing custom FASTA files](#) (**findMotifs.pl**, **homer2**)

[Analyzing data for RNA motifs](#) (**findMotifs.pl/findMotifsGenome.pl**)

[Tips for motif finding](#)

[Creating custom motif files](#)

Regardless of how you invoke HOMER, the same basic steps are executed to discover regulatory elements:

Preprocessing:

1. Extraction of Sequences (**findMotifs.pl/findMotifsGenome.pl**)

If genomic regions are provided as input, the appropriate genomic DNA is extracted. If gene accession numbers are provided, the appropriate promoter regions are selected.

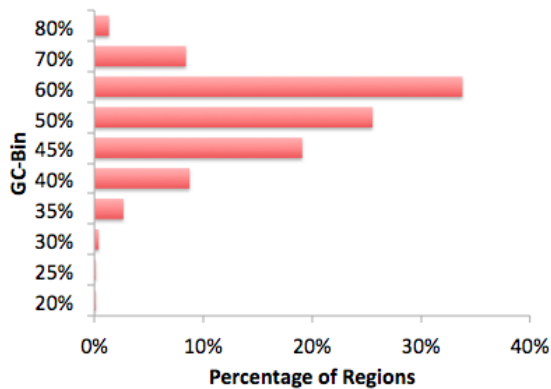
2. Background Selection (**findMotifs.pl/findMotifsGenome.pl**)

If the background sequences were not explicitly defined, HOMER will automatically select them for you. If you are using genomic positions, sequences will be randomly selected from the genome, matched for GC% content (to make GC normalization easier in the next step). If you are using promoter based analysis, all promoters (except those chosen for analysis) will be used as background. Custom backgrounds can be specified with "**-bg <file>**".

3. GC Normalization (**findMotifs.pl/findMotifsGenome.pl**)

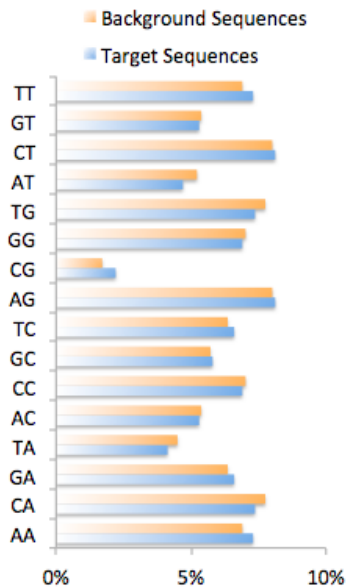
Sequences in the target and background sets are then binned based on their GC-content (5% intervals). Background sequences are weighted to resemble the same GC-content distribution observed in the target sequences. This helps avoid HOMER avoid simply finding motifs that are GC-rich when analyzing sequences from CpG Islands. To perform CpG% normalization instead of GC%(G+C) normalization, use "**-cpg**". An example of the GC%-distribution of regions from a

ChIP-Seq experiment:



4. Autonormalization (New with v3.0, homer2/findMotifs.pl/findMotifsGenome.pl)

Often the target sequences have an imbalance in the sequence content other than GC%. This can be caused by biological phenomenon, such as codon-bias in exons, or experimental bias caused by preferential sequencing of A-rich stretches etc. If these sources of bias are strong enough, HOMER will lock on to them as features that significantly differentiate the target and background sequences. HOMER now offers autonormalization as a technique to remove (or partially remove) imbalances in short oligo sequences (i.e. AA) by assigning weights to background sequences. The procedure attempts to minimize the difference in short oligo frequency (summed over all oligos) between target and background data sets. It calculates the desired weights for each background sequence to help minimize the error. Due to the complexity of the problem, HOMER uses a simple hill-climbing approach by making small adjustment in background weight at a time. It also penalizes large changes in background weight to avoid trivial solutions that a increase or decrease the weights of outlier sequences to extreme values. The length of short oligos is controlled by the "-nlen <#>" option.



Discovering Motifs *de novo* (homer2)

By default, HOMER uses the new homer2 version of the program for motif finding. If you wish to use the old version when running any of the HOMER family of programs, add "-homer1" to the command line.

5. Parsing input sequences into an Oligo Table

Input sequences parsed in to oligos of desired motif length, and read into an Oligo Table. The Oligo Table hold each unique oligo in the data set, remembering how many times it occurs in the target and background sequences. This is done to make searching for motif (which are essentially collections of oligos) much more efficient. However, this also destroys the relationship between individual oligos and their sequence of origin.

6. Oligo Autonormalization (optional)

While the Autonormalization described in step 4 above is applied to full sequences (i.e. ~200 bp), you can also apply the autonormalization concept to the Oligo Table. The idea is still to equalize the smaller oligos (i.e. 1,2,3 bp) within the larger motif lengthed oligos (i.e. 10,12,14 bp etc.). This is a little more dangerous since the total number of motif lengthed oligos can be very large (i.e. 500k for 10 bp, much more for longer motifs), meaning there are a lot of weights to "adjust". However, this can help if there is an extreme sequence bias that you might be having trouble scrubbing out of the data set (the **"-olen <#>"** option).

7. Global Search phase

After creating (and possibly normalizing) the Oligo Table, HOMER conducts a global search for enriched "oligos". The basic idea is that if a "Motif" is going to be enriched, then the oligos considered part of the motif should also be enriched. First, HOMER screens each possible oligo for enrichment. To increase sensitivity, HOMER then allows mismatches in the oligo when searching for enrichment. To speed up this process, which can be very resource consuming for longer oligos with a large number of possible mismatches, HOMER will skip oligos when allowing multiple mismatches if they were not promising, for example if they had more background instances than target instances, or if allowing more mismatches results in a lower enrichment value. The **"-mis <#>"** controls how many mismatches will be allowed.

Calculating Motif Enrichment:

Motif enrichment is calculated using either the cumulative hypergeometric or cumulative binomial distributions. These two statistics assume that the classification of input sequences (i.e. target vs. background) is independent of the occurrence of motifs within them. The statistics consider the total number of target sequences, background sequences and how many of each type contains the motif that is being checked for enrichment. From these numbers we can calculate the probability of observing the given number (or more) of target sequences with the motif by chance if we assume there is no relationship between the target sequences and the motif. The hypergeometric and binomial distributions are similar, except that the hypergeometric assumes sampling without replacement, while the binomial assumes sampling with replacement. The motif enrichment problem is more accurately described by the hypergeometric, however, the binomial has advantages. The difference between them is usually minor if there are a large number of sequences and the background sequences >> target sequences. In these cases, the binomial is preferred since it is faster to calculate. As a result it is the default statistic for **findMotifsGenome.pl** where the number of sequences is typically higher. However, if you use your own background that has a limited number of sequences, it might be a good idea to switch to the hypergeometric (use **"-h"** to force use of the hypergeometric). **findMotifs.pl** expects smaller number for promoter analysis and uses the hypergeometric by default.

One important note: Since HOMER uses an Oligo Table for much of the internal calculations of motif enrichment, where it does not explicitly know how many of the original sequences contain the motif, it approximates this number using the total number of observed motif occurrences in background and target sequences. It assumes the occurrences were equally distributed among the target or background sequences with replacement, were some of the sequences are likely to have more than one occurrence. It uses the expected number sequences to calculate the enrichment statistic (the final output reflects the actual enrichment based on the original sequences).

8. Matrix Optimization

HOMER takes the most enriched oligos from the global optimization step, transforms them into simple position specific probability matrices, and further optimizes them with a sensitive local optimization algorithm. This step is performed separately for each oligo, and will create the "motif probability matrix" as well as determine the optimal detection threshold to maximize the enrichment of the motif in the target vs. background sequences. The detection threshold is simply done by scoring each oligo in the data to the probability matrix, and then sorting the oligos by their similarity to the matrix. HOMER then steps down the list, effectively decreasing the detection threshold, including more and more oligos until an optimal enrichment is found. After this step, HOMER will create several new probability matrices based on the oligos found in different detection thresholds and check which one has the highest enrichment. This process is repeated until the enrichment can no longer be improved, producing a final motif.

9. Mask and Repeat

After the first "promising oligo" is optimized into a motif, the sequences bound by the motif to are removed from the analysis and the next promising oligo is optimized for the 2nd motif, and so on. This is repeated until the desired number of motifs are found ("**-S <#>**", default: 25). This is where the there is an important difference between the old (homer) and new (homer2) versions. The old version of homer would simply mask the oligos bound by the motif from the Oligo Table. For example if the motif was GAGGAW then GAGGAA and GAGGAT would be removed from the Oligo Table to avoid having the next motif find the same sequences. However, if GAGGAW was enriched in the data, there is a good chance that any 6-mer oligo like nGAGGA or AGGAWn would also be somewhat enriched in the data. This would cause homer to find multiple versions of the same motif and provide a little bit of confusion in the results.

To avoid this problem in the new version of HOMER (homer2), once a motif is optimized, HOMER revisits the original sequences and masks out the oligos making up the instance of the motif as well as oligos immediately adjacent to the site that overlap with at least one nucleotide. This helps provide much cleaner results, and allows greater sensitivity when co-enriched motifs. To make revert back to the old way of motif masking with homer2, specify "**-quickMask**" at the command line. You can also run the old version with "**-homer1**".

Screening for Enrichment of Known Motifs (homer2):

10. Load Motif Library

In order to search for Known Motifs in your data, HOMER loads a list of previously determined motifs from previous data. You can also add you own motifs by specifying them at the command line ("**-mknown <file>**") or by editing the primary file ("data/knownTFs/known.motifs"). HOMER doesn't screen all of TRANSFAC - partially due to motif quality (which can be low), and paritically due to the fact that we need a detection threshold.

11. Screen Each Motif

To find the enrichment for each motif, HOMER scans each sequence for instances of the motif and calculates the final enrichment by considering how many target vs. background sequences are considered "bound". ZOOPS (zero or one occurence per sequence) counting is used and the hypergeometric or binomial is used to calculate the significance.

Motif Analysis Output:

12. Motif Files (homer2, findMotifs.pl, findMotifsGenome.pl)

The true output of HOMER are "*.motif" files which contain the information necessary to identify future instance of motifs. They are reported in the output directories from findMotifs.pl and findMotifsGenome.pl. A typical motif file will look something like:

```
>ASTTCCTCTT 1-ASTTCCTCTT 8.059752 -23791.535714 0 T:17311.0(44 ...
```

0.726	0.002	0.170	0.103
0.002	0.494	0.354	0.151
0.016	0.017	0.014	0.954
0.005	0.006	0.027	0.963
0.002	0.995	0.002	0.002
0.002	0.989	0.008	0.002
0.004	0.311	0.148	0.538
0.002	0.757	0.233	0.009
0.276	0.153	0.030	0.542
0.189	0.214	0.055	0.543

The first row starts with a ">" followed by various information, and the other rows are the positions specific probabilities for each nucleotide (A/C/G/T). The header row is actually TAB delimited, and contains the following information:

1. ">" + Consensus sequence (not actually used for anything, can be blank) example:
>ASTTCCTCTT
2. Motif name (should be unique if several motifs are in the same file) example: 1-
ASTTCCTCTT or NFkB
3. Log odds detection threshold, used to determine bound vs. unbound sites (**mandatory**)
example: 8.059752
4. log P-value of enrichment, example: -23791.535714
5. 0 (A place holder for backward compatibility, used to describe "gapped" motifs in old version, turns out it wasn't very useful :)
6. Occurrence Information separated by commas, example:
T:17311.0(44.36%),B:2181.5(5.80%),P:1e-10317
 1. T:#(%) - number of target sequences with motif, % of total of total targets
 2. B:#(%) - number of background sequences with motif, % of total background
 3. P:# - final enrichment p-value
7. Motif statistics separated by commas, example:
Tpos:100.7,Tstd:32.6,Bpos:100.1,Bstd:64.6,StrandBias:0.0,Multiplicity:1.13
 1. Tpos: average position of motif in target sequences (0 = start of sequences)
 2. Tstd: standard deviation of position in target sequences
 3. Bpos: average position of motif in background sequences (0 = start of sequences)
 4. Bstd: standard deviation of position in background sequences
 5. StrandBias: log ratio of + strand occurrences to - strand occurrences.
 6. Multiplicity: The average number of occurrences per sequence in sequences with 1 or more binding site.

You can [easily create your own motif files](#), just remember that the **first 3 columns are required!!!**

13. De novo motif output (findMotifs.pl/findMotifsGenome.pl/compareMotifs.pl)

HOMER takes the motifs identified from *de novo* motif discovery step and tries to process and present them in a useful manner. An HTML page is created in the output directory named homerResults.html along with a directory named "homerResults/" that contains all of the image and other support files to create the page. These pages are explicitly created by running a subprogram called "**compareMotifs.pl**".

Comparison of Motif Matrices:

Motifs are first checked for redundancy to avoid presenting the same motifs over and over again. This is done by aligning each pair of motifs at each position (and their reverse opposites) and scoring their similarity to determine their best alignment. Starting with HOMER v3.3, matrices are compared using Pearson's correlation coefficient by converting each matrix into a vector of values. Neutral frequencies (0.25) are used in where the motif matrices do not overlap.

The old comparison was done by comparing the probability matrices using the formula below which manages the expectations of the calculations by scrambling the nucleotide identities as a

control. (freq1 and freq2 are the matrices for motif1 and motif2)

$$SimilarityScore = \frac{1}{MotifLength} \sum_i^{MotifLength} - \frac{(Observed_i - Expect_i)}{Expect_i}$$
$$Observed_i = \sum_j^{A,C,G,T} - (freq1_{ij} - freq2_{ij})^2$$
$$Expect_i = \sum_j^{A,C,G,T} \sum_k^{A,C,G,T} \frac{-(freq1_{ij} - freq2_{ik})^2}{4}$$

The output will be a score ranging from some lower bound (depending on the matrix frequencies) to 1, where 1 is complete similarity. By default the threshold for assigning similar motifs is 0.6, which is a reasonable cutoff in practice. This can be changed if you run **compareMotifs.pl** and change the **"-reduceThresh <#>"** parameter.

Motifs are next compared against a library of known motifs. For this step, all motifs in JASPAR and the "known" motifs are used for comparison. You can specify a custom motif library using **"-mcheck <motif library file>"** when using **findMotifs[Genome].pl** or **"-known <motif library file>"** when calling **compareMotifs.pl** directly.

By default, it looks for the file **"/path-to-homer/data/knownTFs/all.motifs"** to find the motif to compare with the de novo motifs. If **"-rna"** is specified, it will load the file **"/path-to-homer/data/knownTFs/all.rna.motifs"**.

An example of the output HTML is show below:

Homer de novo Motif Results

Known Motif Enrichment Results

Gene Ontology Enrichment Results

If Homer is having trouble matching a motif to a known motif, try copy/pasting the matrix file into [STAMP](#)

More information on motif finding results: [HOMER](#) | [Description of Results](#) | [Tips](#)

Total target sequences = 15212

Total background sequences = 34047

* - possible false positive

Rank	Motif	P-value	log P-value	% of Targets	% of Background	STD(Bg STD)	Best Match/Details	Motif File
1	GGAAATTCCT	1e-1835	-4.228e+03	28.11%	5.16%	37.7bp (63.1bp)	NFkB-p65(RHD)/GMI12787-p65-ChIP-Seq/Homer	motif file (matrix)
2	AAAGAGGAAGTG	1e-1716	-3.953e+03	34.50%	8.65%	47.8bp (62.6bp)	PB0058.1_Stpi1_1	motif file (matrix)
3	TTCTCGCAATCT	1e-1585	-3.651e+03	28.85%	6.39%	41.8bp (62.8bp)	MA0102.1_Cebpa	motif file (matrix)
4	TGAGTTCAT	1e-1004	-2.314e+03	25.07%	7.22%	49.2bp (61.0bp)	NF-E2(bZIP)/K562-NFE2-ChIP-Seq/Homer	motif file (matrix)
5	TCATATGCAAAAT	1e-262	-6.039e+02	5.52%	1.27%	47.8bp (59.7bp)	Oct2(POU/Homeobox)/Bcell-Oct2-ChIP-Seq/Homer	motif file (matrix)
6	ATGACGTCACTCT	1e-198	-4.565e+02	6.42%	2.09%	49.9bp (53.2bp)	c-Jun-CRE(bZIP)/K562-cJun-ChIP-Seq/Homer	motif file (matrix)
7	AAACCACA	1e-146	-3.370e+02	7.66%	3.30%	51.7bp (58.9bp)	RUNX1(Runt)/Jurkat-RUNX1-ChIP-Seq/Homer	motif file (matrix)

Depending on how the **findMotifs[Genome].pl** program that was executed, the "Known Motif Enrichment Results" and "Gene Ontology Enrichment Results" may or may not link to anything. Motifs are sorted based on p-value, and basic statistics about the motif (present in the motif files) is displayed.

The final column contains a link to the "motif file", which is important if you want to search for the motif in other sequences.

In the Best Match/Details column, HOMER will display the known motif which most closely


matched with the *de novo* motif. It is very important that you **TAKE THIS ASSIGNMENT WITH A GRAIN OF SALT!!!!** Unfortunately, sometimes the best match still isn't any good. Also, it is common that the "known" motif isn't any good to begin with. To investigate the assignment further, click on the "More Information" link which provides a page that looks like this:

Basic Information: The section contains basic information, including links to the motif file (normal and reverse opposite) and the pdf version of the motif logo.


Homer de novo Motif Results

motif1

Information for motif1



Reverse Opposite:



p-value:	1e-1835
log p-value:	-4.228e+03
Number of Target Sequences with motif	4276.0
Percentage of Target Sequences with motif	28.11%
Number of Background Sequences with motif	1756.8
Percentage of Background Sequences with motif	5.16%
Average Position of motif in Targets	99.3 +/- 37.7bp
Average Position of motif in Background	99.7 +/- 63.1bp
Strand Bias (log2 ratio + to - strand density)	-0.0
Multiplicity (# of sites on avg that occur together)	1.12
Motif File:	file (matrix) reverse opposite
PDF Format Logos:	forward logo reverse opposite

Followed by matches to known motifs. This section shows the alignments between the *de novo* motif and known motifs. It's important to check and see if these alignments look reasonable:

Matches to Known Motifs

NFkB-p65(RHD)/GM12787-p65-ChIP-Seq/Homer	
Match Rank: 1	
Score: 0.93	
Offset: -2	
Orientation: forward strand	
Alignment: --GGAATTYCCC NGGGGATTCCCC	
MA0061.1_NF-kappaB	
Match Rank: 2	
Score: 0.87	
Offset: -1	
Orientation: forward strand	
Alignment: -GGAATTYCCC GGGAATTCC-	
MA0105.1_NFKB1	
Match Rank: 3	
Score: 0.84	
Offset: -1	
Orientation: forward strand	
Alignment: -GGAATTYCCC GGGGATTCCCC	

Clicking on the "similar motifs" will show the other de novo motifs found during motif finding that resemble the motif but had a lower enrichment value. It contains a similar "header" as the "More Information" link, but below it shows the motifs that were considered similar. It is usually a good idea to check this list over - sometimes a distinct motif will be grouped incorrectly in the list because it shares a couple residues.

Similar de novo motifs found

Rank	Match Score	Redundant Motif	P-value	log P-value	% of Targets	% of Background	Motif file
1	0.918		1e-1776	-4089.766852	26.30%	4.60%	motif file (matrix)
2	0.873		1e-1711	-3941.421170	25.85%	4.62%	motif file (matrix)
3	0.844		1e-968	-2231.146991	25.56%	7.71%	motif file (matrix)
4	0.616		1e-259	-597.025749	12.81%	5.44%	motif file (matrix)
5	0.662		1e-233	-537.315538	13.40%	6.12%	motif file (matrix)
6	0.795		1e-222	-512.488031	22.69%	13.20%	motif file (matrix)
7	0.874		1e-148	-341.450152	20.88%	13.25%	motif file (matrix)







To rerun this part of the analysis on any arbitrary set of motifs, simply run the "compareMotifs.pl" command (use without any command line parameters to get the usage options).

14. Known motif output

Known motif enrichment is displayed as a HTML file (knownResults.html). The page sorts the results based on enrichment and displays basic information:

Homer Known Motif Enrichment Results

[Homer de novo Motif Results](#)
[Gene Ontology Enrichment Results](#)
[Known Motif Enrichment Results \(txt file\)](#)
Total Target Sequences = 15213, Total Background Sequences = 34081

Rank	Motif	Name	P-value	log P-value	# Target Sequences with Motif	% of Targets Sequences with Motif	# Background Sequences with Motif	% of Background Sequences with Motif	Motif File	PDF
1		NFkB-p65(RHD)/GM12787-p65-ChIP-Seq/Homer	1e-1707	-3.931e+03	3855.0	25.34%	1506.4	4.42%	motif file (matrix)	pdf
2		CEBP(bZIP)/CEBPb-ChIP-Seq/Homer	1e-1310	-3.018e+03	3423.0	22.50%	1551.0	4.55%	motif file (matrix)	pdf
3		PU.1(ETS)/ThioMac-PU.1-ChIP-Seq/Homer	1e-1288	-2.967e+03	3413.0	22.44%	1569.7	4.61%	motif file (matrix)	pdf
4		AP-1(bZIP)/ThioMac-PU.1-ChIP-Seq/Homer	1e-947	-2.183e+03	3251.0	21.37%	1900.8	5.58%	motif file (matrix)	pdf
5		NFkB-p65-Rel(RHD)/LPS-exp/Homer	1e-936	-2.157e+03	1163.0	7.65%	166.2	0.49%	motif file (matrix)	pdf
6		ETS1(ETS)/Jurkat-ETS1-ChIP-Seq/Homer	1e-885	-2.039e+03	4447.0	29.23%	3568.0	10.47%	motif file (matrix)	pdf



Can't figure something out? Questions, comments, concerns, or other feedback:
cbenner@ucsd.edu



HOMER

Software for motif discovery and ChIP-Seq analysis

Motif Finding with HOMER from FASTA files

Most of HOMER's functionality is built around either promoter or genomic position based analysis, and aims to manage the sequence manipulation, hiding it from the user. However, if you have some sequences that you would like HOMER to analyze, the program **findMotifs.pl** accepts **FASTA** formatted files for analysis. Alternatively you could use the **homer2** executable which also accepts FASTA files as input.

HOMER is designed to analyze high-throughput data using differential motif discovery, which means you **MUST** have both target and background sequences, and in each case you should have several (preferably thousands) of sequences in each set that are roughly the same length.

A quick note about FASTA files - Each sequence should have a unique identifier. In theory, HOMER should be flexible with what is in the header line, but if you're having trouble please just keep it simple with minimal quite-space, especially tabs. For example:

```
>NM_003456
AAGGCCTGAGATAGCTAGAGCTGAGAGTTTTCCACACG
```

Running findMotifs.pl with FASTA files:

To find motifs from FASTA files, run findMotifs.pl with the target sequence FASTA file as the first command-line argument, and use the option "**-fasta <file>**" to specify the background FASTA file. You CANNOT NOT specify a background file - that would defeat the purpose of differential motif finding.

```
findMotifs.pl <targetSequences.fa> fasta <output directory> -fasta  
<background.fa> [options]
```

NOTE: you must choose an "organism" for the 2nd argument to keep with the structure of the command, even though this isn't actually relevant for FASTA based analysis. Organism doesn't have to match the data in the FASTA files. You can use a valid organism or just put "**fasta**" as a place holder. i.e.:

```
findMotifs.pl chuckNorrisGenes.fa human analysis_output/ -fasta  
normalHumanGenes.fa
```

Many other options are available to control motif finding parameters.

findMotifs.pl will perform GC normalization and autonormalization by default (see [here for more details](#)).

Finding instances of motifs with FASTA files:

To find instance of a motif, run the same command used for motif discovery above but add the option "**-find <motif file>**". Motif results will be sent to stdout, so to capture the results in a file Add "**> outputfile**" to the end of the command.

findMotifs.pl <targetSequences.fa> fasta <output directory> -fasta <background.fa> [options] -find motif1.motif > outputfile.txt

For more information on the output file format, see [here](#).

Using homer2 directly with FASTA files:

homer2 is the motif finding executable, and it can choke down FASTA files if you want to avoid all the nonsense above. Running the **homer2** command will also give you access to other options for optimizing the motif finding process. **homer2** works by first specifying a command, and then the appropriate options:

homer2 <command> [options]
i.e. homer2 denovo -i input.fa -b background.fa > outputfile.txt

To find instances of the output motifs, use "**homer2 find**". To see other commands, just type "**homer2**".



Can't figure something out? Questions, comments, concerns, or other feedback:
cbenner@ucsd.edu



HOMER

Software for motif discovery and next-gen sequencing analysis

RNA Motif Analysis

HOMER was not originally designed with RNA in mind, but it can be used to successfully analyze data for RNA motifs. By RNA motifs, we mean short sequence elements in RNA sequences akin to DNA motifs, not structural elements such as hairpins and stuff like that. For example, HOMER can be used to successfully determine miRNA seeds in sets of co-regulated mRNAs, or RNA binding elements in CLIP-Seq data.

The `"-rna"` option can be used with `findMotifs.pl` and `findMotifsGenome.pl`, resulting in + strand only motif searching and motif display/matching with "U" instead of "T". HOMER does not contain a list of well known "RNA motifs" yet, so no "known motif" analysis is performed. If using FASTA files, please use "T" (normal DNA encoding) in the input files for now.

Analyzing Co-regulated Gene Lists for RNA motifs

HOMER contains preconfigured PROMOTER sets comprised of RefSeq mRNA sequences, or only the 5' and 3' UTRs. These are useful for analyzing gene lists for motifs in their mRNAs. To run the analysis, us `findMotifs.pl` with a mRNA PROMOTER set, and options for RNA motifs will be automatically set.

```
findMotifs.pl mir1-downregulated.genes.txt human-mRNA MotifOutput/ -rna -len 8
```

You don't actually need to specify `-rna` for this case since with the use of `"human-mRNA"` it's understood. Anyway, the output will look something like this:

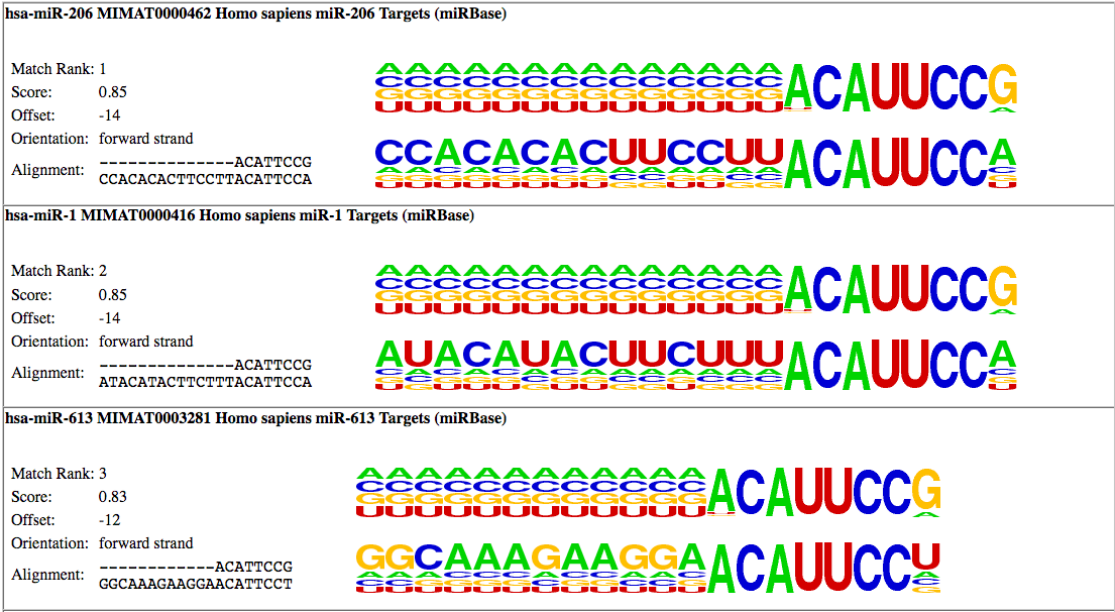
Homer *de novo* Motif Results

[Known Motif Enrichment Results](#)
[Gene Ontology Enrichment Results](#)
If Homer is having trouble matching a motif to a known motif, try copy/pasting the matrix file into [STAMP](#)
More information on motif finding results: [HOMER](#) | [Description of Results](#) | [Tips](#)
Total target sequences = 1351
Total background sequences = 19001
** - possible false positive*

Rank	Motif	P-value	log P-value	% of Targets	% of Background	STD(Bg STD)	Best Match/Details
1	ACAUUCCG	1e-116	-2.684e+02	55.29%	24.70%	1843.3bp (1700.6bp)	hsa-miR-206 MIMAT0000462 Homo sapiens miR-206 Targets (miRBase) More Information Similar Motifs Found
2	AGUAUGAC	1e-41	-9.637e+01	68.39%	49.40%	1877.0bp (1667.9bp)	hsa-miR-485-3p MIMAT0002176 Homo sapiens miR-485-3p Targets (miRBase) More Information Similar Motifs Found
3	CAUCGACG	1e-28	-6.530e+01	60.92%	45.25%	1651.5bp (1428.1bp)	hsa-miR-181a* MIMAT0000270 Homo sapiens miR-181a* Targets (miRBase) More Information Similar Motifs Found

For now, HOMER will try to match the results to the human list of miRNA seeds (from miRBase):

Matches to Known Motifs



In this case, the motif matches the miR-1 consensus seed (which is shared by miR-206 and miR-613).

There are two RNA specific options for findMotifs.pl in rna mode:

- min <#> : minimum length of mRNA to consider (basically removes extremely short mRNA sequences from the analysis)
- max <#> : maximum length of mRNA to consider (removes really long RNAs from the analysis)

Analyzing CLIP-Seq for RNA motifs

HOMER can analyze strand-specific genomic regions for motifs, such as the regions that would be defined from CLIP-Seq. To do this, just run findMotifsGenome.pl using the "-rna" flag (make sure your regions are strand specific!!). For now, HOMER just uses the same random genomic background used for ChIP-Seq motif finding. You could imagine that a better RNA motif finding background would be RNA, i.e. strand specific exon/intron sequences. You'd be right, but managing this with respect to the different experiments (i.e. intronic binding vs. mRNA binding vs. non-coding RNA binding) is tricky and for now left up to the user (you can specify your own strand specific background with "-bg <peak/BED file>"). Trying this with FOX CLIP-Seq data:

```
findMotifsGenome.pl fox2.clip.bed hg17 MotifOutput -rna
```

This will give the following results (which resembles a UGCAUG FOX motif):

Homer *de novo* Motif Results

[Known Motif Enrichment Results](#)

[Gene Ontology Enrichment Results](#)




If Homer is having trouble matching a motif to a known motif, try copy/pasting the matrix file into [STAMP](#)

More information on motif finding results: [HOMER](#) | [Description of Results](#) | [Tips](#)

Total target sequences = 8476

Total background sequences = 37836

* - possible false positive

Rank	Motif	P-value	log P-value	% of Targets	% of Background	STD(Bg STD)	Best Match/Details
1		1e-273	-6.306e+02	30.57%	15.27%	53.9bp (73.6bp)	dme-miR-954-3p MIMAT0020846 Drosophila (miRBase) More Information Similar Motifs Found
2		1e-34	-7.881e+01	5.65%	3.08%	56.0bp (76.5bp)	mmu-miR-3060 MIMAT0014827 Mus musculus (miRBase) More Information Similar Motifs Found
3		1e-21	-4.901e+01	0.28%	0.02%	54.1bp (43.8bp)	hsa-miR-4457 MIMAT0018979 Homo sapiens (miRBase) More Information Similar Motifs Found



Can't figure something out? Questions, comments, concerns, or other feedback:
cbenner@ucsd.edu



HOMER

Software for motif discovery and next-sequencing analysis

Practical Tips to Motif Finding with HOMER

Below are some general tips for getting the most out of you motif analysis when using HOMER. Be sure to look over [this section about judging motif quality!](#)

What to do if motif finding takes too long...

Ctrl+C... If you are using reasonable parameters (see next section), it shouldn't take more than an hour or so, and in most cases much less.

Choosing the length of motifs to find

It's almost always a good idea to start with the default parameters. Resist the urge to find motifs larger than 12 bp the first time around. Longer motifs will show up as different short motifs when finding shorter motifs. If there aren't any truly significant motifs when looking at short motifs, it is unlikely that you will find good long motifs either. And it doesn't take much time to check for short motifs.

i.e. **-S 25 -len 8,10,12**

Once you do find motifs that look promising, try looking for longer motifs.

Finding Long Motifs

The new version of HOMER (v3.0+) is better at looking for long motifs. However, it can be tricky looking for long motifs because the search space gets very large. Also, the running time on longer motifs increases and may break your patience.

Since HOMER is an empirical motif finding program, it starts from actual oligos present in the sequence and attempts to figure out if they are enriched. If you are looking at 20 bp sequences, there is a good chance that they are all more-or-less unique in your data set with only 1 instance in either the target or background sequences. HOMER normally allows mismatches in the original oligo to see if the oligo together with similar oligos are collectively enriched. The problem is that this technique starts to break down at long lengths. It takes many mismatches to find enough related sequences to assess enrichment, and it is computationally expensive to find them.

To maintain sensitivity for longer motifs:

Increase the "-mis <#>" option to allow more mismatches. In practice, I would use at least "-mis 4" or "-mis 5" for sensitive detection of 20 bp motifs. If the data set is for a strong motif (i.e. CTCF ChIP-Seq peaks), then you don't have to worry about this so much since the motif signal is very strong.

To find longer version of a given motif:

The local optimization phase handles long motifs pretty well - long motifs cause more of a problem with the global search phase. Usually long motifs show enrichment for parts at shorter motif lengths. Another strategy is to first find a short version of the motif (i.e. -len 12), and then rerun HOMER and tell it to optimize the motif at a longer motif length with the "-opt <motif file>". To do this with a motif named "motif1.motif":

findMotifsGenome.pl peaks.txt hg18r OutputDirectory -opt motif1.motif -len 30

This will enlarge the motif(s) in the motif1.motif to 30 bp and optimize them.

Other things to try:

- try to reduce the number of target sequences to include only high quality sequences (such as "focused" peaks or peak with the highest peak scores).
- try limiting the length of sequences used (i.e. "-size 50" when using **findMotifsGenome.pl**)
- try limiting the total number of background sequences (i.e. "-N 20000" when using **findMotifsGenome.pl**)

In a practical sense, you should be able to search for motifs of length 20 or 30 when analyzing ~10k peaks with parameters "-len 20,30 -size 50 -N 25000 -mis 5". HOMER wasn't really designed to find really long motifs; since it is an empirical motif finder, the sequence "space" gets a bit sparse at lengths >16, but in practice it still works.

How many sequences can HOMER handle?

In theory, a lot (i.e. millions). It has been designed to work well with ~10k target sequences and 50k background sequences. If you are using a large number of sequences with **findMotifs.pl**, you may want to use the "-b" option, which switches to the [cumulative binomial distribution](#) for motif scoring, which is faster to calculate and gives essentially the same results when using large numbers of sequences. The binomial is used by default in **findMotifsGenome.pl**. (I guess it should be called BOMER !?).

Choosing background sequences

Most of the methods in HOMER attempt to select the proper background for you, but in some cases this doesn't work. Normally, HOMER attempts to normalize the GC content in target and background sequences. If you believe normalizing the CpG content is better, use the option "-cpg" when performing motif finding with either **findMotifs.pl** or **findMotifsGenome.pl**.

In some cases the user may have a better idea of what the background should be, so HOMER offers the following options:

Promoters: When using analyzing promoters with **findMotifs.pl**, if you wish to use a specific set of promoters as background, place them in a text file (1st column is the ID) and use the "-bg <background IDs file>" option. Genes found in the target and background will be removed from the background set so that they don't cancel out each other. Examples:

- Use expressed genes from a microarray as background
- Use only genes represented on the microarray as background

Genomic Regions: When analyzing peaks/regions with **findMotifsGenome.pl**, you can specify the genomic regions of appropriate background regions by placing them in their own peak file and using the "-bg <background peak file>". Examples:

- Specify peaks common to two cell types as background when trying to find motifs specific to a set of cell-type specific peaks - this will help cancel out the primary motif and reveal the co-enriched motifs
- If peaks are near Exons, specify regions on Exons as background to remove triplet bias.

FASTA Files: Here you have (the necessary) freedom to specify whatever you want!

Please note, that if the number of background sequences is small, or similar in number to the number of target sequences, you should consider switching to the hypergeometric distribution to improve accuracy when using **findMotifsGenome.pl** ("-h").

You may also want to disable CpG/GC normalization depending on how you selected your background, which can be done with "-noweight".


Sequence Bias, GC/CpG normalization, and Autonormalization

By default, homer performs several normalization steps to make sure the sequences that are being analyzed look reasonable (details [here](#)). Since GC% differences are the largest source of bias, these are dealt with during the background selection stage to minimize any issues.

Other types of sequence bias may be present in your data. The purpose of the autonormalization routines ("-nlen <#>" and "-olen <#>") are there to help deal with this type of bias. If your results have strong enrichment for simple nucleotide repeats, you may want to try "-olen <#>" which will more aggressively normalize the data.

How to Judge the Quality of the Motifs Found

WARNING: Because this is the hardest thing for people to understand, I'll say it again here. HOMER will print the best guess for the motif next to the motif results, but before you tell your adviser that your factor is enriched for that motif, it is highly recommended that you look at the alignment!!! Here is an example of what might be going on:

8		3.941e-24	-5.389e+01	YY1_01 More Information Similar Motifs Found
---	---	-----------	------------	---

In this case, HOMER has identified YY1 as the "best guess" match for this *de novo* motif. Well, lets click on "More Information" and see what's up:

YY1(M00069)


Match Rank: 2

Score: 6.11

Offset: 0

Orientation: reverse strand

Alignment: AGCAGGCA-----
ANCAGNCAAGATGGCCGNGN



As you can see in this case, the motif aligns to the edge of the known YY1 motif, and not to the core of the YY1 motif (CAAGATGGC). This doesn't mean that the YY1 motif is not enriched in your data, but unless there are other motif results that show enrichment of the other parts of the YY1 motif, it is not likely that the YY1 motif is enriched in your data set.

And as always, remember that HOMER is a *de novo* motif tool!!! Even though HOMER will guess the best match, if it is a novel motif, your don't want to trust that match anyway. Hence, the you can see the importance of viewing the alignment and getting a feel for what evidence exists either for or against this assignment.

There are many cases where HOMER will find motifs with very low p-values, but the motifs might look "suspicious". Poor quality motifs can be loosely classified into the following groups:

Low Complexity Motifs:

(less of a problem with the v3.0+) These types of motifs tend to show preference for same collection of 1, 2, 3, or 4 nucleotides in each position and are typically very degenerate. For example:



These motifs typically arise when a systematic bias exists between target and background sequence sets. Commonly they will be very high in GC-content, in which case you may want to try adding "-gc" to your motif finding command to normalize by total GC-content instead of CpG-content.

Other times this will come up when analyzing sequences for various genomic features that have not been controlled for in the background - for example, comparing sequences from promoters to random genomic background sequences in some organisms will show preferences for purines or pyrimidines. HOMER is very sensitive, so if there is a bias in the composition of the sequences, HOMER will likely pick it up. Autonormalization in the new version minimizes this problem.

Simple Repeat Motifs:

(less of a problem with the v3.0+) Some times motifs will show repeats of certain patterns:



Usually motifs like this will be accompanied by several other motifs looking highly similar. Unless there is a good reason to believe these may be real, it's best to assume there is likely a problem with the background. These can arise if your target sequences are highly enriched on exons (think triplets) and other types of sequences, and if "-gc" doesn't help, you may have to think hard about the types of sequences that you are trying to analyze and try to match them. (i.e. Promoters vs. Exons vs. Exons etc.) You can also try upping the ante by using "-olen <#>" to autonormalize sequence bias at the oligo level.

Small Quantity Motifs / Repeats:

These are a little harder to explain. These look like real motifs but are found in an incredibly low percentage of targets - i.e. like an oligo or part of a repeat that is in a couple of the target sequences that appears as a significant motif. Statistically speaking they are enriched, but likely not real. These are the biggest problem when looking for motifs in promoters from a small list of regulated genes. In principle, in a motif is present in **less than 5% of the targets sequences**, there may be a problem.

Leftover Junk:

These are motifs that appear in your lower in your results list after you've discovered high quality motifs. If an element is highly enriched in your sequences, HOMER will (hopefully) find it, mask it, and then continue to look for motifs. In this case, many of the other motifs that HOMER finds will be offsets or degenerate versions of highly enriched motif(s) found at the beginning. For example (another PU.1 example):

The top motif identified:

Rank	Motif	P-value	log P-pvalue	Best Match/Details
1		0.000e+00	-2.938e+04	PU.1/ThioMac-PU.1-ChIP-Seq/Homer More Information

Examples further down the list:

8		0.000e+00	-3.711e+03	PU.1/ThioMac-PU.1-ChIP-Seq/Homer More Information
22		0.000e+00	-1.692e+03	STAT1/Stat More Information
74		6.727e-190	-4.356e+02	MAF(M00648) More Information

This are not necessarily negative results, but they should be place in context. This commonly happens in ChIP-Seq data sets where the immunoprecipitated protein is highly expressed and binds strongly a ton of binding sites. These "other" motifs are likely also capable of binding PU.1 and probably represent low affinity binding sites, but giving them too much individual attention is not recommended in this context given they are motifs that have been constructed using leftover oligos in the motif finding process that didn't make it into the most highly enrichment motifs. A safer way to approach these elements is to repeat the motif finding procedure with regions lacking the top motif, or by adding "-mask <motif file>" to the motif finding command to cleanly mask the top motif from the motif finding procedure.



Can't figure something out? Questions, comments, concerns, or other feedback:
cbenner@ucsd.edu



HOMER

Software for motif discovery and next-gen sequencing analysis

Creating Custom Motif Matrices

A common task when doing regulatory element analysis is to scan for a specific sequence/motif - but not necessarily with one that is found using motif finding. Often you want to find a very specific sequence, or you want to load your own motif matrix derived from another source. This page will help explain how to get HOMER to play nice with your custom motifs.

Below is a description of the HOMER *.motif format for specifying motifs, as well as some tricks & tips on creating simple motif files.

*.motif format files

HOMER works exclusively with homer motif formatted files - so if you want to find a sequence or motif with HOMER, you must first create a "motif" file. A typical motif file will look something like:

```
>ASTTCCTCTT 1-ASTTCCTCTT 8.059752 -23791.535714 0
T:17311.0(44 ...
0.726 0.002 0.170 0.103
0.002 0.494 0.354 0.151
0.016 0.017 0.014 0.954
0.005 0.006 0.027 0.963
0.002 0.995 0.002 0.002
0.002 0.989 0.008 0.002
0.004 0.311 0.148 0.538
0.002 0.757 0.233 0.009
0.276 0.153 0.030 0.542
0.189 0.214 0.055 0.543
```

The first row starts with a ">" followed by various information, and the other rows are the positions specific probabilities for each nucleotide (A/C/G/T). These values do not need to be between 0-1. HOMER will automatically normalize whatever values are there, so interger counts are ok. The header row is actually TAB delimited, and contains the following information:

1. ">" + **Consensus sequence (not actually used for anything, can be blank) example: >ASTTCCTCTT**
2. **Motif name (should be unique if several motifs are in the same file) example: 1-ASTTCCTCTT or NFkB**
3. **Log odds detection threshold, used to determine bound vs. unbound sites (mandatory) example: 8.059752**

4. (optional) log P-value of enrichment, example: -23791.535714
5. (optional) 0 (A place holder for backward compatibility, used to describe "gapped" motifs in old version, turns out it wasn't very useful :)
6. (optional) Occurrence Information separated by commas, example:
T:17311.0(44.36%),B:2181.5(5.80%),P:1e-10317
 1. T:#(%) - number of target sequences with motif, % of total of total targets
 2. B:#(%) - number of background sequences with motif, % of total background
 3. P:# - final enrichment p-value
7. (optional) Motif statistics separated by commas, example:
Tpos:100.7,Tstd:32.6,Bpos:100.1,Bstd:64.6,StrandBias:0.0,Multiplicity:1.13
 1. Tpos: average position of motif in target sequences (0 = start of sequences)
 2. Tstd: standard deviation of position in target sequences
 3. Bpos: average position of motif in background sequences (0 = start of sequences)
 4. Bstd: standard deviation of position in background sequences
 5. StrandBias: log ratio of + strand occurrences to - strand occurrences.
 6. Multiplicity: The average number of occurrences per sequence in sequences with 1 or more binding site.

Only the first 3 columns are needed. In fact, the rest of the columns are really just statistics from motif finding and aren't important when searching for instances of a motif.

The MOST IMPORTANT value is the 3rd column - this sets the detection threshold, which specifies whether a given sequence is enough of a "match" to be considered recognized by the motif. More on that below.

Creating Simple Motifs with seq2profile.pl

HOMER comes with a handy little tool called **seq2profile.pl**. This program automates the creation of motifs from consensus sequences from letters ACGTN.

seq2profile.pl <consensus> [# mismatches] [name] > output.motif
i.e. **seq2profile.pl GGAAGT 0 ets > output.motif**

Output from this example looks like this:

```
>GGAAGT ets 8.28973911259755
0.001 0.001 0.997 0.001
0.001 0.001 0.997 0.001
0.997 0.001 0.001 0.001
0.997 0.001 0.001 0.001
0.001 0.001 0.997 0.001
0.001 0.001 0.001 0.997
```

This script will automatically set the 3rd column to a threshold to only detect

perfect matches. Using this file with tools like `annotatePeaks.pl` will only find sequences matching GGAAGT (or ACTTCC on the reverse strand) If we want to allow up to one mismatch:

i.e. `seq2profile.pl GGAAGT 1 ets > output.motif`

Output from this example looks like this:

```
>GGAAGT ets 1.38498834263571
0.001 0.001 0.997 0.001
0.001 0.001 0.997 0.001
0.997 0.001 0.001 0.001
0.997 0.001 0.001 0.001
0.001 0.001 0.997 0.001
0.001 0.001 0.001 0.997
```

The detection threshold now changed to allow up to one mismatch. This will now identify GGAAGT, GGAAGA, GGAAGC, GGAAGG, ... etc.

Creating motif files manually

You can also create a motif using a text editor or Excel. The first line should contain 3 tab-separated columns with ">whatever", "name", and "threshold", followed by motif matrix. You can also start from an existing motif or use `seq2profile.pl` to start a motif for you.

Once you get the probabilities the way you want it (i.e. maybe you copied them from JASPAR, or from in vitro affinity selection, or a set of binding sites, or whatever), you need to pick the correct detection threshold. During motif discovery, HOMER optimizes this threshold as part of the algorithm. However, since you're making up your own motif, you need to figure out how degenerate you want your motif to be. This is easily the biggest hang-up for people new to the concept of probability matrices and motif finding.

Motif Scanning

In order to select the proper detection threshold, it helps to understand how motifs are "scored" by HOMER. Any given sequence can be scored using the probability matrix. HOMER calculates the score by adding the "log-odds probabilities" for the nucleotide found in each position.

Score for GGATGT

$$\text{score} = \log(pG_1/0.25) + \log(pG_2/0.25) + \log(pA_3/0.25) + \log(pT_4/0.25) + \log(pG_5/0.25) + \log(pT_6/0.25)$$

If the nucleotide in the given position has a high probability in the motif matrix (i.e. 0.9), a positive number is added to the score. If the nucleotide as a neutral probability (i.e. 0.25), then the score is unchanged, and if a

nucleotide is disfavored in the probability matrix (i.e. 0.001), then a negative number is added to the score.

Homer fixes the log-odds expectation at 0.25 for each nucleotide. If an organism has a strong imbalance in nucleotide composition, this causes HOMER sacrifices some sensitivity by keeping the expectation at 0.25 instead of adjusting this to reflect the general sequence composition. However, the huge upside to fixing this at 0.25 is that you can use the motif with HOMER on any sequence in any context and always get recognize the same sequences, which makes life much easier.

If the score is above the threshold (from the 3rd column of the motif file), HOMER will identify the sequence as "recognized" by the motif. If the score is lower, the sequence will be ignored. Most motif thresholds are around 5.0-10.0

To select the correct threshold, you may need to "guess and check" your results to ensure your motif is recognizing the correct sequences. Usually, you can start with a threshold around 5-10. For example, run **findMotifs.pl/findMotifsGenome.pl** with "**-find motifFile.motif**". The score of each motif is found in the 6th column of the output. Looking through the file and the sequences that were identified will give you a better idea of what to set the score at.



Can't figure something out? Questions, comments, concerns, or other feedback:
cbenner@ucsd.edu



HOMER

Software for motif discovery and next-gen sequencing analysis

Primary Research Data

Below are collections of data produced by or in collaboration with the [Glass Lab](#) at UCSD.

UCSC Genome Browser visualization files

Each of the files below can be loaded as custom tracks into the [UCSC Genome Browser](#). A couple of things to note before trying this:

1. Pay attention to the **GENOME VERSION**. Researchers are constantly trying to improve reference genomes, and when a new version comes out the absolute position of everything changes. If you upload these files to the wrong version, sometimes it will complain because certain chromosomes or random fragments do not exist. If you get this error make sure the version is correct. Sometimes it won't complain, which is worse since all of the peaks etc. will be in the wrong positions.
2. **DO NOT UNZIP** these files when uploading them. UCSC takes zipped files, and if you try to upload an unzipped version you'll find it's harder to get it to successfully load.
3. If you're having trouble loading them, try "right-clicking" on the links below and "Copy Link Location". You can past this in to the field when loading custom tracks - this way UCSC will directly load the file from the biowhat.ucsd.edu server instead of from your computer.
4. Don't like my ugly colors for some of the tracks? Colors for each experiment were generated randomly, and if you don't like them, or they are hard to see, or you want to stick to a different color coding scheme, click on the "manage custom tracks" button and then click directly on the name of the track. This will give you access to the configuration data for the track. There will be a part of the text box with "color=r,g,b" that you can directly modify to get the color you want.

Notes on Homer UCSC files

Files posted here were created in BedGraph format. For details on how they were made, click [here](#). Each of these files have been made such that when zipped they are approximately 50 MB. This involves smoothing the raw data to a degree so that the file size doesn't get too big.

All files were updated on 08/31/2010 to correct for a change in the way

UCSC handles the files.

Data from the [Glass Lab](#)

Heinz et al. - PU.1 cistrome in Macrophages and B cells

Data below from:

Heinz S, Benner C, Spann N, Bertolino E, Lin YC, Laslo P, Cheng JX, Murre C, Singh H, Glass CK. [Simple Combinations of Lineage-Determining Transcription Factors Prime cis-Regulatory Elements Required for Macrophage and B Cell Identities](#). Mol Cell 2010 May 28;38(4):576-589. PMID: [20513432](#)

Link to the raw data: [GSE21512](#)

Macrophage (Thioglycollate Elicited Peritoneal Macrophages)

PU.1 [[UCSC BedGraph](#)] [[Alignment BED](#)] (mm8)
C/EBPb [[UCSC BedGraph](#)] [[Alignment BED](#)] (mm8)
C/EBPa [[UCSC BedGraph](#)] [[Alignment BED](#)] (mm8)
H3K4me1 [[UCSC BedGraph](#)] [[Alignment BED](#)] (mm8)
H3K4me3 [[UCSC BedGraph](#)] [[Alignment BED](#)] (mm8)
Input [[UCSC BedGraph](#)] [[Alignment BED](#)] (mm8)

Naive B cells (Spleen)

PU.1 [[UCSC BedGraph](#)] [[Alignment BED](#)] (mm8)
Oct-2 [[UCSC BedGraph](#)] [[Alignment BED](#)] (mm8)
H3K4me1 [[UCSC BedGraph](#)] [[Alignment BED](#)] (mm8)
H3K4me3 [[UCSC BedGraph](#)] [[Alignment BED](#)] (mm8)
Input [[UCSC BedGraph](#)] [[Alignment BED](#)] (mm8)

Bone Marrow Derived Macrophages

PU.1 [[UCSC BedGraph](#)] [[Alignment BED](#)] (mm8)
PU.1 (LXRa/b -/-) [[UCSC BedGraph](#)] [[Alignment BED](#)] (mm8)
H3K4me1 [[UCSC BedGraph](#)] [[Alignment BED](#)] (mm8)
H3K4me1 (LXRa/b -/-) [[UCSC BedGraph](#)] [[Alignment BED](#)] (mm8)

RAW 264.7 (Macrophage-like Cell Line, expressing BirA and BLRP tagged LXRb)

BLRP-LXRb (1h GW3965 [synthetic LXR ligand]) [[UCSC BedGraph](#)] [[Alignment BED](#)] (mm8)
BirA input (1h GW3965 [synthetic LXR ligand]) [[UCSC BedGraph](#)]

[[Alignment BED](#)] (mm8)

PUER Cells (PU.1-/- line with PU.1-ER fusion)

PU.1 (PU.1-/-) [[UCSC BedGraph](#)] [[Alignment BED](#)] (mm8)
C/EBPb (PU.1-/-) [[UCSC BedGraph](#)] [[Alignment BED](#)] (mm8)
H3K4me1 (PU.1-/-) [[UCSC BedGraph](#)] [[Alignment BED](#)] (mm8)

PU.1 (0h tamoxifen) [[UCSC BedGraph](#)] [[Alignment BED](#)] (mm8)
C/EBPb (0h tamoxifen) [[UCSC BedGraph](#)] [[Alignment BED](#)] (mm8)
H3K4me1 (0h tamoxifen) [[UCSC BedGraph](#)] [[Alignment BED](#)] (mm8)
MNase-Seq (mononucleosomes) (0h tamoxifen) [*No UCSC File*]
[[Alignment BED](#)] (mm8)

PU.1 (1h tamoxifen) [[UCSC BedGraph](#)] [[Alignment BED](#)] (mm8)
C/EBPb (1h tamoxifen) [[UCSC BedGraph](#)] [[Alignment BED](#)] (mm8)
H3K4me1 (1h tamoxifen) [[UCSC BedGraph](#)] [[Alignment BED](#)] (mm8)
MNase-Seq (mononucleosomes) (1h tamoxifen) [*No UCSC File*]
[[Alignment BED](#)] (mm8)

PU.1 (6h tamoxifen) [[UCSC BedGraph](#)] [[Alignment BED](#)] (mm8)
C/EBPb (6h tamoxifen) [[UCSC BedGraph](#)] [[Alignment BED](#)] (mm8)

PU.1 (24h tamoxifen) [[UCSC BedGraph](#)] [[Alignment BED](#)] (mm8)
C/EBPb (24h tamoxifen) [[UCSC BedGraph](#)] [[Alignment BED](#)] (mm8)
H3K4me1 (24h tamoxifen) [[UCSC BedGraph](#)] [[Alignment BED](#)]
(mm8)

PU.1 (48h tamoxifen) [[UCSC BedGraph](#)] [[Alignment BED](#)] (mm8)
C/EBPb (48h tamoxifen) [[UCSC BedGraph](#)] [[Alignment BED](#)] (mm8)

pre-proB cells (E2A-/-) [Reconstituted with either wild-type E2A(E47) or mutant(bHLH)]

PU.1 [[UCSC BedGraph](#)] [[Alignment BED](#)] (mm8)
PU.1 (E47ER, 6h after tamoxifen treatment) [[UCSC BedGraph](#)]
[[Alignment BED](#)] (mm8)
PU.1 (bHLHER, 6h after tamoxifen treatment) [[UCSC BedGraph](#)]
[[Alignment BED](#)] (mm8)

pre-proB cells (EBF-/-)

PU.1 [[UCSC BedGraph](#)] [[Alignment BED](#)] (mm8)

proB cells (RAG1-/-)

PU.1 [[UCSC BedGraph](#)] [[Alignment BED](#)] (mm8)

Sullivan et al. - SRF (Serum Response Factor) cistrome in Macrophages

Data below from:

Sullivan AL, Benner C, Heinz S, Huang W, Xie L, Miano JM, Glass CK. [SRF utilizes distinct promoter and enhancer-based mechanisms to regulate cytoskeletal gene expression in macrophages](#). Mol Cell Biol. 2010 Dec 6. PMID: [21135125](#)

Macrophage (Thioglycollate Elicited Peritoneal Macrophages)

SRF [[UCSC BedGraph](#)] [[Alignment BED](#)] (mm8) [[FASTQ](#)]
SRF in SRF^{-/-} mice [[UCSC BedGraph](#)] [[Alignment BED](#)] (mm8) [[FASTQ](#)][[FASTQ](#)]
PU.1 in SRF^{-/-} mice [[UCSC BedGraph](#)] [[Alignment BED](#)] (mm8) (for wild type PU.1 see above) [[FASTQ](#)]

PUER Cells (PU.1^{-/-} line with PU.1-ER fusion)

SRF (PU.1^{-/-}) [[UCSC BedGraph](#)] [[Alignment BED](#)] (mm8) [[FASTQ](#)]
SRF (0h tamoxifen) [[UCSC BedGraph](#)] [[Alignment BED](#)] (mm8) [[FASTQ](#)]
SRF (24h tamoxifen) [[UCSC BedGraph](#)] [[Alignment BED](#)] (mm8) (for wild type PU.1 see above) [[FASTQ](#)]

Data from the [Murre Lab](#)

Yin et al. - E2A cistrome in B cells

Data below from:

Lin YC, Jhunjhunwala S, Benner C, Heinz S, Welinder E, Mansson R, Sigvardsson M, Hagman J, Espinoza CA, Dutkowski J, Ideker T, Glass CK, Murre C. [A global network of transcription factors, involving E2A, EBF1 and FOXO1, that orchestrates the B cell fate](#). Nat Immunol. 2010 Jul;11(7):635-43.

Link to the Raw Data: [GSE21978](#)

pre-proB Cells (EBF^{-/-})

E2A [[UCSC BedGraph](#)] [[Alignment BED](#)] (mm8)
H3K4me1 [[UCSC BedGraph](#)] [[Alignment BED](#)] (mm8)
H3K4me2 [[UCSC BedGraph](#)] [[Alignment BED](#)] (mm8)

H3K4me3 [[UCSC BedGraph](#)] [[Alignment BED](#)] (mm8)
H3K27me3 [[UCSC BedGraph](#)] [[Alignment BED](#)] (mm8)
H3K9ac/H3K14ac [[UCSC BedGraph](#)] [[Alignment BED](#)] (mm8)

proB Cells (RAG1-/-)

E2A [[UCSC BedGraph](#)] [[Alignment BED](#)] (mm8)
EBF [[UCSC BedGraph](#)] [[Alignment BED](#)] (mm8)
CTCF [[UCSC BedGraph](#)] [[Alignment BED](#)] (mm8)
Foxo1 [[UCSC BedGraph](#)] [[Alignment BED](#)] (mm8)
H3K4me1 [[UCSC BedGraph](#)] [[Alignment BED](#)] (mm8)
H3K4me2 [[UCSC BedGraph](#)] [[Alignment BED](#)] (mm8)
H3K4me3 [[UCSC BedGraph](#)] [[Alignment BED](#)] (mm8)
H3K27me3 [[UCSC BedGraph](#)] [[Alignment BED](#)] (mm8)
H3K9ac/H3K14ac [[UCSC BedGraph](#)] [[Alignment BED](#)] (mm8)

pre-proB Cells (E2A-/-) [Reconstituted with either wild-type E2A(E47) or mutant(bHLH)]

E2A (E47ER, 1h after tamoxifen treatment) [[UCSC BedGraph](#)] [[Alignment BED](#)] (mm8)
E2A (E47ER, 6h after tamoxifen treatment) [[UCSC BedGraph](#)] [[Alignment BED](#)] (mm8)
H3K4me1 (E47ER, 1h after tamoxifen treatment) [[UCSC BedGraph](#)] [[Alignment BED](#)] (mm8)
H3K4me1 (E47ER, 6h after tamoxifen treatment) [[UCSC BedGraph](#)] [[Alignment BED](#)] (mm8)
H3K4me1 (bHLHER, 6h after tamoxifen treatment) [[UCSC BedGraph](#)] [[Alignment BED](#)] (mm8)

A12 T Cells (E2A-/-, double positive) [Reconstituted with either wild-type E2A(E47) or mutant(bHLH)]

E2A (E47ER, 6h after tamoxifen treatment) [[UCSC BedGraph](#)] [[Alignment BED](#)] (mm8)
H3K4me1 (E47ER, 6h after tamoxifen treatment) [[UCSC BedGraph](#)] [[Alignment BED](#)] (mm8)
H3K4me1 (bHLHER, 6h after tamoxifen treatment) [[UCSC BedGraph](#)] [[Alignment BED](#)] (mm8)

Lymphocyte Inputs

Naive B cell Input [[UCSC BedGraph](#)] [[Alignment BED](#)] (mm8)
Lymphoid Cells [[UCSC BedGraph](#)] [[Alignment BED](#)] (mm8)

Data from the [Katzenellenbogen Lab](#)

Stender et al. - Estrogen Receptor (ERa) DNA binding mutant cistrome

Data below from:

Stender JD, Kim K, Charn TH, Komm B, Chang KC, Kraus WL, Benner C, Glass CK, Katzenellenbogen BS. [Genome-Wide Analysis of Estrogen Receptor- \$\alpha\$ DNA Binding and Tethering Mechanisms Identifies Runx1 as a Novel Tethering Factor in Receptor-Mediated Transcriptional Activation](#). Mol Cell Biol. 2010 Jun 14.

Link to the raw data: [GSE22610](#)

MBA-MB-231 (Human Breast Cancer ER-negative Cell Line) [Stables with either wild-type ERa or ERa with a mutated DNA-binding domain]

ERa (Estradiol) [[UCSC BedGraph](#)] [[Alignment BED](#)] (hg18)
ERa-mutant (Estradiol) [[UCSC BedGraph](#)] [[Alignment BED](#)] (hg18)



Can't figure something out? Questions, comments, concerns, or other feedback:
cbenner@ucsd.edu

track type=bigBed name="HOMER Known Motifs (mm9)" description="HOMER Known Motifs (mm9)"
bigDataUrl=http://biowhat.ucsd.edu/bigWig/homer.Known.mm9.bigbed visibility=3

track type=bigBed name="HOMER Known Motifs (hg18)" description="HOMER Known Motifs (hg18)"
bigDataUrl=http://biowhat.ucsd.edu/bigWig/homer.Known.hg18.bigbed visibility=3

track type=bigBed name="HOMER Known Motifs (hg19)" description="HOMER Known Motifs (hg19)"
bigDataUrl=http://biowhat.ucsd.edu/bigWig/homer.Known.hg19.bigbed visibility=3

>GATTGCATCA	AARE(HLH)/mES-cMyc-ChIP-Seq/Homer	8.228816	-5.216482e+02	0
20001.0,5819.0,277.2,249.0,2.83e-227				
0.095	0.079	0.734	0.092	
0.735	0.071	0.173	0.021	
0.001	0.001	0.008	0.990	
0.001	0.001	0.001	0.997	
0.053	0.001	0.891	0.056	
0.001	0.997	0.001	0.001	
0.956	0.001	0.042	0.001	
0.002	0.001	0.001	0.996	
0.001	0.983	0.008	0.008	
0.997	0.001	0.001	0.001	
>ATGACTCATC	AP-1(bZIP)/ThioMac-PU.1-ChIP-Seq/Homer	6.049537	-1.782996e+03	0
9805.3,5781.0,3085.1,2715.0,0.00e+00				
0.419	0.275	0.277	0.028	
0.001	0.001	0.001	0.997	
0.010	0.002	0.965	0.023	
0.984	0.003	0.001	0.012	
0.062	0.579	0.305	0.054	
0.026	0.001	0.001	0.972	
0.043	0.943	0.001	0.012	
0.980	0.005	0.001	0.014	
0.050	0.172	0.307	0.471	
0.149	0.444	0.211	0.195	
>NNTGCCCTCAGGGCA	AP2gamma(AP2)/MCF7-TFAP2c-ChIP-Seq/Homer	6.761322	-1.120409e+04	
0 50000.2,17714.0,11304.9,9725.0,0.00e+00				
0.227	0.338	0.176	0.259	
0.343656343656344	0.27972027972028	0.170829170829171	0.205794205794206	
0.167	0.154	0.119	0.56	
0.183	0.061	0.494	0.262	
0.006	0.503	0.461	0.03	
0.001	0.997	0.001	0.001	
0.001	0.9	0.001	0.098	
0.03003003003003	0.273273273273273	0.014014014014014	0.682682682682683	
0.078	0.449	0.398	0.075	
0.72	0.012	0.23	0.038	
0.088	0.001	0.91	0.001	
0.001	0.001	0.997	0.001	
0.023	0.458	0.511	0.008	
0.251	0.498	0.065	0.186	
0.548	0.141	0.143	0.168	
>ATGCCCTGAGGC	AP-2alpha(AP2)/Hela-AP2alpha-ChIP-Seq/Homer	6.800669	-1.029965e+04	0
50001.0,15428.0,20149.5,15273.0,0.00e+00				
0.408	0.209	0.161	0.221	
0.153	0.088	0.111	0.647	
0.166	0.063	0.596	0.175	
0.008	0.636	0.348	0.008	
0.001	0.997	0.001	0.001	
0.001	0.941	0.001	0.057	
0.020	0.184	0.018	0.778	
0.099	0.353	0.506	0.042	
0.683	0.009	0.302	0.006	
0.028	0.001	0.970	0.001	

0.001	0.001	0.997	0.001			
0.014	0.607	0.368	0.011			
>AGTAAACAAAAAAGAACANA				FOXA1:AR/LNCAP-AR-ChIP-Seq/Homer	10.603910	-8.019073e+02
0	50000.0,1426.0,863.3,387.0,0.00e+00					
0.571	0.008	0.138	0.283			
0.343	0.028	0.598	0.031			
0.023	0.405	0.001	0.571			
0.951	0.047	0.001	0.001			
0.977	0.021	0.001	0.001			
0.997	0.001	0.001	0.001			
0.001	0.664	0.003	0.333			
0.946	0.003	0.010	0.041			
0.498	0.164	0.182	0.156			
0.504	0.164	0.177	0.156			
0.509	0.166	0.179	0.146			
0.612	0.107	0.114	0.166			
0.672	0.001	0.270	0.057			
0.151	0.001	0.778	0.070			
0.483	0.153	0.099	0.265			
0.968	0.001	0.005	0.026			
0.003	0.991	0.005	0.001			
0.887	0.001	0.001	0.111			
0.254	0.213	0.221	0.312			
0.353	0.088	0.234	0.324			
>AGAACAGNCTGTTCTT				ARE(NR)/LNCAP-AR-ChIP-Seq/Homer	7.778994	-1.055434e+04 0
50001.0,9390.0,9296.7,7639.0,0.00e+00						
0.467	0.001	0.427	0.104			
0.054	0.001	0.923	0.022			
0.390	0.201	0.276	0.133			
0.890	0.026	0.039	0.045			
0.001	0.988	0.010	0.001			
0.832	0.001	0.043	0.124			
0.096	0.291	0.414	0.199			
0.262	0.233	0.243	0.262			
0.203	0.409	0.292	0.096			
0.123	0.046	0.001	0.830			
0.001	0.011	0.987	0.001			
0.047	0.042	0.018	0.892			
0.127	0.268	0.198	0.407			
0.011	0.944	0.001	0.044			
0.102	0.411	0.001	0.485			
0.144	0.261	0.240	0.356			
>CCAGGAACAG				AR-halfsite(NR)/LNCaP-AR-ChIP-Seq/Homer	5.193832	-2.924684e+03 0
85575.8,13548.0,47304.5,15468.0,0.00e+00						
0.122	0.483	0.277	0.118			
0.204	0.613	0.009	0.174			
0.439	0.124	0.187	0.250			
0.335	0.001	0.663	0.001			
0.001	0.001	0.997	0.001			
0.471	0.273	0.168	0.089			
0.997	0.001	0.001	0.001			
0.001	0.994	0.004	0.001			
0.965	0.001	0.001	0.033			
0.031	0.331	0.533	0.104			

>GGGTCACGTGAC ATF3(bZIP)/K562-ATF3-ChIP-Seq/Homer 7.571939 -2.427874e+03 0
50002.0,3382.0,3325.3,1744.0,0.00e+00
0.140 0.323 0.399 0.139
0.146 0.257 0.488 0.109
0.175 0.184 0.640 0.001
0.040 0.235 0.072 0.653
0.001 0.997 0.001 0.001
0.997 0.001 0.001 0.001
0.001 0.944 0.008 0.047
0.048 0.008 0.943 0.001
0.001 0.001 0.001 0.997
0.001 0.001 0.997 0.001
0.632 0.073 0.252 0.043
0.037 0.621 0.172 0.170

>VNAV CAGCTGGC Atoh1(bHLH)/Cerebellum-Atoh1-ChIP-Seq/Homer 6.030696 -5502.642506 0
T:3441.0(59.72%),B:1043.2(2.36%),P:1e-2386 Tpos:-0.2,Tstd:10.8,Bpos:-0.1,Bstd:12.6,StrandBias:-
0.0,Multiplicity:1.07
0.296 0.237 0.323 0.146
0.209 0.237 0.271 0.285
0.488 0.038 0.468 0.008
0.280 0.420 0.300 0.002
0.002 0.995 0.002 0.002
0.995 0.002 0.002 0.002
0.002 0.002 0.719 0.278
0.381 0.616 0.002 0.002
0.002 0.002 0.002 0.995
0.002 0.002 0.995 0.002
0.002 0.269 0.500 0.230
0.036 0.461 0.058 0.446

>NNNCTTTCCAGGAAA Bcl6(Zf)/Liver-Bcl6-ChIP-Seq(GSE31578)/Homer 6.522635 -1780.142566 0
T:3881.0(16.65%),B:1480.6(5.67%),P:1e-773 Tpos:50.2,Tstd:23.5,Bpos:48.5,Bstd:27.3,StrandBias:-
0.0,Multiplicity:1.06
0.225 0.277 0.206 0.292
0.271 0.206 0.219 0.304
0.221 0.218 0.348 0.213
0.218 0.390 0.164 0.228
0.120 0.234 0.061 0.585
0.032 0.207 0.052 0.709
0.027 0.058 0.031 0.884
0.017 0.888 0.023 0.072
0.025 0.648 0.012 0.315
0.623 0.009 0.025 0.343
0.116 0.013 0.842 0.029
0.065 0.033 0.867 0.035
0.788 0.048 0.136 0.028
0.867 0.041 0.053 0.039
0.412 0.175 0.222 0.190

>NGGCACGTGC bHLHE40(HLH)/HepG2-BHLHE40-ChIP-Seq/Homer 5.056879 -8.811631e+02 0
50001.9,419.0,676.2,340.0,0.00e+00
0.214214214214214 0.325325325325325 0.385385385385385 0.0750750750750751
0.175 0.181 0.403 0.241
0.025025025025025 0.001001001001001 0.524524524524524 0.449449449449449
0.001 0.997 0.001 0.001
0.997 0.001 0.001 0.001

0.001	0.997	0.001	0.001				
0.001	0.001	0.997	0.001				
0.001	0.001	0.001	0.997				
0.001	0.001	0.997	0.001				
0.444	0.519	0.001	0.036				
>NHAACBGYYV BMYB(HTH)/Hela-BMYB-ChIPSeq(GSE27030)/Homer				6.194378	-662.008225	0	
T:2008.0(34.61%),B:5932.4(15.30%),P:1e-287							
0.211	0.282	0.234	0.273				
0.310	0.274	0.014	0.402				
0.991	0.001	0.001	0.007				
0.997	0.001	0.001	0.001				
0.008	0.990	0.001	0.001				
0.129	0.245	0.303	0.322				
0.001	0.001	0.997	0.001				
0.209	0.386	0.066	0.339				
0.161	0.432	0.004	0.403				
0.317	0.234	0.300	0.150				
>CNNBRGCGCCCCCTGSTGGC BORIS(Zf)/K562-CTCFL-ChIP-Seq/Homer				9.032880	-28414.037074	0	
T:12097.0(39.79%),B:538.5(1.81%),P:1e-12339							
Tpos:50.2,Tstd:15.4,Bpos:51.2,Bstd:24.7,StrandBias:0.0,Multiplicity:1.02							
0.173	0.416	0.246	0.165				
0.226	0.275	0.229	0.270				
0.288	0.299	0.259	0.154				
0.105	0.320	0.268	0.307				
0.334	0.114	0.416	0.136				
0.060	0.326	0.554	0.060				
0.052	0.562	0.052	0.334				
0.037	0.005	0.827	0.131				
0.023	0.960	0.013	0.004				
0.001	0.997	0.001	0.001				
0.245	0.670	0.027	0.058				
0.001	0.689	0.001	0.309				
0.001	0.997	0.001	0.001				
0.050	0.043	0.017	0.890				
0.253	0.073	0.525	0.149				
0.004	0.418	0.546	0.032				
0.172	0.150	0.055	0.623				
0.001	0.001	0.997	0.001				
0.019	0.063	0.865	0.053				
0.192	0.432	0.150	0.226				
>GTCATAAAAN Cdx2(Homeobox)/mES-Cdx2-ChIP-Seq/Homer				6.986576	-7.212192e+03	0	
50001.0,7166.0,11890.3,7880.0,0.00e+00							
0.209	0.058	0.733	0.001				
0.001	0.444	0.001	0.554				
0.489	0.493	0.001	0.017				
0.997	0.001	0.001	0.001				
0.001	0.001	0.001	0.997				
0.997	0.001	0.001	0.001				
0.997	0.001	0.001	0.001				
0.997	0.001	0.001	0.001				
0.678	0.115	0.073	0.135				
0.243	0.285	0.154	0.318				
>NATGTTGCAA CEBP:AP1/ThioMac-CEBPb-ChIP-Seq/Homer				6.428620	-5.728655e+03	0	
71889.0,33959.0,12413.6,9808.0,0.00e+00							

0.290	0.093	0.336	0.281			
0.464	0.160	0.356	0.020			
0.016	0.001	0.006	0.977			
0.045	0.001	0.921	0.033			
0.378	0.003	0.001	0.618			
0.081	0.049	0.117	0.752			
0.004	0.001	0.994	0.001			
0.063	0.867	0.001	0.069			
0.997	0.001	0.001	0.001			
0.986	0.009	0.001	0.003			
>ATTGCGCAAC CEBP(bZIP)/CEBPb-ChIP-Seq/Homer 6.128409 -3.226308e+04 0						
130502.0,65251.0,37208.0,33247.0,0.00e+00						
0.812	0.102	0.061	0.026			
0.001	0.003	0.001	0.995			
0.001	0.001	0.026	0.972			
0.104	0.001	0.795	0.100			
0.076	0.624	0.055	0.244			
0.192	0.061	0.644	0.104			
0.102	0.757	0.003	0.138			
0.971	0.027	0.001	0.001			
0.993	0.001	0.005	0.001			
0.037	0.664	0.143	0.157			
>CGGTTTCAAA CHR/Cell-Cycle-Exp/Homer 7.763324 -8.207362e+01 0						
20264.3,437.0,1292.0,122.0,2.27e-36						
0.233	0.453	0.313	0.001			
0.408	0.051	0.540	0.001			
0.172	0.051	0.582	0.195			
0.283	0.073	0.035	0.609			
0.001	0.012	0.001	0.986			
0.012	0.001	0.001	0.986			
0.001	0.679	0.185	0.135			
0.997	0.001	0.001	0.001			
0.997	0.001	0.001	0.001			
0.986	0.012	0.001	0.001			
>NCCACGTG c-Myc(HLH)/LNCAP-cMyc-ChIP-Seq/Homer 6.613495 -1.042210e+03 0						
50000.0,1465.0,6122.5,1251.0,0.00e+00						
0.327	0.218	0.319	0.135			
0.115	0.600	0.224	0.061			
0.022	0.950	0.017	0.012			
0.890	0.001	0.094	0.015			
0.001	0.996	0.001	0.002			
0.001	0.001	0.997	0.001			
0.002	0.080	0.001	0.917			
0.013	0.008	0.953	0.026			
>GNCCACGTGG c-Myc/mES-cMyc-ChIP-Seq/Homer 6.808840 -2.997760e+03 0						
50000.9,5819.0,7652.0,3705.0,0.00e+00						
0.251	0.225	0.369	0.155			
0.311	0.197	0.336	0.156			
0.138	0.504	0.336	0.022			
0.001	0.997	0.001	0.001			
0.992	0.001	0.006	0.001			
0.001	0.932	0.001	0.066			
0.112	0.001	0.886	0.001			
0.001	0.001	0.001	0.997			

0.001	0.001	0.997	0.001				
0.023	0.320	0.526	0.130				
>CGGTGACGTCAC CRE(bZIP)/Promoter/Homer				5.918254	-9.269630e+02	0	
124715.9,9565.0,6809.3,1652.0,0.00e+00							
0.089	0.490	0.225	0.196				
0.077	0.378	0.437	0.109				
0.067	0.308	0.604	0.021				
0.001	0.030	0.001	0.968				
0.001	0.203	0.795	0.001				
0.981	0.001	0.017	0.001				
0.001	0.976	0.001	0.022				
0.023	0.001	0.975	0.001				
0.001	0.027	0.001	0.971				
0.001	0.800	0.198	0.001				
0.977	0.001	0.021	0.001				
0.005	0.649	0.281	0.065				
>GCTAATCC CRX(Homeobox)/Retina-Crx-ChIP-Seq/Homer				4.852430	-2.388287e+03	0	
49999.6,3506.0,4648.4,2130.0,0.00e+00							
0.148	0.245	0.445	0.162				
0.132	0.528	0.062	0.278				
0.022	0.004	0.001	0.973				
0.890	0.003	0.023	0.084				
0.975	0.011	0.006	0.008				
0.001	0.006	0.183	0.811				
0.001	0.978	0.016	0.005				
0.012	0.754	0.013	0.222				
>ATAGTGCCACCTGGTGGCCA CTCF(Zf)/CD4+-CTCF-ChIP-Seq/Homer				8.704837	-6.281855e+03	0	
15000.0,4645.0,2877.2,2765.0,0.00e+00							
0.447	0.221	0.181	0.151				
0.037	0.340	0.236	0.386				
0.499	0.061	0.330	0.110				
0.033	0.377	0.528	0.061				
0.023	0.378	0.005	0.593				
0.061	0.005	0.887	0.047				
0.079	0.905	0.005	0.010				
0.002	0.994	0.001	0.003				
0.501	0.475	0.007	0.016				
0.002	0.527	0.004	0.467				
0.003	0.995	0.001	0.001				
0.030	0.036	0.004	0.930				
0.382	0.042	0.446	0.130				
0.020	0.273	0.686	0.021				
0.047	0.039	0.014	0.900				
0.002	0.001	0.995	0.002				
0.040	0.034	0.873	0.053				
0.161	0.527	0.061	0.250				
0.277	0.428	0.117	0.178				
0.541	0.092	0.267	0.100				
>TGCAGTTCCAANAGTGGCCA CTCF-SatelliteElement/CD4+-CTCF-ChIP-Seq/Homer				11.159698	-		
1.535293e+03 0 15000.0,4550.0,350.4,347.0,0.00e+00							
0.051	0.010	0.033	0.905				
0.003	0.005	0.991	0.001				
0.015	0.930	0.001	0.054				
0.935	0.008	0.001	0.056				

0.203	0.064	0.707	0.026
0.023	0.023	0.018	0.936
0.244	0.041	0.165	0.550
0.049	0.905	0.013	0.033
0.090	0.738	0.005	0.167
0.373	0.329	0.136	0.162
0.380	0.254	0.260	0.105
0.303	0.314	0.208	0.175
0.398	0.149	0.152	0.301
0.293	0.157	0.378	0.172
0.051	0.319	0.044	0.586
0.275	0.057	0.517	0.152
0.244	0.041	0.638	0.077
0.041	0.951	0.003	0.005
0.001	0.995	0.001	0.003
0.928	0.003	0.015	0.054

>ACTYKNATTCGTGNTACTTC Mouse_Recombination_Hotspot/Testis-DMC1-ChIP-Seq/Homer 9.682136 -
4505.162225 0 T:1398.0(34.48%),B:285.0(0.62%),P:1e-1956
Tpos:99.2,Tstd:46.8,Bpos:105.3,Bstd:54.4,StrandBias:-0.1,Multiplicity:1.01

0.482	0.109	0.229	0.180
0.257	0.535	0.088	0.120
0.107	0.256	0.001	0.636
0.036	0.527	0.055	0.382
0.225	0.156	0.238	0.382
0.331	0.287	0.197	0.185
0.676	0.080	0.162	0.082
0.037	0.146	0.314	0.503
0.005	0.010	0.006	0.979
0.009	0.613	0.001	0.377
0.199	0.170	0.438	0.193
0.319	0.001	0.057	0.623
0.001	0.001	0.997	0.001
0.212	0.272	0.262	0.255
0.030	0.053	0.018	0.899
0.559	0.026	0.372	0.043
0.008	0.964	0.002	0.026
0.032	0.075	0.001	0.892
0.030	0.071	0.030	0.869
0.250	0.427	0.060	0.263

>ANACAGCTGC E2A(HLH)/proBcell-E2A-ChIP-Seq/Homer 6.427207 -6.404287e+03 0
91747.1,10405.0,32260.7,13392.0,0.00e+00

0.352	0.171	0.238	0.239
0.292	0.294	0.253	0.161
0.428	0.218	0.353	0.001
0.001	0.997	0.001	0.001
0.997	0.001	0.001	0.001
0.001	0.238	0.753	0.008
0.001	0.871	0.127	0.001
0.001	0.001	0.001	0.997
0.001	0.001	0.997	0.001
0.007	0.476	0.178	0.339

>CWGGCGGGAA E2F/Hela-E2F1-ChIP-Seq/Hoemr 7.531299 -346.110499 0
T:208.0(17.85%),B:657.0(1.49%),P:1e-150 Tpos:24.4,Tstd:9.9,Bpos:25.2,Bstd:16.2,StrandBias:-0.2,Multiplicity:1.04
0.213 0.424 0.224 0.141

0.257	0.158	0.210	0.377
0.062	0.147	0.790	0.002
0.002	0.002	0.995	0.002
0.002	0.995	0.002	0.002
0.002	0.002	0.995	0.002
0.002	0.080	0.917	0.002
0.002	0.015	0.982	0.002
0.995	0.002	0.002	0.002
0.949	0.002	0.048	0.002
>VDTTTCCTCCGCCA E2F7(E2F)/Hela-E2F7-ChIP-Seq(GSE32673)/Homer 8.690744 -482.887939 0			
T:238.0(34.64%),B:844.9(2.16%),P:1e-209 Tpos:102.2,Tstd:51.1,Bpos:102.9,Bstd:91.2,StrandBias:-			
0.3,Multiplicity:1.11			
0.208	0.350	0.276	0.166
0.290	0.171	0.215	0.323
0.104	0.105	0.179	0.612
0.001	0.152	0.001	0.846
0.001	0.037	0.001	0.961
0.001	0.997	0.001	0.001
0.001	0.730	0.268	0.001
0.001	0.997	0.001	0.001
0.001	0.001	0.987	0.011
0.001	0.978	0.020	0.001
0.001	0.653	0.284	0.062
0.410	0.222	0.140	0.228
>TTCGCGCGAAAA E2F(E2F)/Cell-Cycle-Exp/Homer 8.486500 -4.535971e+01 0			
20264.3,437.0,727.4,66.0,2.00e-20			
0.035	0.275	0.114	0.576
0.001	0.169	0.106	0.724
0.035	0.538	0.427	0.001
0.001	0.274	0.724	0.001
0.001	0.997	0.001	0.001
0.001	0.001	0.997	0.001
0.001	0.598	0.400	0.001
0.001	0.239	0.759	0.001
0.882	0.050	0.051	0.017
0.929	0.001	0.069	0.001
0.917	0.017	0.065	0.001
0.530	0.084	0.264	0.122
>NNCACCTGCN E2A-nearPU.1(HLH)/Bcell-PU.1-ChIP-Seq/Homer 7.417520 -1.383219e+03 0			
9675.5,3971.0,1336.4,1163.0,0.00e+00			
0.218	0.303	0.259	0.220
0.316	0.261	0.257	0.166
0.001	0.997	0.001	0.001
0.997	0.001	0.001	0.001
0.001	0.997	0.001	0.001
0.001	0.997	0.001	0.001
0.001	0.001	0.001	0.997
0.001	0.001	0.997	0.001
0.135	0.370	0.251	0.244
0.279	0.268	0.213	0.240
>GGTCCCTAGGGA EBF(E2F)/proBcell-EBF-ChIP-Seq/Homer 8.590014 -2.594541e+03 0			
50001.0,1220.0,2937.3,1638.0,0.00e+00			
0.287	0.122	0.352	0.240
0.076	0.283	0.458	0.182

0.005	0.160	0.001	0.833				
0.001	0.997	0.001	0.001				
0.001	0.845	0.001	0.153				
0.001	0.997	0.001	0.001				
0.227	0.348	0.032	0.393				
0.396	0.024	0.347	0.233				
0.001	0.001	0.997	0.001				
0.117	0.001	0.881	0.001				
0.001	0.001	0.997	0.001				
0.849	0.001	0.133	0.017				
>GTCCCCAGGGGA EBF1(EBF)/Near-E2A-ChIP-Seq/Homer				6.752028	-1.219094e+03	0	
91747.1,10405.0,15735.7,4045.0,0.00e+00							
0.184	0.170	0.443	0.203				
0.067	0.170	0.263	0.501				
0.021	0.618	0.075	0.286				
0.012	0.930	0.001	0.057				
0.021	0.862	0.001	0.116				
0.038	0.917	0.001	0.044				
0.538	0.055	0.003	0.405				
0.146	0.006	0.770	0.078				
0.179	0.001	0.794	0.026				
0.019	0.001	0.979	0.001				
0.210	0.045	0.724	0.021				
0.500	0.204	0.242	0.054				
>GGYAGCAYDTGCTDCCNNN EBNA1(EBV virus)/Raji-EBNA1-ChIP-Seq(GSE30709)/Homer				11.187041			
-3519.754114 0 T:587.0(49.54%),B:0.7(0.04%),P:1e-1528 Tpos:102.4,Tstd:23.7,Bpos:45.5,Bstd:2.0,StrandBias:-0.0,Multiplicity:1.01							
0.001	0.001	0.997	0.001				
0.235	0.001	0.763	0.001				
0.218	0.390	0.003	0.389				
0.997	0.001	0.001	0.001				
0.011	0.001	0.987	0.001				
0.007	0.623	0.144	0.226				
0.691	0.174	0.110	0.025				
0.181	0.282	0.107	0.430				
0.377	0.122	0.238	0.263				
0.037	0.124	0.130	0.709				
0.246	0.196	0.550	0.008				
0.001	0.976	0.001	0.022				
0.001	0.001	0.001	0.997				
0.383	0.001	0.363	0.253				
0.001	0.760	0.001	0.238				
0.001	0.997	0.001	0.001				
0.181	0.555	0.033	0.231				
0.286	0.198	0.231	0.285				
0.245	0.224	0.337	0.194				
0.248	0.261	0.268	0.223				
>CCGGTCACGTGA E-box(HLH)/Promoter/Homer				8.816220	-5.659171e+02	0	
149303.9,11385.0,3045.4,815.0,0.00e+00							
0.050	0.498	0.381	0.070				
0.104	0.417	0.369	0.110				
0.070	0.234	0.603	0.094				
0.073	0.122	0.795	0.010				
0.008	0.127	0.030	0.835				

0.001	0.997	0.001	0.001				
0.990	0.004	0.002	0.005				
0.002	0.931	0.003	0.064				
0.105	0.022	0.872	0.001				
0.002	0.002	0.001	0.995				
0.001	0.001	0.997	0.001				
0.726	0.047	0.223	0.004				
>TCCGCCACGCA EGR(Zf)/K562-EGR1-ChIP-Seq/Homer				8.525436	-1.079903e+03	0	
50000.0,933.0,735.3,414.0,0.00e+00							
0.116	0.262	0.171	0.451				
0.1111111111111111	0.84984984984985	0.016016016016016	0.023023023023023				
0.022	0.951	0.001	0.026				
0.207	0.006	0.667	0.12				
0.001	0.997	0.001	0.001				
0.001	0.992	0.001	0.006				
0.00599400599400599	0.982017982017982	0.000999000999000999	0.010989010989011				
0.874	0.118	0.007	0.001				
0.001	0.997	0.001	0.001				
0.104	0.008	0.816	0.072				
0.000999000999000999	0.955044955044955	0.010989010989011	0.032967032967033				
0.685	0.181	0.054	0.08				
>NGCGTGGGCGGR Egr2/Thymocytes-Egr2-ChIP-Seq(GSE34254)/Homer				9.168701	-1390.042185	0	
T:776.0(22.95%),B:766.2(1.70%),P:1e-603							
0.216	0.231	0.231	0.322				
0.114	0.182	0.678	0.026				
0.158	0.552	0.035	0.255				
0.001	0.001	0.997	0.001				
0.001	0.001	0.137	0.861				
0.055	0.001	0.943	0.001				
0.001	0.001	0.997	0.001				
0.001	0.001	0.997	0.001				
0.159	0.587	0.001	0.253				
0.032	0.001	0.953	0.014				
0.092	0.066	0.513	0.329				
0.248	0.192	0.357	0.204				
>NTGGGTGTGGCC EKLF(Zf)/Erythrocyte-Klf1-ChIP-Seq/Homer				9.250875	-4.799045e+02	0	
50000.5,662.0,1049.8,258.0,3.80e-209							
0.268	0.306	0.230	0.195				
0.402	0.001	0.051	0.546				
0.001	0.001	0.997	0.001				
0.001	0.001	0.997	0.001				
0.001	0.001	0.997	0.001				
0.035	0.049	0.001	0.915				
0.001	0.001	0.997	0.001				
0.001	0.001	0.155	0.843				
0.001	0.001	0.997	0.001				
0.047	0.054	0.781	0.117				
0.039	0.630	0.024	0.308				
0.168	0.411	0.067	0.354				
>AACCGGAAGT ELF1(ETS)/Jurkat-ELF1-ChIP-Seq/Homer				6.819311	-3.928513e+03	0	
49999.4,4425.0,7422.2,3925.0,0.00e+00							
0.494	0.156	0.189	0.161				
0.358	0.289	0.249	0.105				
0.058	0.717	0.201	0.024				

0.092	0.889	0.011	0.008				
0.007	0.012	0.979	0.002				
0.012	0.011	0.969	0.008				
0.961	0.008	0.016	0.015				
0.960	0.014	0.008	0.018				
0.085	0.062	0.833	0.020				
0.099	0.241	0.049	0.611				
>AGGTGTTAAN				Eomes(T-box)/H9-Eomes-ChIP-Seq/Homer	5.080208	-7.056207e+03	0
68321.0,34160.0,16134.4,12976.0,0.00e+00							
0.573	0.094	0.198	0.135				
0.238761238761239		0.015984015984016		0.613386613386613		0.131868131868132	
0.002	0.202	0.777	0.019				
0.051	0.002	0.002	0.945				
0.002	0.001	0.996	0.001				
0.333	0.108	0.007	0.552				
0.024	0.057	0.366	0.553				
0.983	0.001	0.012	0.004				
0.79	0.026	0.104	0.08				
0.187	0.247	0.251	0.315				
>AAGGTCACNGTGACC				ERE(NR/IR3)/MCF7-ERa-ChIP-Seq/Homer	7.666577	-1.551930e+04	0
29999.0,13064.0,9699.8,9266.0,0.00e+00							
0.353	0.257	0.221	0.168				
0.664	0.016	0.268	0.052				
0.060	0.001	0.801	0.138				
0.028	0.004	0.950	0.018				
0.057	0.033	0.154	0.755				
0.001	0.928	0.031	0.040				
0.934	0.007	0.026	0.033				
0.144	0.421	0.262	0.173				
0.219	0.308	0.251	0.222				
0.166	0.283	0.387	0.164				
0.037	0.015	0.007	0.941				
0.042	0.024	0.933	0.001				
0.729	0.164	0.038	0.069				
0.022	0.934	0.005	0.040				
0.158	0.763	0.001	0.078				
>ACAGGAAGTG				ERG(ETS)/VCaP-ERG-ChIP-Seq/Homer	6.433411	-8.637491e+03	0
102752.0,51376.0,33886.2,25441.0,0.00e+00							
0.536	0.119	0.250	0.096				
0.050	0.877	0.072	0.001				
0.810	0.188	0.001	0.001				
0.001	0.001	0.997	0.001				
0.001	0.001	0.997	0.001				
0.997	0.001	0.001	0.001				
0.964	0.001	0.001	0.034				
0.330	0.001	0.668	0.001				
0.034	0.272	0.062	0.631				
0.267	0.160	0.437	0.136				
>CAAAGGTCAG				Erra(NR)/HepG2-Erra-ChIP-Seq/Homer	5.749423	-3.265720e+03	0
51812.0,25906.0,19655.6,13972.0,0.00e+00							
0.023	0.791	0.047	0.139				
0.604	0.358	0.028	0.011				
0.682	0.021	0.288	0.009				
0.877	0.001	0.118	0.004				

0.001	0.001	0.997	0.001				
0.015	0.011	0.814	0.159				
0.025	0.139	0.176	0.660				
0.001	0.985	0.010	0.004				
0.971	0.013	0.008	0.008				
0.149	0.267	0.491	0.094				
>GTGACCTTGA Esrrb(NR)/mES-Esrrb-ChIP-Seq/Homer				7.149281	-4.195354e+04	0	
110598.0,55299.0,40077.5,37002.0,0.00e+00							
0.202	0.185	0.376	0.237				
0.010	0.072	0.012	0.906				
0.032	0.070	0.894	0.005				
0.865	0.070	0.015	0.050				
0.014	0.969	0.008	0.009				
0.010	0.961	0.024	0.005				
0.009	0.032	0.023	0.936				
0.014	0.144	0.012	0.830				
0.038	0.196	0.611	0.155				
0.483	0.135	0.295	0.087				
>ACAGGAAGTG ETS1(ETS)/Jurkat-ETS1-ChIP-Seq/Homer				6.525370	-6.554553e+03	0	
49998.8,18212.0,13876.1,10375.0,0.00e+00							
0.540	0.088	0.332	0.041				
0.093	0.721	0.181	0.004				
0.719	0.272	0.008	0.001				
0.001	0.001	0.997	0.001				
0.001	0.001	0.997	0.001				
0.997	0.001	0.001	0.001				
0.869	0.001	0.001	0.129				
0.116	0.001	0.882	0.001				
0.024	0.137	0.001	0.838				
0.112	0.128	0.654	0.106				
>AACAGGAAGT Ets1-distal(ETS)/CD4+-PolII-ChIP-Seq/Homer				8.637957	-1.344900e+02	0	
50000.7,930.0,4299.8,274.0,0.00e+00							
0.453	0.329	0.213	0.005				
0.864	0.001	0.079	0.056				
0.001	0.859	0.093	0.047				
0.967	0.031	0.001	0.001				
0.001	0.001	0.997	0.001				
0.001	0.001	0.997	0.001				
0.976	0.001	0.001	0.022				
0.821	0.001	0.001	0.177				
0.262	0.001	0.736	0.001				
0.002	0.254	0.030	0.713				
>AGGAAACAGCTG ETS:E-box/HPC7-Scl-ChIP-Seq/Homer				10.015277	-7.583151e+02	0	
50000.1,6776.0,793.0,511.0,0.00e+00							
0.895	0.032	0.072	0.001				
0.001	0.001	0.997	0.001				
0.001	0.001	0.997	0.001				
0.997	0.001	0.001	0.001				
0.8888888888888889	0.003003003003003	0.032032032032032	0.0760760760760761				
0.546	0.018	0.435	0.001				
0.001	0.997	0.001	0.001				
0.965	0.001	0.033	0.001				
0.014014014014014	0.209209209209209	0.572572572572573	0.204204204204204				
0.311	0.65	0.011	0.028				

0.0529470529470529	0.022977022977023	0.015984015984016	0.908091908091908
0.039039039039039	0.01001001001001	0.853853853853854	0.0970970970970971
>AACCGGAAGT ETS(ETS)/Promoter/Homer 7.147950 -1.835584e+03 0			
149303.9,11385.0,9009.1,2591.0,0.00e+00			
0.489	0.205	0.188	0.117
0.499	0.210	0.246	0.045
0.034	0.788	0.159	0.019
0.086	0.895	0.005	0.014
0.006	0.003	0.990	0.002
0.012	0.003	0.983	0.003
0.990	0.005	0.002	0.003
0.978	0.007	0.003	0.012
0.049	0.022	0.923	0.006
0.040	0.157	0.016	0.787
>ACAGGATGTGGT ETS:RUNX/Jurkat-RUNX1-ChIP-Seq/Homer 8.581039 -4.133034e+02 0			
49998.5,2030.0,601.0,239.0,3.20e-180			
0.456	0.109	0.328	0.107
0.231	0.506	0.263	0.001
0.904	0.085	0.001	0.010
0.001	0.001	0.997	0.001
0.007	0.001	0.991	0.001
0.997	0.001	0.001	0.001
0.412	0.001	0.001	0.586
0.001	0.001	0.997	0.001
0.001	0.001	0.001	0.997
0.001	0.001	0.997	0.001
0.001	0.001	0.997	0.001
0.001	0.192	0.263	0.545
>AACCGGAAGT ETV1(ETS)/GIST48-ETV1-ChIP-Seq/Homer 6.285881 -9.273229e+03 0			
50002.0,24179.0,21021.9,17157.0,0.00e+00			
0.426	0.192	0.247	0.135
0.564	0.108	0.218	0.111
0.081	0.779	0.116	0.024
0.371	0.595	0.023	0.012
0.013	0.012	0.968	0.008
0.014	0.012	0.963	0.011
0.949	0.016	0.016	0.019
0.857	0.020	0.021	0.102
0.148	0.068	0.761	0.023
0.085	0.209	0.101	0.605
>ATTCCTGTN EWS:ERG-fusion(ETS)/CADO_ES1-EWS:ERG-ChIP-Seq/Homer 7.338790 -			
3.812985e+02 0 49999.2,613.0,1122.4,221.0,2.54e-166			
0.871	0.009	0.119	0.001
0.001	0.155	0.001	0.843
0.001	0.001	0.001	0.997
0.001	0.001	0.001	0.997
0.001	0.997	0.001	0.001
0.001	0.997	0.001	0.001
0.001	0.001	0.041	0.957
0.001	0.105	0.805	0.089
0.194	0.277	0.053	0.476
0.220	0.269	0.276	0.235
>AACAGGAAAT EWS:FLI1-fusion(ETS)/SK_N_MC-EWS:FLI1-ChIP-Seq/Homer 4.875418 -			
5.041852e+02 0 49998.6,1404.0,2393.5,470.0,1.08e-219			

0.356	0.230	0.227	0.187				
0.638	0.001	0.201	0.160				
0.001	0.997	0.001	0.001				
0.802	0.196	0.001	0.001				
0.001	0.001	0.997	0.001				
0.001	0.001	0.997	0.001				
0.997	0.001	0.001	0.001				
0.997	0.001	0.001	0.001				
0.644	0.001	0.354	0.001				
0.001	0.309	0.001	0.689				
>AAAGTAAACA	FOXA1(Forkhead)/LNCAP-FOXA1-ChIP-Seq/Homer	6.337526	-1.795356e+04	0			
57926.0,28963.0,37706.3,32309.0,0.00e+00							
0.483	0.026	0.010	0.481				
0.644	0.067	0.240	0.049				
0.575	0.002	0.111	0.313				
0.128	0.001	0.870	0.001				
0.001	0.152	0.001	0.846				
0.893	0.105	0.001	0.001				
0.969	0.029	0.001	0.001				
0.997	0.001	0.001	0.001				
0.001	0.775	0.001	0.223				
0.997	0.001	0.001	0.001				
>AAAGTAAACA	FOXA1(Forkhead)/MCF7-FOXA1-ChIP-Seq/Homer	6.568115	-1.686773e+04	0			
183170.2,22729.0,52010.5,26378.0,0.00e+00							
0.498	0.005	0.001	0.496				
0.681	0.037	0.248	0.034				
0.596	0.001	0.096	0.307				
0.114	0.001	0.884	0.001				
0.001	0.153	0.001	0.845				
0.931	0.067	0.001	0.001				
0.985	0.013	0.001	0.001				
0.997	0.001	0.001	0.001				
0.001	0.714	0.001	0.284				
0.997	0.001	0.001	0.001				
>CTTGTTTACATA	Foxa2(Forkhead)/Liver-Foxa2-ChIP-Seq/Homer	6.535409	-5.346685e+03	0			
50000.0,10171.0,10002.1,6507.0,0.00e+00							
0.101	0.435	0.265	0.199				
0.154	0.323	0.146	0.377				
0.001	0.001	0.001	0.997				
0.180	0.001	0.818	0.001				
0.001	0.001	0.001	0.997				
0.001	0.001	0.043	0.955				
0.001	0.001	0.274	0.724				
0.946	0.001	0.052	0.001				
0.004	0.984	0.001	0.012				
0.441	0.135	0.001	0.423				
0.092	0.366	0.139	0.404				
0.508	0.025	0.063	0.405				
>CTGTTTAC	Foxo1(Forkhead)/RAW-Foxo1-ChIP-Seq/Homer	6.164667	-1.532727e+03	0			
50000.0,13334.0,15175.6,7019.0,0.00e+00							
0.050	0.583	0.184	0.184				
0.001	0.001	0.001	0.997				
0.001	0.001	0.997	0.001				
0.001	0.117	0.001	0.881				

0.001	0.001	0.001	0.997			
0.001	0.001	0.183	0.815			
0.769	0.175	0.001	0.055			
0.001	0.897	0.001	0.101			
>NYYTGTTTACHN FOXPI(Forkhead)/H9-FOXP1-ChIP-Seq(GSE31006)/Homer				7.727176	-1690.121227	
0 T:1106.0(5.21%),B:126.1(0.47%),P:1e-734						
0.226	0.242	0.226	0.306			
0.164	0.337	0.180	0.320			
0.046	0.385	0.230	0.339			
0.001	0.001	0.001	0.997			
0.022	0.001	0.976	0.001			
0.001	0.001	0.001	0.997			
0.001	0.001	0.001	0.997			
0.001	0.001	0.001	0.997			
0.997	0.001	0.001	0.001			
0.001	0.873	0.002	0.124			
0.282	0.312	0.099	0.308			
0.313	0.197	0.224	0.266			
>AGGTCANTGACCTN FXR(NR/IR1)/Liver-FXR-ChIP-Seq/Homer				6.873163	-1.228492e+03	0
50000.9,897.0,1067.4,538.0,0.00e+00						
0.531	0.067	0.360	0.041			
0.064	0.001	0.862	0.073			
0.025	0.013	0.835	0.127			
0.060	0.151	0.167	0.622			
0.005	0.860	0.079	0.057			
0.906	0.028	0.041	0.024			
0.262	0.223	0.260	0.255			
0.027	0.066	0.035	0.872			
0.054	0.076	0.863	0.008			
0.621	0.160	0.152	0.066			
0.128	0.836	0.013	0.023			
0.053	0.890	0.001	0.056			
0.053	0.338	0.072	0.537			
0.138	0.347	0.250	0.265			
>AACCGGAAGT GABPA(ETS)/Jurkat-GABPa-ChIP-Seq/Homer				6.581766	-6.079783e+03	0
50002.1,8665.0,12624.1,7889.0,0.00e+00						
0.379	0.214	0.240	0.167			
0.488	0.123	0.269	0.120			
0.081	0.733	0.156	0.030			
0.290	0.634	0.038	0.038			
0.034	0.029	0.915	0.022			
0.033	0.042	0.900	0.024			
0.864	0.038	0.046	0.053			
0.864	0.029	0.035	0.072			
0.136	0.069	0.756	0.038			
0.073	0.213	0.079	0.635			
>NCCTTATCTC Gata2(Zf)/K562-GATA2-ChIP-Seq/Homer				6.674828	-9.977944e+03	0
50001.0,18776.0,11604.3,9805.0,0.00e+00						
0.152	0.286	0.226	0.336			
0.021	0.386	0.244	0.350			
0.001	0.775	0.172	0.052			
0.001	0.004	0.001	0.994			
0.001	0.001	0.001	0.997			
0.997	0.001	0.001	0.001			

0.001	0.001	0.001	0.997			
0.001	0.997	0.001	0.001			
0.266	0.001	0.028	0.705			
0.067	0.380	0.379	0.175			
>AGATGKDGAGATAAG				GATA-DR4(Zf)/iTreg-Gata3-ChIP-Seq(GSE20898)/Homer	8.179760	-
371.868484	0	T:286.0(4.18%),B:198.1(0.48%),P:1e-161				
Tpos:103.5,Tstd:37.8,Bpos:101.8,Bstd:61.5,StrandBias:-0.0,Multiplicity:1.01						
0.817	0.001	0.001	0.181			
0.001	0.001	0.997	0.001			
0.981	0.017	0.001	0.001			
0.001	0.001	0.001	0.997			
0.027	0.245	0.538	0.190			
0.227	0.145	0.246	0.382			
0.301	0.127	0.336	0.236			
0.227	0.200	0.382	0.190			
0.836	0.001	0.001	0.162			
0.001	0.001	0.997	0.001			
0.997	0.001	0.001	0.001			
0.001	0.001	0.001	0.997			
0.872	0.001	0.001	0.126			
0.637	0.001	0.272	0.090			
0.236	0.244	0.429	0.090			
>AGATSTNDNDSAGATAASN				GATA-DR8(Zf)/iTreg-Gata3-ChIP-Seq(GSE20898)/Homer	9.250016	-
436.379904	0	T:290.0(4.24%),B:160.2(0.39%),P:1e-189				
Tpos:98.4,Tstd:34.7,Bpos:99.7,Bstd:58.5,StrandBias:0.2,Multiplicity:1.01						
0.778	0.001	0.001	0.220			
0.001	0.001	0.997	0.001			
0.997	0.001	0.001	0.001			
0.001	0.001	0.001	0.997			
0.001	0.347	0.419	0.233			
0.186	0.108	0.269	0.438			
0.240	0.305	0.245	0.210			
0.342	0.126	0.257	0.275			
0.336	0.203	0.270	0.191			
0.311	0.312	0.179	0.198			
0.312	0.125	0.287	0.276			
0.054	0.502	0.420	0.024			
0.683	0.001	0.001	0.315			
0.001	0.001	0.997	0.001			
0.997	0.001	0.001	0.001			
0.001	0.001	0.001	0.997			
0.689	0.001	0.077	0.233			
0.498	0.024	0.334	0.143			
0.174	0.299	0.341	0.186			
0.330	0.221	0.209	0.239			
>NNNNNBAGATAWYATCTVHN				GATA-IR3(Zf)/iTreg-Gata3-ChIP-Seq(GSE20898)/Homer	8.212313	-
725.789950	0	T:607.0(8.87%),B:480.8(1.18%),P:1e-315				
Tpos:101.3,Tstd:39.9,Bpos:100.4,Bstd:63.0,StrandBias:0.0,Multiplicity:1.03						
0.296	0.200	0.290	0.213			
0.290	0.191	0.277	0.242			
0.258	0.208	0.296	0.238			
0.305	0.221	0.255	0.219			
0.294	0.167	0.266	0.273			
0.167	0.337	0.307	0.189			

0.593	0.001	0.001	0.405
0.001	0.001	0.997	0.001
0.997	0.001	0.001	0.001
0.001	0.001	0.001	0.997
0.423	0.168	0.236	0.172
0.391	0.001	0.213	0.395
0.168	0.257	0.183	0.392
0.997	0.001	0.001	0.001
0.001	0.001	0.001	0.997
0.001	0.997	0.001	0.001
0.413	0.001	0.001	0.585
0.197	0.298	0.334	0.172
0.302	0.290	0.127	0.281
0.253	0.245	0.212	0.290
>NAGATWNB NATCTNN	GATA-IR4(Zf)/iTreg-Gata3-ChIP-Seq(GSE20898)/Homer	9.537620	-
709.341419	0	T:472.0(6.90%),B:263.6(0.64%),P:1e-308	
Tpos:101.8,Tstd:35.2,Bpos:95.4,Bstd:65.9,StrandBias:0.1,Multiplicity:1.02			
0.213	0.325	0.242	0.220
0.614	0.001	0.001	0.384
0.001	0.001	0.997	0.001
0.997	0.001	0.001	0.001
0.001	0.001	0.001	0.997
0.382	0.189	0.183	0.246
0.308	0.209	0.297	0.186
0.184	0.280	0.202	0.334
0.279	0.186	0.234	0.301
0.997	0.001	0.001	0.001
0.001	0.001	0.001	0.997
0.001	0.997	0.001	0.001
0.381	0.001	0.001	0.617
0.228	0.234	0.299	0.239
0.245	0.276	0.204	0.275
>AGATAASR	GATA3(Zf)/iTreg-Gata3-ChIP-Seq(GSE20898)/Homer	5.938282	-2390.596730 0
T:3368.0(49.21%),B:5753.5(14.09%),P:1e-1038 Tpos:100.1,Tstd:42.5,Bpos:100.6,Bstd:63.6,StrandBias:-0.0,Multiplicity:1.21			
0.662	0.066	0.006	0.266
0.001	0.007	0.991	0.001
0.989	0.004	0.001	0.006
0.002	0.023	0.001	0.974
0.825	0.061	0.011	0.103
0.778	0.048	0.129	0.045
0.184	0.401	0.348	0.067
0.433	0.167	0.359	0.041
>CAGATAAGGN	Gata1(Zf)/K562-GATA1-ChIP-Seq/Homer	6.402981	-6.924523e+03 0
50000.0,9843.0,7580.0,5717.0,0.00e+00			
0.109	0.469	0.365	0.057
0.700	0.024	0.001	0.275
0.001	0.001	0.997	0.001
0.997	0.001	0.001	0.001
0.001	0.001	0.001	0.997
0.997	0.001	0.001	0.001
0.997	0.001	0.001	0.001
0.023	0.145	0.830	0.001
0.311	0.237	0.434	0.018

<http://biowhat.ucsd.edu/homer/custom.motifs>[6/22/12 17:50:39]

0.659 0.158 0.113 0.070
 0.064 0.852 0.049 0.035
 0.778 0.110 0.074 0.038
 0.630 0.117 0.122 0.132
 0.075 0.176 0.121 0.628
 0.039 0.095 0.073 0.792
 0.041 0.844 0.043 0.073
 0.026 0.918 0.027 0.029
 0.052 0.875 0.026 0.048

>AACTACAATTCCCAGAATGC GFY-Staf/Promoters/Homer 4.609527 -7.226177e+02 0
 49998.0,6173.0,1028.1,575.0,1.48e-314

0.543 0.005 0.452 0.001
 0.997 0.001 0.001 0.001
 0.001 0.997 0.001 0.001
 0.001 0.007 0.001 0.991
 0.997 0.001 0.001 0.001
 0.001 0.997 0.001 0.001
 0.898 0.026 0.075 0.001
 0.518 0.092 0.080 0.311
 0.097 0.259 0.080 0.564
 0.035 0.001 0.001 0.963
 0.001 0.963 0.001 0.035
 0.001 0.997 0.001 0.001
 0.001 0.997 0.001 0.001
 0.942 0.001 0.056 0.001
 0.113 0.096 0.713 0.078
 0.527 0.266 0.181 0.026
 0.830 0.007 0.162 0.001
 0.071 0.016 0.426 0.487
 0.009 0.085 0.824 0.082
 0.005 0.958 0.007 0.031

>CGTGGGTGGTCC GLI3(Zf)/GLI3-ChIP-Chip/Homer 8.554905 -1.507418e+03 0
 50000.0,5214.0,1326.9,874.0,0.00e+00

0.088 0.564 0.128 0.219
 0.071 0.264 0.546 0.119
 0.028 0.024 0.015 0.933
 0.020 0.005 0.959 0.015
 0.001 0.001 0.902 0.096
 0.005 0.013 0.980 0.002
 0.033 0.006 0.011 0.951
 0.007 0.002 0.990 0.001
 0.009 0.002 0.972 0.016
 0.005 0.074 0.041 0.880
 0.014 0.958 0.015 0.013
 0.078 0.634 0.041 0.246

>NAGAACAGNCTGTTCT GRE(NR/IR3)/A549-GR-ChIP-Seq/Homer 7.914401 -5.701193e+03 0
 49999.0,2919.0,1870.5,1738.0,0.00e+00

0.251 0.287 0.290 0.171
 0.458 0.001 0.412 0.128
 0.030 0.001 0.968 0.001
 0.354 0.209 0.303 0.134
 0.997 0.001 0.001 0.001
 0.001 0.997 0.001 0.001
 0.789 0.001 0.134 0.076

0.102	0.273	0.361	0.264				
0.282	0.218	0.214	0.286				
0.251	0.374	0.269	0.106				
0.091	0.118	0.001	0.790				
0.001	0.001	0.997	0.001				
0.001	0.001	0.001	0.997				
0.138	0.304	0.199	0.359				
0.001	0.974	0.001	0.024				
0.129	0.429	0.001	0.441				
>VAGRACAKWCTGTYC				GRE/RAW264.7-GRE-ChIP-Seq/Homer	7.809859	-4041.196255	0
T:1437.0(71.35%),B:1197.7(2.63%),P:1e-1755				Tpos:101.8,Tstd:27.3,Bpos:98.0,Bstd:69.4,StrandBias:-0.0,Multiplicity:1.13			
0.232	0.373	0.231	0.163				
0.619	0.006	0.262	0.113				
0.058	0.005	0.924	0.013				
0.433	0.120	0.287	0.160				
0.957	0.008	0.015	0.020				
0.002	0.963	0.008	0.027				
0.767	0.019	0.135	0.079				
0.123	0.231	0.249	0.397				
0.259	0.179	0.165	0.397				
0.210	0.403	0.228	0.160				
0.101	0.044	0.009	0.846				
0.006	0.015	0.972	0.007				
0.021	0.026	0.018	0.935				
0.179	0.288	0.131	0.401				
0.041	0.865	0.014	0.080				
>CACAGCAGGGGG				HEB [?] /mES-Nanog-ChIP-Seq/Homer	7.697673	-1.565462e+03	0
53740.0,26870.0,5118.7,3845.0,0.00e+00							
0.120	0.837	0.034	0.009				
0.481	0.232	0.014	0.273				
0.014	0.982	0.002	0.001				
0.920	0.003	0.011	0.066				
0.020	0.018	0.953	0.009				
0.008	0.942	0.005	0.044				
0.875	0.001	0.042	0.082				
0.028	0.001	0.920	0.050				
0.002	0.001	0.993	0.004				
0.302	0.062	0.504	0.132				
0.282	0.036	0.519	0.163				
0.233	0.129	0.501	0.137				
>TACGTGCV				HIF-1a(HLH)/MCF7-HIF1a-ChIP-Seq/Homer	7.055265	-637.276768	0
T:341.0(31.40%),B:42.8(2.30%),P:1e-276				Tpos:102.1,Tstd:40.3,Bpos:89.3,Bstd:78.5,StrandBias:0.0,Multiplicity:1.08			
0.028	0.149	0.273	0.550				
0.868	0.001	0.130	0.001				
0.005	0.959	0.001	0.035				
0.004	0.015	0.970	0.011				
0.002	0.053	0.001	0.944				
0.131	0.056	0.789	0.024				
0.328	0.650	0.002	0.020				
0.223	0.345	0.269	0.163				
>GGTTAAACATTAAC				Hnf1(Homeobox)/Liver-Foxa2-Chip-Seq/Homer	8.866137	-3.759539e+02	0
50000.0,10171.0,989.7,528.0,5.31e-164							
0.262	0.035	0.478	0.224				

0.142	0.061	0.681	0.117				
0.050	0.043	0.033	0.874				
0.072	0.054	0.071	0.802				
0.899	0.029	0.037	0.035				
0.860	0.083	0.011	0.045				
0.540	0.053	0.007	0.400				
0.150	0.503	0.184	0.163				
0.766	0.004	0.014	0.215				
0.030	0.002	0.065	0.903				
0.028	0.051	0.026	0.895				
0.828	0.040	0.045	0.088				
0.889	0.053	0.043	0.015				
>CAGAGGNCAAAGTCCA				HNF4a(NR/DR1)/HepG2-HNF4a-ChIP-Seq/Homer	7.857574	-	
1.347630e+04 0 52990.0,26495.0,11708.0,10770.0,0.00e+00							
0.241	0.406	0.165	0.188				
0.471	0.170	0.148	0.211				
0.372	0.139	0.393	0.097				
0.518	0.009	0.376	0.097				
0.113	0.016	0.800	0.071				
0.193	0.074	0.375	0.358				
0.138	0.318	0.212	0.332				
0.018	0.937	0.003	0.042				
0.982	0.012	0.004	0.002				
0.788	0.019	0.187	0.007				
0.875	0.001	0.118	0.006				
0.009	0.009	0.963	0.020				
0.061	0.021	0.273	0.645				
0.057	0.466	0.123	0.354				
0.032	0.917	0.002	0.050				
0.806	0.066	0.018	0.111				
>NTATYGATCH				HNF6(Homeobox)/Liver-Hnf6-ChIP-Seq(ERP000394)	4.573747	-51168.547695	0
T:33356.0(56.70%),B:3854.7(6.81%),P:1e-2222							
0.249	0.192	0.318	0.241				
0.130	0.105	0.159	0.606				
0.997	0.001	0.001	0.001				
0.001	0.001	0.001	0.997				
0.001	0.510	0.001	0.488				
0.001	0.001	0.997	0.001				
0.997	0.001	0.001	0.001				
0.001	0.029	0.001	0.969				
0.001	0.674	0.001	0.324				
0.237	0.380	0.142	0.242				
>GYCATCMATCAT				HOXA2(Homeobox)/mES-Hoxa2-ChIP-Seq/Homer	9.388852	-2166.132663	0
T:1175.0(15.51%),B:430.2(1.04%),P:1e-940							
0.175	0.250	0.455	0.120				
0.168	0.316	0.077	0.439				
0.329	0.600	0.064	0.007				
0.977	0.001	0.001	0.021				
0.001	0.001	0.001	0.997				
0.057	0.761	0.001	0.181				
0.485	0.460	0.001	0.054				
0.989	0.001	0.001	0.009				
0.001	0.001	0.001	0.997				
0.001	0.996	0.002	0.001				

<http://biowhat.ucsd.edu/homer/custom.motifs>[6/22/12 17:50:39]

0.004995004995005	0.013986013986014	0.98001998001998	0.000999000999000999
0.88	0.085	0.013	0.022
0.891	0.012	0.081	0.016
0.138861138861139	0.283716283716284	0.341658341658342	0.235764235764236
0.212787212787213	0.345654345654346	0.304695304695305	0.136863136863137
0.018	0.11	0.005	0.867
0.019019019019019	0.014014014014014	0.0670670670670671	0.8998998998999
0.001	0.993	0.001	0.005
0.005	0.486	0.02	0.489
0.513486513486513	0.107892107892108	0.282717282717283	0.0959040959040959
0.042	0.136	0.69	0.132
0.55	0.175	0.17	0.105
0.416	0.243	0.24	0.101
>ACTGAAACCA IRF4(IRF)/GM12878-IRF4-ChIP-Seq/Homer 7.369037 -469.191332 0			
T:492.0(12.60%),B:969.2(2.25%),P:1e-203 Tpos:48.7,Tstd:23.9,Bpos:50.0,Bstd:34.2,StrandBias:-0.2,Multiplicity:1.02			
0.438	0.218	0.120	0.224
0.205	0.403	0.234	0.157
0.031	0.291	0.083	0.595
0.006	0.040	0.922	0.032
0.964	0.034	0.001	0.001
0.815	0.001	0.042	0.142
0.953	0.006	0.001	0.040
0.020	0.680	0.248	0.052
0.104	0.611	0.012	0.273
0.432	0.086	0.258	0.223
>AGTTTCAGTTTC ISRE(IRF)/ThioMac-LPS-exp/HOMER 10.512507 -1.194927e+02 0			
87179.0,1722.0,789.4,104.0,0.00e+00			
0.653	0.001	0.229	0.116
0.008	0.288	0.676	0.028
0.001	0.003	0.001	0.995
0.001	0.027	0.001	0.971
0.001	0.001	0.025	0.973
0.003	0.975	0.019	0.002
0.583	0.007	0.240	0.170
0.079	0.393	0.471	0.056
0.016	0.001	0.001	0.982
0.001	0.001	0.004	0.994
0.009	0.045	0.004	0.943
0.001	0.881	0.010	0.108
>GATGACTCATCN Jun-API(bZIP)/K562-cJun-ChIP-Seq/Homer 8.486263 -3.459607e+04 0			
74838.0,37419.0,20963.2,20135.0,0.00e+00			
0.244	0.149	0.396	0.211
0.473	0.218	0.268	0.041
0.001	0.001	0.001	0.997
0.001	0.001	0.991	0.007
0.997	0.001	0.001	0.001
0.051	0.462	0.440	0.047
0.021	0.001	0.001	0.977
0.024	0.974	0.001	0.001
0.997	0.001	0.001	0.001
0.050	0.258	0.227	0.465
0.210	0.394	0.153	0.243
0.241	0.319	0.211	0.229
>ATGACGTCATCC c-Jun-CRE(bZIP)/K562-cJun-ChIP-Seq/Homer 7.337919 -6.299471e+03 0			

74838.0,37419.0,8277.4,7112.0,0.00e+00

0.598 0.118 0.273 0.010

0.003 0.002 0.001 0.994

0.003 0.004 0.955 0.038

0.978 0.002 0.011 0.008

0.020 0.643 0.127 0.210

0.219 0.139 0.609 0.032

0.005 0.011 0.001 0.983

0.028 0.965 0.003 0.004

0.987 0.002 0.002 0.008

0.010 0.272 0.112 0.606

0.063 0.652 0.111 0.174

0.217 0.379 0.174 0.230

>ATGACGTCATCN JunD(bZIP)/K562-JunD-ChIP-Seq/Homer 8.840059 -6.690263e+03 0

49999.0,4821.0,3322.5,2871.0,0.00e+00

0.636 0.078 0.276 0.010

0.001 0.001 0.001 0.997

0.001 0.001 0.981 0.017

0.994 0.001 0.004 0.001

0.001 0.785 0.041 0.173

0.165 0.034 0.800 0.001

0.001 0.003 0.001 0.995

0.016 0.982 0.001 0.001

0.997 0.001 0.001 0.001

0.009 0.282 0.080 0.630

0.134 0.529 0.131 0.206

0.281 0.254 0.256 0.209

>GCCACACCCA Klf4(Zf)/mES-Klf4-ChIP-Seq/Homer 7.333962 -5.449559e+03 0

25000.0,10220.0,5867.0,5098.0,0.00e+00

0.163 0.004 0.827 0.006

0.061 0.907 0.014 0.017

0.001 0.994 0.001 0.004

0.859 0.132 0.008 0.001

0.001 0.997 0.001 0.001

0.855 0.001 0.143 0.001

0.001 0.997 0.001 0.001

0.001 0.997 0.001 0.001

0.001 0.997 0.001 0.001

0.764 0.083 0.006 0.147

>CTAATTAGCN Lhx3(Homeobox)/Forebrain-p300-ChIP-Seq/Homer 5.544268 -9.444635e+02 0

50001.4,2417.0,11333.0,2046.0,0.00e+00

0.061 0.539 0.113 0.286

0.006 0.161 0.001 0.832

0.924 0.068 0.005 0.003

0.995 0.001 0.001 0.003

0.001 0.001 0.001 0.997

0.001 0.006 0.095 0.898

0.827 0.001 0.165 0.007

0.223 0.116 0.623 0.039

0.195 0.472 0.197 0.136

0.299 0.306 0.138 0.257

>GGCAATTAAA Unknown/Homeobox/Limb-p300-ChIP-Seq/Homer 6.765817 -3.134587e+02 0

49999.0,2102.0,16051.2,1529.0,7.36e-137

0.189 0.302 0.378 0.131

0.066	0.372	0.444	0.118				
0.086	0.629	0.013	0.273				
0.421	0.346	0.019	0.213				
0.632	0.030	0.190	0.149				
0.104	0.014	0.013	0.868				
0.463	0.005	0.023	0.508				
0.896	0.020	0.014	0.071				
0.894	0.044	0.050	0.013				
0.616	0.113	0.087	0.184				
>BTCAAGGTCA				Nr5a2(NR)/Pancreas-LRH1-ChIP-Seq(GSE34295)/Homer	7.021126	-8380.456657	0
T:9098.0(36.25%),B:1828.3(7.50%),P:1e-3639							
0.100	0.266	0.313	0.321				
0.051	0.344	0.038	0.567				
0.012	0.917	0.066	0.005				
0.974	0.004	0.011	0.011				
0.961	0.003	0.024	0.012				
0.007	0.003	0.982	0.008				
0.015	0.003	0.973	0.009				
0.112	0.224	0.051	0.613				
0.020	0.871	0.019	0.090				
0.832	0.037	0.062	0.069				
>GGGTACTANAGGTCA				LXRE(NR/DR4)/BLRP(RAW)-LXRb-ChIP-Seq/Homer	9.099916	-	
8.147036e+02 0 20000.0,958.0,332.4,269.0,0.00e+00							
0.423	0.001	0.568	0.008				
0.020	0.001	0.931	0.048				
0.008	0.001	0.918	0.073				
0.001	0.020	0.001	0.978				
0.084	0.322	0.028	0.566				
0.959	0.001	0.016	0.024				
0.044	0.752	0.024	0.181				
0.044	0.191	0.044	0.722				
0.406	0.219	0.159	0.216				
0.215	0.274	0.272	0.239				
0.706	0.127	0.147	0.020				
0.052	0.001	0.927	0.020				
0.016	0.008	0.825	0.151				
0.044	0.072	0.155	0.730				
0.001	0.938	0.001	0.060				
0.932	0.008	0.048	0.012				
>TGCTGACTCA				MafA(bZIP)/Islet-MafA-ChIP-Seq(GSE30298)/Homer	6.979327	-833.929476	0
T:901.0(37.28%),B:3455.7(7.76%),P:1e-362							
0.061	0.144	0.032	0.763				
0.009	0.001	0.989	0.001				
0.004	0.988	0.001	0.007				
0.017	0.005	0.017	0.961				
0.001	0.001	0.975	0.023				
0.948	0.011	0.024	0.017				
0.010	0.731	0.195	0.064				
0.179	0.123	0.114	0.584				
0.125	0.549	0.192	0.134				
0.620	0.100	0.159	0.121				
>ACCACGTGGTCN				Max(HLH)/K562-Max-ChIP-Seq/Homer	6.608033	-1.622921e+04	0
50000.0,20359.0,23098.8,19884.0,0.00e+00							
0.410	0.165	0.297	0.127				

0.256	0.414	0.224	0.106			
0.001	0.997	0.001	0.001			
0.997	0.001	0.001	0.001			
0.001	0.869	0.001	0.129			
0.109	0.001	0.889	0.001			
0.001	0.001	0.001	0.997			
0.001	0.001	0.997	0.001			
0.104	0.241	0.409	0.246			
0.124	0.291	0.168	0.417			
0.164	0.355	0.167	0.314			
0.264	0.261	0.210	0.264			
>CCAAAAATAG Mef2a(MADS)/HL1-Mef2a.biotin-ChIP-Seq/Homer/				7.668953	-289.885563	0
T:248.0(14.49%),B:915.5(2.05%),P:1e-125						
0.120	0.781	0.040	0.060			
0.025	0.465	0.002	0.509			
0.409	0.171	0.172	0.250			
0.561	0.116	0.165	0.159			
0.746	0.049	0.180	0.026			
0.888	0.002	0.002	0.109			
0.926	0.003	0.002	0.070			
0.006	0.002	0.007	0.986			
0.924	0.002	0.073	0.002			
0.047	0.084	0.856	0.014			
>GGCVGTTR MYB(HTH)/ERMYB-Myb-ChIPSeq(GSE22095)/Homer				2.200585	-4460.104388	0
T:7048.0(56.60%),B:6592.8(18.57%),P:1e-1936						
Tpos:100.4,Tstd:37.0,Bpos:100.3,Bstd:69.7,StrandBias:0.0,Multiplicity:1.34						
0.105	0.001	0.843	0.051			
0.170	0.001	0.663	0.166			
0.001	0.997	0.001	0.001			
0.323	0.320	0.355	0.001			
0.001	0.001	0.997	0.001			
0.001	0.001	0.001	0.997			
0.001	0.001	0.001	0.997			
0.532	0.001	0.466	0.001			
>AGCAGCTGCTGC MyoD(HLH)/Myotube-MyoD-ChIP-Seq/Homer				7.249827	-2.493974e+04	0
49999.2,13125.0,15210.6,14377.0,0.00e+00						
0.432	0.137	0.291	0.140			
0.400	0.102	0.455	0.042			
0.001	0.997	0.001	0.001			
0.966	0.001	0.001	0.032			
0.001	0.014	0.984	0.001			
0.001	0.996	0.002	0.001			
0.093	0.001	0.001	0.905			
0.001	0.001	0.997	0.001			
0.003	0.518	0.079	0.400			
0.123	0.268	0.093	0.516			
0.187	0.234	0.377	0.202			
0.165	0.373	0.197	0.265			
>GGCCATTAAC Nanog(Homeobox)/mES-Nanog-ChIP-Seq/Homer				4.457057	-2.020737e+03	0
51498.0,25749.0,9393.2,6781.0,0.00e+00						
0.401	0.024	0.489	0.086			
0.085	0.283	0.572	0.060			
0.026	0.762	0.016	0.196			
0.302	0.619	0.035	0.045			

0.919	0.019	0.011	0.052				
0.001	0.003	0.001	0.995				
0.001	0.364	0.003	0.632				
0.736	0.116	0.036	0.112				
0.580	0.115	0.240	0.065				
0.182	0.477	0.227	0.114				
>GCCATCTGTT				NeuroD1(bHLH)/Islet-NeuroD1-ChIP-Seq(GSE30298)/Homer	7.081424	-5655.918830	0
T:4616.0(54.33%),B:3592.2(8.94%),P:1e-2456							
0.321	0.100	0.511	0.068				
0.128	0.821	0.034	0.017				
0.015	0.977	0.004	0.004				
0.977	0.004	0.009	0.010				
0.010	0.042	0.221	0.727				
0.035	0.952	0.008	0.005				
0.003	0.003	0.004	0.990				
0.007	0.002	0.990	0.001				
0.005	0.265	0.222	0.508				
0.050	0.356	0.046	0.548				
>NNTGTTTATTTTGGCA				NF1:FOXA1/LNCAP-FOXA1-ChIP-Seq/Homer	10.273683	-1.367298e+03	0
50000.0,14301.0,949.6,809.0,0.00e+00							
0.273	0.184	0.193	0.350				
0.248	0.286	0.166	0.299				
0.075	0.001	0.001	0.924				
0.256	0.008	0.732	0.004				
0.024	0.031	0.001	0.945				
0.001	0.001	0.121	0.877				
0.035	0.001	0.218	0.745				
0.490	0.017	0.357	0.137				
0.125	0.354	0.063	0.458				
0.107	0.260	0.047	0.587				
0.001	0.093	0.001	0.905				
0.021	0.001	0.001	0.977				
0.001	0.001	0.990	0.008				
0.001	0.009	0.984	0.006				
0.126	0.861	0.007	0.007				
0.616	0.112	0.002	0.270				
>TTGCCAAG				NF1-halfsite(CTF)/LNCaP-NF1-ChIP-Seq/Homer	6.734334	-9.654380e+03	0
50000.0,22740.0,32422.7,25145.0,0.00e+00							
0.029	0.338	0.192	0.441				
0.001	0.001	0.001	0.997				
0.001	0.001	0.997	0.001				
0.001	0.635	0.363	0.001				
0.001	0.997	0.001	0.001				
0.997	0.001	0.001	0.001				
0.621	0.001	0.377	0.001				
0.215	0.298	0.472	0.016				
>CTTGGCACNGTGCCAA				NF1(CTF)/LNCAP-NF1-ChIP-Seq/Homer	7.598341	-1.324803e+04	0
50000.0,14301.0,12622.8,10808.0,0.00e+00							
0.113	0.473	0.201	0.213				
0.067	0.426	0.038	0.470				
0.056	0.003	0.158	0.783				
0.001	0.001	0.995	0.003				
0.019	0.004	0.964	0.014				
0.164	0.782	0.043	0.011				

0.506	0.186	0.033	0.275			
0.179	0.356	0.257	0.208			
0.262	0.222	0.211	0.304			
0.180	0.271	0.352	0.197			
0.264	0.029	0.194	0.513			
0.008	0.053	0.784	0.155			
0.016	0.960	0.005	0.019			
0.005	0.993	0.001	0.001			
0.784	0.155	0.003	0.058			
0.442	0.064	0.421	0.073			
>GAATGGAAAAAATGAGTCAT NFAT:AP1/Jurkat-NFATC1-ChIP-Seq/Homer 8.950862 -2.450578e+03 0						
50000.6,4315.0,952.1,826.0,0.00e+00						
0.198	0.289	0.369	0.145			
0.455	0.147	0.247	0.152			
0.389	0.159	0.314	0.139			
0.194	0.037	0.094	0.675			
0.011	0.001	0.987	0.001			
0.004	0.001	0.994	0.001			
0.924	0.024	0.043	0.009			
0.940	0.001	0.052	0.007			
0.801	0.054	0.096	0.049			
0.522	0.165	0.202	0.111			
0.456	0.052	0.089	0.404			
0.375	0.205	0.292	0.129			
0.061	0.036	0.024	0.879			
0.047	0.043	0.788	0.122			
0.817	0.022	0.050	0.111			
0.105	0.324	0.503	0.068			
0.117	0.069	0.067	0.747			
0.162	0.630	0.123	0.085			
0.721	0.056	0.094	0.129			
0.136	0.265	0.234	0.366			
>ATTTTCCATT NFAT(RHD)/Jurkat-NFATC1-ChIP-Seq/Homer 6.822934 -2.986043e+03 0						
50000.6,4315.0,2875.8,1844.0,0.00e+00						
0.539	0.029	0.061	0.372			
0.058	0.165	0.237	0.539			
0.001	0.001	0.030	0.968			
0.001	0.001	0.001	0.997			
0.001	0.001	0.001	0.997			
0.001	0.997	0.001	0.001			
0.001	0.997	0.001	0.001			
0.925	0.001	0.073	0.001			
0.031	0.358	0.059	0.552			
0.081	0.254	0.139	0.526			
>GATGACTCAGCA NF-E2(bZIP)/K562-NFE2-ChIP-Seq/Homer 9.748382 -1.054915e+04 0						
49999.0,9248.0,3227.2,3131.0,0.00e+00						
0.212	0.221	0.461	0.106			
0.714	0.074	0.202	0.010			
0.002	0.001	0.002	0.995			
0.001	0.001	0.987	0.011			
0.994	0.002	0.001	0.003			
0.027	0.796	0.137	0.040			
0.061	0.006	0.012	0.921			
0.058	0.898	0.016	0.028			

0.939	0.003	0.023	0.035				
0.011	0.011	0.899	0.079				
0.007	0.964	0.023	0.006				
0.846	0.032	0.040	0.082				
>GGAAATTCCC NFkB-p65-Rel(RHD)/LPS-exp/Homer 9.229901 -9.602478e+01 0							
87190.0,1722.0,597.7,81.0,1.98e-42							
0.001	0.001	0.993	0.005				
0.017	0.003	0.932	0.048				
0.972	0.002	0.001	0.025				
0.983	0.002	0.013	0.001				
0.866	0.006	0.001	0.128				
0.020	0.101	0.042	0.838				
0.001	0.163	0.001	0.835				
0.091	0.907	0.001	0.001				
0.001	0.980	0.001	0.018				
0.138	0.849	0.012	0.001				
>AGCCAATCGG NFY(CCAAT)/Promoter/Homer 4.705513 -1.941776e+03 0							
149303.9,11385.0,11230.3,3044.0,0.00e+00							
0.474	0.049	0.467	0.010				
0.228	0.049	0.669	0.054				
0.001	0.997	0.001	0.001				
0.001	0.997	0.001	0.001				
0.997	0.001	0.001	0.001				
0.997	0.001	0.001	0.001				
0.001	0.001	0.001	0.997				
0.081	0.493	0.399	0.027				
0.460	0.041	0.498	0.001				
0.107	0.179	0.696	0.019				
>AASCACTCAA Nkx2.5(Homeobox)/HL1-Nkx2.5.biotin-ChIP-Seq/Homer 5.898166 -4609.073239 0							
T:11212.0(56.23%),B:7132.4(24.37%),P:1e-1998							
0.358	0.149	0.331	0.163				
0.353	0.153	0.327	0.169				
0.193	0.413	0.385	0.011				
0.003	0.901	0.002	0.095				
0.650	0.086	0.002	0.263				
0.015	0.971	0.002	0.013				
0.002	0.002	0.002	0.995				
0.023	0.561	0.006	0.411				
0.492	0.200	0.197	0.113				
0.598	0.040	0.158	0.205				
>GNCCACGTGG n-Myc(HLH)/mES-nMyc-ChIP-Seq/Homer 5.719269 -3.700101e+03 0							
50003.0,10078.0,13222.0,7084.0,0.00e+00							
0.256	0.257	0.359	0.128				
0.348	0.177	0.340	0.135				
0.086	0.584	0.261	0.069				
0.001	0.997	0.001	0.001				
0.974	0.001	0.010	0.016				
0.001	0.962	0.001	0.036				
0.057	0.001	0.941	0.001				
0.014	0.011	0.001	0.974				
0.001	0.001	0.997	0.001				
0.069	0.233	0.613	0.085				
>TTCAAGGTCA Nr5a2(NR)/mES-Nr5a2-ChIP-Seq/Homer 7.613762 -4.374139e+02 0							
50000.0,599.0,647.1,195.0,1.08e-190							

0.052	0.279	0.285	0.383				
0.009	0.381	0.006	0.604				
0.003	0.956	0.039	0.003				
0.991	0.004	0.001	0.004				
0.993	0.001	0.005	0.001				
0.004	0.001	0.992	0.003				
0.002	0.005	0.987	0.006				
0.130	0.326	0.059	0.485				
0.003	0.931	0.003	0.063				
0.704	0.040	0.109	0.147				
>CTGCGCATGCGC NRF1(NRF)/MCF7-NRF1-ChIP-Seq/Homer				8.595777	-4.549400e+03	0	
128398.8,6696.0,14017.4,4778.0,0.00e+00							
0.078	0.628	0.197	0.098				
0.052	0.170	0.106	0.673				
0.036	0.006	0.955	0.003				
0.040	0.936	0.018	0.006				
0.017	0.004	0.976	0.003				
0.005	0.983	0.007	0.006				
0.734	0.168	0.068	0.031				
0.041	0.077	0.167	0.715				
0.005	0.013	0.975	0.007				
0.011	0.959	0.004	0.026				
0.008	0.020	0.922	0.050				
0.006	0.928	0.009	0.057				
>GTGCGCATGCGC NRF1/Promoter/Homer				1.457174	-1.519451e+03	0	
149303.9,11385.0,12173.3,2895.0,0.00e+00							
0.001	0.415	0.432	0.152				
0.001	0.391	0.001	0.607				
0.001	0.001	0.997	0.001				
0.001	0.997	0.001	0.001				
0.001	0.001	0.997	0.001				
0.001	0.997	0.001	0.001				
0.997	0.001	0.001	0.001				
0.001	0.001	0.001	0.997				
0.001	0.001	0.997	0.001				
0.001	0.997	0.001	0.001				
0.001	0.001	0.997	0.001				
0.001	0.997	0.001	0.001				
>TGACCTTTNCNT Nur77(NR)/K562-NR4A1-ChIP-Seq(GSE31363)/Homer				8.736456	-127.180670	0	
T:63.0(21.07%),B:617.0(1.27%),P:1e-55 Tpos:103.6,Tstd:39.0,Bpos:98.4,Bstd:58.0,StrandBias:0.1,Multiplicity:1.05							
0.001	0.001	0.001	0.997				
0.177	0.001	0.821	0.001				
0.997	0.001	0.001	0.001				
0.001	0.997	0.001	0.001				
0.088	0.910	0.001	0.001				
0.001	0.001	0.001	0.997				
0.001	0.001	0.001	0.997				
0.001	0.001	0.001	0.997				
0.308	0.300	0.167	0.224				
0.209	0.535	0.122	0.134				
0.266	0.295	0.261	0.178				
0.181	0.255	0.136	0.428				
>ATATGCAAAT Oct2(POU/Homeobox)/Bcell-Oct2-ChIP-Seq/Homer				7.581119	-3.525761e+02	0	
9675.5,3971.0,381.2,319.0,0.00e+00							

0.696	0.132	0.077	0.095				
0.110	0.103	0.028	0.759				
0.974	0.006	0.001	0.019				
0.025	0.013	0.032	0.930				
0.020	0.001	0.972	0.007				
0.020	0.952	0.001	0.027				
0.997	0.001	0.001	0.001				
0.966	0.001	0.020	0.013				
0.992	0.001	0.001	0.006				
0.064	0.019	0.026	0.891				
>ATTTGCATAA	Oct4(POU/Homeobox)/mES-Oct4-ChIP-Seq/Homer			7.325577	-5.831779e+03	0	
55120.0,27560.0,7917.0,6798.0,0.00e+00							
0.852	0.048	0.048	0.052				
0.001	0.001	0.001	0.997				
0.001	0.043	0.001	0.955				
0.155	0.001	0.001	0.843				
0.107	0.001	0.891	0.001				
0.008	0.990	0.001	0.001				
0.997	0.001	0.001	0.001				
0.001	0.004	0.001	0.994				
0.676	0.047	0.125	0.152				
0.434	0.107	0.063	0.396				
>ATTTGCATAACAATG	OCT4-SOX2-TCF-NANOG((POU/Homeobox/HMG)/mES-ChIP-Seq/Homer						
8.751420	-7.183775e+03	0	30001.0,11899.0,4508.4,4187.0,0.00e+00				
0.768	0.081	0.045	0.106				
0.023	0.012	0.003	0.962				
0.029	0.102	0.014	0.856				
0.190	0.008	0.007	0.795				
0.162	0.089	0.728	0.022				
0.085	0.908	0.001	0.005				
0.974	0.010	0.001	0.015				
0.081	0.020	0.004	0.894				
0.537	0.103	0.186	0.174				
0.624	0.056	0.012	0.308				
0.127	0.595	0.177	0.101				
0.935	0.018	0.010	0.038				
0.925	0.015	0.031	0.030				
0.398	0.012	0.038	0.551				
0.222	0.053	0.644	0.081				
>RCCATMTGTT	Olig2(bHLH)/Neuron-Olig2-ChIP-Seq(GSE30882)/Homer			6.053393	-12730.683530	0	
T:21120.0(60.40%),B:7084.4(21.03%),P:1e-5528							
0.371	0.097	0.325	0.206				
0.179	0.644	0.159	0.018				
0.215	0.782	0.002	0.001				
0.754	0.001	0.094	0.151				
0.001	0.007	0.362	0.630				
0.389	0.513	0.050	0.048				
0.009	0.018	0.014	0.959				
0.023	0.001	0.975	0.001				
0.022	0.185	0.283	0.510				
0.132	0.291	0.086	0.490				
>GGGGGAATCCCC	NFkB-p50,p52(RHD)/p50-ChIP-Chip/Homer			8.671915	-1.074349e+02	0	
7947.8,301.0,472.7,108.0,2.20e-47							
0.138	0.091	0.682	0.089				

0.073	0.035	0.840	0.053
0.058	0.001	0.918	0.023
0.042	0.001	0.956	0.001
0.099	0.044	0.856	0.001
0.818	0.030	0.151	0.001
0.683	0.055	0.160	0.102
0.155	0.034	0.090	0.721
0.001	0.803	0.093	0.103
0.021	0.956	0.001	0.021
0.001	0.962	0.001	0.036
0.054	0.902	0.033	0.011
>NAGACATGTCCGGACATGTC	p53(p53)/p53-ChIP-Chip/Homer	11.188694	-4.476468e+03 0
15000.0,1480.0,1123.3,1111.0,0.00e+00			
0.265	0.244	0.220	0.271
0.392	0.061	0.377	0.170
0.296	0.035	0.539	0.130
0.623	0.006	0.307	0.064
0.001	0.967	0.011	0.020
0.863	0.020	0.023	0.094
0.212	0.017	0.037	0.734
0.001	0.001	0.997	0.001
0.029	0.378	0.001	0.592
0.096	0.695	0.020	0.189
0.149	0.413	0.057	0.382
0.391	0.049	0.411	0.149
0.183	0.024	0.699	0.094
0.593	0.001	0.378	0.029
0.001	0.997	0.001	0.001
0.727	0.041	0.012	0.220
0.104	0.018	0.020	0.857
0.020	0.009	0.970	0.001
0.072	0.333	0.008	0.587
0.161	0.527	0.037	0.275
>ACATGCCCGGGCAT	p53(p53)/mES-cMyc-ChIP-Seq/Homer	11.048114	-2.025438e+02 0
50000.9,5819.0,201.5,125.0,1.09e-88			
0.592	0.006	0.328	0.074
0.006	0.958	0.035	0.001
0.741	0.066	0.063	0.130
0.214	0.031	0.092	0.664
0.001	0.004	0.994	0.001
0.001	0.854	0.022	0.122
0.015	0.962	0.001	0.022
0.024	0.724	0.016	0.236
0.246	0.022	0.686	0.046
0.012	0.006	0.978	0.004
0.117	0.018	0.864	0.001
0.006	0.992	0.001	0.001
0.724	0.088	0.058	0.129
0.166	0.086	0.186	0.562
>AACATGCCCGAGACATGCCCN	p53(p53)/Saos-p53-ChIP-Seq/Homer	8.592185	-9967.933520 0
T:4803.0(12.78%),B:238.8(0.66%),P:1e-4322			
Tpos:100.2,Tstd:33.3,Bpos:106.1,Bstd:69.8,StrandBias:0.0,Multiplicity:1.05			
0.275	0.108	0.421	0.198
0.481	0.052	0.334	0.135

0.019	0.826	0.070	0.086			
0.657	0.118	0.056	0.170			
0.221	0.028	0.099	0.653			
0.002	0.003	0.990	0.006			
0.077	0.406	0.021	0.498			
0.150	0.535	0.078	0.238			
0.168	0.360	0.132	0.341			
0.348	0.137	0.341	0.175			
0.241	0.078	0.526	0.156			
0.499	0.022	0.405	0.076			
0.007	0.987	0.004	0.003			
0.652	0.100	0.029	0.220			
0.165	0.054	0.108	0.674			
0.083	0.046	0.854	0.018			
0.126	0.343	0.057	0.476			
0.211	0.416	0.109	0.266			
0.190	0.349	0.138	0.325			
0.254	0.269	0.241	0.239			
>NNNGCATGTCCNGACATGCC p63(p53)/Keratinocyte-p63-ChIP-Seq/Homer				6.985591	-4.946037e+04	
0	65270.0,32635.0,20429.1,20153.0,0.00e+00					
0.264	0.260	0.222	0.254			
0.312	0.205	0.298	0.185			
0.293	0.105	0.339	0.262			
0.415	0.032	0.440	0.113			
0.002	0.992	0.003	0.004			
0.747	0.126	0.037	0.089			
0.325	0.013	0.174	0.487			
0.004	0.002	0.992	0.002			
0.092	0.378	0.009	0.520			
0.156	0.485	0.090	0.269			
0.168	0.372	0.152	0.308			
0.308	0.216	0.311	0.165			
0.310	0.077	0.434	0.178			
0.497	0.008	0.410	0.085			
0.001	0.997	0.001	0.001			
0.526	0.178	0.012	0.283			
0.108	0.039	0.126	0.728			
0.006	0.004	0.988	0.002			
0.107	0.465	0.029	0.400			
0.253	0.380	0.083	0.283			
>AGGGGATTTCCC NFkB-p65(RHD)/GM12787-p65-ChIP-Seq/Homer				7.188169	-1.019078e+04	0
50003.0,10349.0,10830.4,8612.0,0.00e+00						
0.386	0.093	0.197	0.324			
0.098	0.005	0.728	0.170			
0.002	0.001	0.992	0.004			
0.001	0.001	0.965	0.033			
0.354	0.001	0.622	0.023			
0.602	0.138	0.172	0.089			
0.355	0.072	0.016	0.557			
0.031	0.014	0.017	0.937			
0.001	0.193	0.001	0.805			
0.001	0.816	0.001	0.182			
0.002	0.973	0.001	0.023			
0.261	0.609	0.019	0.112			

```

>ACCGTGACTAATTNN    PAX3:FKHR-fusion(Paired/Homeobox)/Rh4-PAX3:FKHR-ChIP-Seq/Homer
7.413611    -2.455901e+03  0    49998.3,7065.0,1176.9,1007.0,0.00e+00
0.589  0.052  0.146  0.214
0.003  0.595  0.401  0.001
0.001  0.746  0.001  0.252
0.472  0.006  0.505  0.017
0.018  0.001  0.075  0.906
0.194  0.003  0.787  0.015
0.782  0.015  0.102  0.100
0.087  0.911  0.001  0.001
0.222  0.101  0.025  0.652
0.958  0.002  0.039  0.001
0.997  0.001  0.001  0.001
0.091  0.022  0.018  0.868
0.060  0.213  0.195  0.532
0.273  0.199  0.290  0.237
0.322  0.216  0.270  0.192
>GCAGCCAAGCGTGACC    PAX5(Paired/Homeobox)/GM12878-PAX5-ChIP-Seq/Homer    7.209436    -
2.512931e+03  0    49999.7,3602.0,1236.0,981.0,0.00e+00
0.002  0.234  0.462  0.302
0.037  0.890  0.001  0.073
0.856  0.025  0.073  0.046
0.089  0.202  0.605  0.104
0.002  0.569  0.127  0.302
0.116  0.598  0.262  0.024
0.710  0.012  0.264  0.013
0.509  0.033  0.194  0.264
0.002  0.271  0.727  0.001
0.001  0.883  0.015  0.101
0.453  0.033  0.466  0.048
0.030  0.022  0.114  0.834
0.159  0.011  0.827  0.003
0.664  0.081  0.188  0.067
0.244  0.700  0.038  0.017
0.227  0.364  0.170  0.239
>GTCACGCTCNCCTGA    PAX5-shortForm(Paired/Homeobox)/GM12878-PAX5-ChIP-Seq/Homer    8.480103    -
5.219521e+02  0    49999.7,3602.0,266.3,199.0,2.08e-227
0.001  0.009  0.971  0.019
0.023976023976024    0.0549450549450549    0.000999000999000999    0.92007992007992
0.001  0.979  0.001  0.019
0.978  0.001  0.014  0.007
0.014  0.803  0.01  0.173
0.194  0.05  0.755  0.001
0.001  0.59  0.408  0.001
0.149  0.171  0.024  0.656
0.062  0.534  0.178  0.226
0.128  0.368  0.345  0.159
0.157157157157157    0.477477477477477    0.256256256256256    0.109109109109109
0.005  0.033  0.031  0.931
0.005  0.033  0.938  0.024
0.448448448448448    0.384384384384384    0.166166166166166    0.001001001001001
>GSCTGTCACTCA    PBX1(Homeobox)/MCF7-PBX1-ChIP-Seq(GSE28007)/Homer    8.655388    -1942.998760
0    T:879.0(4.38%),B:2.9(0.28%),P:1e-843    Tpos:98.2,Tstd:51.6,Bpos:110.0,Bstd:34.0,StrandBias:-
0.1,Multiplicity:1.04

```

0.185	0.033	0.686	0.096			
0.008	0.534	0.428	0.030			
0.001	0.988	0.010	0.001			
0.022	0.001	0.001	0.976			
0.001	0.001	0.997	0.001			
0.003	0.074	0.001	0.922			
0.001	0.996	0.002	0.001			
0.981	0.002	0.001	0.016			
0.284	0.495	0.216	0.004			
0.001	0.016	0.014	0.969			
0.002	0.994	0.003	0.001			
0.979	0.002	0.001	0.018			
>CCTGTCAATCAN Pbx3(Homeobox)/GM12878-PBX3-ChIP-Seq/Homer				8.248480	-3.747190e+03	0
50001.1,3840.0,1545.5,1332.0,0.00e+00						
0.145	0.386	0.366	0.103			
0.108	0.734	0.116	0.043			
0.042	0.051	0.001	0.906			
0.010	0.001	0.988	0.001			
0.001	0.201	0.001	0.797			
0.001	0.997	0.001	0.001			
0.914	0.050	0.001	0.035			
0.417	0.379	0.180	0.024			
0.001	0.001	0.001	0.997			
0.001	0.996	0.002	0.001			
0.880	0.035	0.001	0.084			
0.314	0.265	0.223	0.198			
>TCATCAATCA Pdx1(Homeobox)/Islet-Pdx1-ChIP-Seq/Homer				5.932256	-5.335033e+02	0
49999.4,2592.0,1628.2,489.0,2.01e-232						
0.092	0.404	0.076	0.428			
0.272	0.662	0.021	0.045			
0.965	0.022	0.012	0.001			
0.001	0.001	0.001	0.997			
0.038	0.478	0.032	0.452			
0.797	0.031	0.001	0.171			
0.895	0.047	0.048	0.010			
0.001	0.001	0.024	0.974			
0.001	0.963	0.002	0.034			
0.873	0.001	0.001	0.125			
>TGACCTTTGCCCA PPARE(NR/DR1)/3T3L1-Pparg-ChIP-Seq/Homer				6.968824	-3.087186e+03	0
50000.0,6530.0,7756.8,3908.0,0.00e+00						
0.084	0.043	0.060	0.813			
0.092	0.128	0.718	0.062			
0.610	0.245	0.106	0.039			
0.109	0.847	0.009	0.035			
0.026	0.916	0.001	0.057			
0.006	0.112	0.002	0.880			
0.027	0.267	0.038	0.668			
0.014	0.015	0.020	0.952			
0.070	0.165	0.690	0.075			
0.233	0.495	0.135	0.137			
0.027	0.827	0.017	0.129			
0.108	0.788	0.002	0.102			
0.069	0.598	0.016	0.317			
0.654	0.129	0.100	0.117			

>AGGTCTCTAACC PRDM14(Zf)/H1-PRDM14-ChIP-Seq/Homer 7.194475 -6.574707e+03 0
50000.0,13792.0,4687.1,4050.0,0.00e+00

0.510 0.011 0.380 0.099
0.027 0.021 0.942 0.009
0.012 0.033 0.908 0.047
0.075 0.012 0.155 0.758
0.006 0.924 0.020 0.050
0.056 0.273 0.001 0.669
0.001 0.832 0.081 0.085
0.001 0.001 0.001 0.997
0.997 0.001 0.001 0.001
0.765 0.034 0.027 0.174
0.065 0.587 0.224 0.124
0.130 0.417 0.072 0.381

>CGGAAGTGAAAC PU.1-IRF/Bcell-PU.1-ChIP-Seq/Homer 6.851957 -1.496398e+03 0
82930.0,41465.0,11784.0,7928.0,0.00e+00

0.318 0.406 0.201 0.076
0.287 0.002 0.677 0.034
0.057 0.006 0.936 0.001
0.988 0.005 0.004 0.004
0.968 0.001 0.002 0.029
0.203 0.140 0.646 0.011
0.149 0.036 0.063 0.751
0.018 0.013 0.965 0.004
0.853 0.052 0.078 0.017
0.753 0.118 0.117 0.012
0.774 0.035 0.153 0.038
0.111 0.532 0.288 0.069

>AGAGGAAGTG PU.1(ETS)/ThioMac-PU.1-ChIP-Seq/Homer 7.613173 -1.947446e+04 0
58623.0,41203.0,17624.1,16914.0,0.00e+00

0.643 0.001 0.149 0.207
0.122 0.171 0.706 0.002
0.830 0.012 0.157 0.001
0.001 0.001 0.997 0.001
0.001 0.001 0.997 0.001
0.997 0.001 0.001 0.001
0.990 0.001 0.001 0.008
0.024 0.074 0.900 0.001
0.001 0.005 0.001 0.993
0.079 0.097 0.773 0.051

>AGGTCAAGGTCA RARg(NR)/ES-RARg-ChIP-Seq(GSE30538)/Homer 11.126594 -118.145370 0
T:61.0(5.34%),B:151.6(0.32%),P:1e-51 Tpos:100.2,Tstd:35.5,Bpos:98.3,Bstd:56.1,StrandBias:0.1,Multiplicity:1.18

0.681 0.001 0.301 0.017
0.027 0.001 0.951 0.021
0.001 0.001 0.727 0.271
0.066 0.013 0.001 0.920
0.001 0.908 0.074 0.017
0.997 0.001 0.001 0.001
0.816 0.001 0.166 0.017
0.126 0.001 0.872 0.001
0.001 0.001 0.997 0.001
0.083 0.212 0.264 0.440
0.001 0.808 0.070 0.121
0.793 0.001 0.121 0.085

>GGAGCTGTCCATGGTGCTGA	REST-NRSF(Zf)/Jurkat-NRSF-ChIP-Seq/Homer	12.268316	-
6.591589e+03 0 14999.0,4244.0,1558.5,1549.0,0.00e+00			
0.112 0.019 0.771 0.098			
0.062 0.130 0.716 0.093			
0.448 0.303 0.037 0.213			
0.108 0.066 0.717 0.109			
0.006 0.966 0.007 0.021			
0.002 0.002 0.001 0.995			
0.048 0.121 0.822 0.009			
0.038 0.014 0.065 0.883			
0.001 0.985 0.001 0.013			
0.001 0.980 0.001 0.018			
0.579 0.080 0.212 0.130			
0.165 0.086 0.125 0.624			
0.041 0.008 0.931 0.019			
0.001 0.002 0.996 0.001			
0.002 0.003 0.001 0.994			
0.066 0.159 0.701 0.074			
0.010 0.953 0.014 0.023			
0.012 0.020 0.010 0.958			
0.040 0.013 0.914 0.033			
0.761 0.075 0.135 0.029			
>GTAGGTCACCTGGGTCA	Reverb(NR/DR2)/BLRP(RAW)-Reverba-ChIP-Seq/Homer	8.709195	-
4.586519e+02 0 49999.9,2934.0,795.2,323.0,6.46e-200			
0.102 0.300 0.464 0.135			
0.105 0.155 0.139 0.600			
0.540 0.011 0.429 0.021			
0.094 0.007 0.841 0.057			
0.012 0.025 0.933 0.029			
0.022 0.029 0.136 0.814			
0.032 0.927 0.012 0.029			
0.936 0.011 0.004 0.049			
0.012 0.511 0.404 0.073			
0.056 0.063 0.071 0.811			
0.231 0.004 0.761 0.004			
0.086 0.001 0.759 0.155			
0.025 0.040 0.932 0.003			
0.147 0.102 0.144 0.607			
0.011 0.895 0.052 0.042			
0.865 0.033 0.035 0.066			
>GGTTGCCATGGCAA	Rfx1(HTH)/NPC-Rfx1-ChIP-Seq/Homer	8.058432	-4.056616e+03 0
49998.8,17909.0,4678.5,3805.0,0.00e+00			
0.139 0.180 0.397 0.284			
0.023 0.004 0.971 0.001			
0.005 0.085 0.022 0.889			
0.048 0.191 0.025 0.735			
0.107 0.075 0.670 0.148			
0.001 0.930 0.006 0.063			
0.001 0.893 0.001 0.105			
0.827 0.070 0.031 0.072			
0.081 0.036 0.080 0.803			
0.093 0.001 0.905 0.001			
0.066 0.007 0.925 0.002			
0.163 0.599 0.095 0.143			

<http://biowhat.ucsd.edu/homer/custom.motifs>[6/22/12 17:50:39]

0.997	0.001	0.001	0.001				
0.332	0.146	0.324	0.198				
0.352	0.213	0.211	0.224				
0.249	0.303	0.207	0.241				
>GCTGTGGTTT RUNX-AML(Runt)/CD4+-PolII-ChIP-Seq/Homer				7.385715	-1.017665e+03	0	
91747.1,10405.0,7588.6,2259.0,0.00e+00							
0.070	0.223	0.453	0.255				
0.117	0.600	0.048	0.235				
0.002	0.006	0.001	0.991				
0.001	0.001	0.997	0.001				
0.004	0.009	0.025	0.962				
0.002	0.001	0.994	0.003				
0.001	0.019	0.948	0.032				
0.001	0.008	0.045	0.946				
0.034	0.161	0.140	0.666				
0.410	0.057	0.100	0.433				
>TAGGGCAAAGGTCA RXR(NR/DR1)/3T3L1-RXR-ChIP-Seq/Homer				6.641182	-2.545959e+03	0	
49998.4,7850.0,8696.3,4154.0,0.00e+00							
0.107	0.233	0.178	0.482				
0.580	0.027	0.320	0.073				
0.089	0.005	0.845	0.061				
0.122	0.012	0.851	0.015				
0.109	0.130	0.598	0.164				
0.064	0.740	0.159	0.037				
0.950	0.023	0.015	0.012				
0.635	0.044	0.298	0.022				
0.876	0.003	0.101	0.020				
0.036	0.003	0.944	0.017				
0.026	0.008	0.893	0.074				
0.031	0.134	0.244	0.591				
0.060	0.761	0.111	0.068				
0.870	0.043	0.048	0.039				
>ANCAGCTG SCL/HPC7-Scl-ChIP-Seq/Homer				5.100492	-4.653757e+02	0	
50000.9,1774.0,11903.4,1309.0,7.76e-203							
0.435	0.178	0.199	0.188				
0.261	0.239	0.379	0.121				
0.011011011011011	0.982982982982983	0.003003003003003	0.003003003003003				
0.672	0.047	0.044	0.237				
0.01	0.039	0.704	0.247				
0.258	0.66	0.078	0.004				
0.23	0.035	0.008	0.727				
0.037037037037037	0.005005005005005	0.764764764764765	0.193193193193193				
>CCAAAAGGG SEP3(MADS)/Arabidopsis-Flower-Sep3-ChIP-Seq/Homer				6.193342	-1.051932e+03	0	
49999.0,9325.0,16689.1,5654.0,0.00e+00							
0.105	0.831	0.024	0.040				
0.181	0.699	0.028	0.092				
0.886	0.051	0.016	0.047				
0.932	0.016	0.014	0.038				
0.976	0.003	0.003	0.018				
0.930	0.003	0.016	0.051				
0.869	0.016	0.038	0.077				
0.291	0.014	0.451	0.245				
0.127	0.009	0.849	0.015				
0.124	0.047	0.772	0.057				

>CCCATTGTTC Sox2(HMG)/mES-Sox2-ChIP-Seq/Homer 6.785470 -1.702649e+03 0
20000.0,2266.0,3527.6,1915.0,0.00e+00
0.060 0.378 0.287 0.275
0.005 0.838 0.073 0.084
0.004 0.766 0.001 0.229
0.692 0.001 0.001 0.306
0.001 0.001 0.001 0.997
0.001 0.001 0.001 0.997
0.001 0.001 0.997 0.001
0.001 0.001 0.001 0.997
0.001 0.223 0.051 0.725
0.121 0.592 0.080 0.207

>CCWTTGTY Sox3(HMG)/NPC-Sox3-ChIP-Seq(GSE33059)/Homer 6.563838 -8049.249245 0
T:10715.0(60.64%),B:5489.6(17.74%),P:1e-3495
0.019 0.695 0.147 0.139
0.032 0.705 0.003 0.260
0.481 0.001 0.001 0.517
0.001 0.001 0.001 0.997
0.001 0.001 0.001 0.997
0.007 0.027 0.843 0.123
0.056 0.001 0.113 0.830
0.076 0.330 0.185 0.409

>CCATTGTTNY Sox6(HMG)/Myotubes-Sox6-ChIP-Seq(GSE32627)/Homer 6.331137 -1012.113794 0
T:2244.0(40.71%),B:6639.4(15.51%),P:1e-439
0.074 0.489 0.235 0.202
0.240 0.448 0.120 0.191
0.533 0.017 0.155 0.295
0.003 0.014 0.013 0.970
0.026 0.019 0.013 0.942
0.018 0.068 0.909 0.005
0.006 0.024 0.011 0.959
0.107 0.228 0.180 0.484
0.199 0.305 0.241 0.254
0.135 0.317 0.190 0.358

>GGCCCCGCCCCC Sp1(Zf)/Promoter/Homer 8.685072 -2.306247e+03 0
149303.9,11385.0,33332.5,6807.0,0.00e+00
0.181 0.218 0.541 0.060
0.050 0.001 0.948 0.001
0.013 0.870 0.061 0.056
0.033 0.815 0.001 0.151
0.049 0.949 0.001 0.001
0.001 0.997 0.001 0.001
0.025 0.001 0.973 0.001
0.001 0.997 0.001 0.001
0.001 0.997 0.001 0.001
0.001 0.881 0.001 0.117
0.103 0.824 0.001 0.071
0.037 0.512 0.221 0.230

>ACATCCTGGT SPDEF(ETS)/VCaP-SPDEF-ChIP-Seq/Homer 6.871950 -3.587168e+02 0
49998.6,3295.0,1371.2,401.0,1.63e-156
0.625 0.001 0.373 0.001
0.072 0.460 0.316 0.152
0.569 0.001 0.001 0.429
0.001 0.001 0.001 0.997

0.001	0.997	0.001	0.001				
0.001	0.997	0.001	0.001				
0.001	0.001	0.361	0.637				
0.001	0.001	0.916	0.082				
0.046	0.286	0.389	0.278				
0.303	0.071	0.001	0.625				
>ATCACCCCAT Srebp1a(HLH)/HepG2-Srebp1a-ChIP-Seq/Homer				7.911684	-1.844166e+02	0	
50000.0,1441.0,919.8,167.0,8.11e-81							
0.485	0.157	0.353	0.005				
0.014	0.007	0.001	0.978				
0.001	0.968	0.030	0.001				
0.972	0.001	0.017	0.011				
0.001	0.892	0.089	0.018				
0.238	0.400	0.305	0.057				
0.001	0.997	0.001	0.001				
0.035	0.931	0.001	0.033				
0.965	0.012	0.015	0.008				
0.007	0.423	0.039	0.532				
>CGGTCACGCCAC Srebp2(HLH)/HepG2-Srebp2-ChIP-Seq/Homer				8.765329	-1.042700e+02	0	
49999.0,1004.0,622.4,88.0,5.20e-46							
0.328	0.566	0.001	0.105				
0.235	0.138	0.398	0.229				
0.338	0.139	0.521	0.001				
0.001	0.010	0.001	0.988				
0.013	0.960	0.026	0.001				
0.997	0.001	0.001	0.001				
0.001	0.802	0.196	0.001				
0.191	0.375	0.433	0.001				
0.001	0.997	0.001	0.001				
0.025	0.961	0.001	0.013				
0.951	0.047	0.001	0.001				
0.001	0.703	0.102	0.194				
>CCATATATGGNA CArG(MADS)/PUER-Srf-ChIP-Seq/Homer				6.911457	-3.735581e+03	0	
49999.7,9076.0,4011.2,2872.0,0.00e+00							
0.001	0.994	0.003	0.001				
0.001	0.947	0.001	0.051				
0.557	0.037	0.004	0.402				
0.288	0.014	0.017	0.680				
0.786	0.013	0.041	0.160				
0.170	0.048	0.012	0.771				
0.693	0.017	0.012	0.278				
0.405	0.003	0.017	0.575				
0.060	0.001	0.938	0.001				
0.003	0.001	0.994	0.003				
0.266	0.302	0.230	0.202				
0.413	0.341	0.104	0.142				
>ATTTCCCAGVAKSCY ZNF143 STAF(Zf)/CUTLL-ZNF143-ChIP-Seq(GSE29600)/Homer				8.623605	-		
6988.764358 0 T:3019.0(47.50%),B:44.9(2.41%),P:1e-3035							
Tpos:100.0,Tstd:26.1,Bpos:99.0,Bstd:65.5,StrandBias:-0.1,Multiplicity:1.12							
0.536	0.180	0.148	0.136				
0.263	0.177	0.135	0.425				
0.063	0.340	0.087	0.510				
0.182	0.052	0.011	0.755				
0.001	0.997	0.001	0.001				

0.001	0.997	0.001	0.001
0.001	0.997	0.001	0.001
0.936	0.001	0.059	0.004
0.042	0.114	0.527	0.317
0.367	0.307	0.313	0.013
0.804	0.045	0.148	0.003
0.078	0.022	0.410	0.490
0.028	0.380	0.484	0.107
0.001	0.997	0.001	0.001
0.234	0.386	0.015	0.366
>NATTTCCNGGAAAT STAT1(Stat)/HelaS3-STAT1-ChIP-Seq/Homer 7.984291 -3.581576e+03 0			
16324.0,4675.0,6674.8,5871.0,0.00e+00			
0.178	0.319	0.294	0.209
0.411	0.163	0.228	0.198
0.154	0.226	0.108	0.512
0.001	0.001	0.001	0.997
0.001	0.001	0.001	0.997
0.139	0.842	0.001	0.018
0.012	0.715	0.001	0.273
0.310	0.186	0.190	0.315
0.267	0.001	0.709	0.023
0.024	0.001	0.839	0.136
0.997	0.001	0.001	0.001
0.997	0.001	0.001	0.001
0.508	0.108	0.232	0.152
0.191	0.222	0.173	0.413
>CNCTTCCNGGAAGN Stat3+il23(Stat)/CD4-Stat3-ChIP-Seq/Homer 6.747055 -1.837800e+03 0			
50000.4,4858.0,2431.0,1358.0,0.00e+00			
0.161	0.362	0.306	0.170
0.322	0.220	0.293	0.165
0.061	0.421	0.169	0.349
0.001	0.022	0.001	0.976
0.001	0.001	0.159	0.839
0.162	0.806	0.012	0.021
0.001	0.809	0.001	0.189
0.298	0.210	0.205	0.287
0.207	0.001	0.791	0.001
0.020	0.004	0.821	0.155
0.842	0.156	0.001	0.001
0.976	0.001	0.018	0.005
0.331	0.178	0.424	0.066
0.163	0.320	0.215	0.302
>CTTCCGGGAA Stat3(Stat)/mES-Stat3-ChIP-Seq/Homer 7.063880 -2.441699e+03 0			
50000.1,3523.0,5759.5,2465.0,0.00e+00			
0.134	0.473	0.159	0.234
0.037	0.039	0.033	0.890
0.044	0.014	0.142	0.800
0.081	0.836	0.036	0.047
0.013	0.848	0.024	0.116
0.188	0.230	0.389	0.194
0.101	0.031	0.851	0.016
0.036	0.038	0.838	0.089
0.816	0.113	0.031	0.040
0.890	0.027	0.036	0.047

>NTTTCAGGAAA STAT4(Stat)/CD4-Stat4-ChIP-Seq/Homer 5.220249 -3.020509e+03 0
50000.4,5315.0,4095.3,2405.0,0.00e+00

0.277 0.235 0.317 0.171
0.038 0.379 0.092 0.491
0.001 0.001 0.001 0.997
0.001 0.001 0.048 0.950
0.074 0.920 0.001 0.005
0.022 0.791 0.001 0.186
0.359 0.137 0.148 0.356
0.187 0.001 0.794 0.018
0.004 0.001 0.917 0.077
0.961 0.037 0.001 0.001
0.997 0.001 0.001 0.001
0.477 0.102 0.396 0.025

>ATTTCTNAGAAA STAT5(Stat)/mCD4+-Stat5a|b-ChIP-Seq/Homer 6.285419 -4.825146e+03 0
50001.0,5612.0,6344.2,4029.0,0.00e+00

0.379 0.156 0.238 0.227
0.163 0.196 0.203 0.438
0.001 0.001 0.001 0.997
0.001 0.001 0.001 0.997
0.001 0.997 0.001 0.001
0.008 0.312 0.001 0.679
0.273 0.219 0.224 0.284
0.694 0.001 0.297 0.008
0.001 0.001 0.997 0.001
0.997 0.001 0.001 0.001
0.997 0.001 0.001 0.001
0.440 0.187 0.208 0.165

>TTCKNAGAA STAT6/Macrophage-Stat6-ChIP-Seq/Homer 6.480479 -2078.479860 0
T:1345.0(52.11%),B:2783.0(6.00%),P:1e-902

0.005 0.007 0.003 0.985
0.005 0.004 0.005 0.986
0.006 0.963 0.005 0.026
0.021 0.824 0.005 0.150
0.165 0.188 0.270 0.376
0.237 0.274 0.222 0.267
0.758 0.007 0.199 0.036
0.005 0.008 0.983 0.004
0.977 0.006 0.008 0.009
0.984 0.007 0.004 0.005

>ANTTCTTAAGAA STAT6(Stat)/CD4-Stat6-ChIP-Seq/Homer 7.328792 -2.341495e+03 0
49999.6,3953.0,2213.9,1380.0,0.00e+00

0.405 0.173 0.244 0.178
0.156 0.268 0.245 0.331
0.001 0.001 0.001 0.997
0.001 0.001 0.001 0.997
0.001 0.993 0.001 0.005
0.115 0.429 0.001 0.455
0.196 0.253 0.174 0.377
0.361 0.184 0.271 0.184
0.439 0.001 0.434 0.126
0.063 0.001 0.935 0.001
0.997 0.001 0.001 0.001
0.997 0.001 0.001 0.001

>ACTTTCGTTTCT T1ISRE(IRF)/Ifnb-Exp/Homer 11.711998 -4.407130e+01 0
87179.0,1722.0,143.2,27.0,7.25e-20
0.924 0.039 0.036 0.001
0.001 0.856 0.142 0.001
0.001 0.001 0.001 0.997
0.001 0.001 0.001 0.997
0.001 0.001 0.001 0.997
0.001 0.898 0.052 0.049
0.215 0.001 0.783 0.001
0.001 0.001 0.001 0.997
0.001 0.001 0.001 0.997
0.001 0.059 0.001 0.939
0.052 0.799 0.001 0.148
0.001 0.107 0.046 0.847

>CCTTTTATAGCC TATA-Box(TBP)/Promoter/Homer 5.682591 -3.995489e+02 0
124715.9,9565.0,11407.0,1769.0,0.00e+00
0.025 0.606 0.283 0.086
0.006 0.850 0.004 0.140
0.091 0.125 0.001 0.783
0.001 0.048 0.001 0.950
0.297 0.001 0.001 0.701
0.234 0.001 0.001 0.764
0.964 0.034 0.001 0.001
0.351 0.001 0.172 0.476
0.704 0.009 0.285 0.001
0.042 0.200 0.716 0.042
0.178 0.355 0.328 0.138
0.117 0.510 0.304 0.069

>GGTGYTGACAGS Tbx20(T-box)/Heart-Tbx20-ChIP-Seq(GSE29636)/Homer 8.538709 -1829.738317 0
T:898.0(41.46%),B:1225.3(2.65%),P:1e-794
Tpos:100.6,Tstd:29.5,Bpos:99.6,Bstd:65.5,StrandBias:0.2,Multiplicity:1.07
0.271 0.165 0.431 0.133
0.077 0.250 0.583 0.090
0.040 0.184 0.001 0.775
0.001 0.001 0.997 0.001
0.165 0.337 0.103 0.395
0.010 0.008 0.041 0.941
0.014 0.001 0.984 0.001
0.956 0.004 0.018 0.022
0.222 0.747 0.011 0.020
0.987 0.001 0.001 0.011
0.158 0.040 0.673 0.129
0.108 0.355 0.428 0.109

>AGGTGTCA Tbx5(T-box)/HL1-Tbx5.biotin-ChIP-Seq/Homer 5.412312 -4385.796453 0
T:31570.0(53.59%),B:17487.0(34.71%),P:1e-1901
0.786 0.002 0.139 0.074
0.168 0.021 0.778 0.034
0.002 0.143 0.842 0.014
0.002 0.083 0.002 0.914
0.002 0.002 0.995 0.002
0.159 0.226 0.066 0.550
0.016 0.495 0.231 0.260
0.719 0.022 0.133 0.127

>ACAGCTGCTG Tcf12(HLH)/GM12878-Tcf12-ChIP-Seq/Homer 7.146225 -1.923082e+03 0

49999.2,3518.0,4153.7,1772.0,0.00e+00

0.376 0.258 0.359 0.006

0.001 0.997 0.001 0.001

0.997 0.001 0.001 0.001

0.001 0.107 0.891 0.001

0.001 0.974 0.024 0.001

0.111 0.001 0.001 0.887

0.001 0.001 0.997 0.001

0.001 0.430 0.244 0.325

0.137 0.182 0.162 0.519

0.157 0.206 0.424 0.212

>ACATCAAAGG Tcf3(HMG)/mES-Tcf3-ChIP-Seq/Homer 8.529040 -1.785733e+03 0

19998.1,4459.0,1668.2,1298.0,0.00e+00

0.809 0.012 0.104 0.075

0.011 0.444 0.325 0.220

0.502 0.016 0.001 0.481

0.038 0.001 0.001 0.960

0.001 0.916 0.082 0.001

0.997 0.001 0.001 0.001

0.997 0.001 0.001 0.001

0.997 0.001 0.001 0.001

0.084 0.001 0.914 0.001

0.122 0.314 0.519 0.046

>ACATCAAAGGGA Tcf4(HMG)/Hct116-Tcf4-ChIP-Seq/Homer 7.170992 -7.836161e+03 0

73006.0,36503.0,8897.4,7814.0,0.00e+00

0.832 0.013 0.097 0.058

0.005 0.501 0.485 0.009

0.737 0.002 0.001 0.260

0.022 0.002 0.001 0.975

0.001 0.901 0.095 0.003

0.997 0.001 0.001 0.001

0.989 0.005 0.005 0.001

0.995 0.001 0.003 0.001

0.058 0.001 0.940 0.001

0.203 0.135 0.592 0.070

0.280 0.307 0.356 0.057

0.445 0.197 0.165 0.193

>ACWTCAAAGG TCFL2(HMG)/K562-TCF7L2-ChIP-Seq(GSE29196)/Homer 7.790900 -1730.715496 0
T:503.0(6.34%),B:1.9(0.15%),P:1e-751 Tpos:101.3,Tstd:31.5,Bpos:96.6,Bstd:38.0,StrandBias:0.1,Multiplicity:1.01

0.997 0.001 0.001 0.001

0.001 0.690 0.308 0.001

0.497 0.001 0.001 0.501

0.001 0.001 0.001 0.997

0.001 0.997 0.001 0.001

0.997 0.001 0.001 0.001

0.997 0.001 0.001 0.001

0.997 0.001 0.001 0.001

0.019 0.001 0.979 0.001

0.001 0.197 0.801 0.001

>NAAACCGGTTCAAACCGGTT Tcfcp2l1(CP2)/mES-Tcfcp2l1-ChIP-Seq/Homer 8.040869 -

8.848373e+03 0 15000.0,4418.0,4521.6,4367.0,0.00e+00

0.268 0.295 0.264 0.173

0.423 0.157 0.294 0.126

0.578 0.053 0.277 0.092

0.469	0.047	0.304	0.180				
0.001	0.965	0.033	0.001				
0.096	0.596	0.001	0.307				
0.418	0.001	0.494	0.087				
0.003	0.019	0.978	0.001				
0.187	0.315	0.023	0.474				
0.077	0.285	0.053	0.584				
0.178	0.363	0.152	0.306				
0.362	0.179	0.305	0.154				
0.627	0.053	0.251	0.069				
0.494	0.019	0.298	0.189				
0.001	0.978	0.020	0.001				
0.070	0.619	0.001	0.310				
0.411	0.001	0.475	0.114				
0.001	0.015	0.983	0.001				
0.189	0.324	0.043	0.444				
0.113	0.296	0.048	0.543				
>NCTGGAATGC TEAD(TEA)/Fibroblast-PU.1-ChIP-Seq/Homer				8.290020	-1.899497e+02	0	
32356.8,1450.0,1492.8,269.0,3.21e-83							
0.208791208791209	0.35964035964036	0.183816183816184	0.247752247752248				
0.304	0.458	0.001	0.237				
0.46	0.001	0.001	0.538				
0.001	0.001	0.997	0.001				
0.001	0.001	0.997	0.001				
0.997	0.001	0.001	0.001				
0.997	0.001	0.001	0.001				
0.001	0.001	0.001	0.997				
0.001	0.022	0.569	0.408				
0.154	0.441	0.001	0.404				
>CTGGCAGGCTGCCA Tlx?/NPC-H3K4me1-ChIP-Seq/Homer				7.865087	-3.753871e+03	0	
82384.0,41192.0,29182.4,20020.0,0.00e+00							
0.068	0.580	0.075	0.278				
0.117	0.001	0.029	0.853				
0.001	0.001	0.994	0.004				
0.017	0.043	0.921	0.018				
0.159	0.795	0.045	0.001				
0.564	0.043	0.001	0.391				
0.134	0.132	0.553	0.180				
0.147	0.310	0.382	0.161				
0.157	0.584	0.156	0.102				
0.403	0.001	0.014	0.582				
0.001	0.031	0.778	0.190				
0.018	0.906	0.059	0.017				
0.002	0.996	0.001	0.001				
0.702	0.158	0.001	0.139				
>GAGGTCAAAGGTCA TR4(NR/DR1)/Hela-TR4-ChIP-Seq/Homer				9.352371	-1.413580e+03	0	
50001.0,2795.0,1078.2,671.0,0.00e+00							
0.292	0.120	0.531	0.057				
0.657	0.008	0.322	0.014				
0.068	0.002	0.911	0.019				
0.024	0.005	0.869	0.101				
0.015	0.058	0.106	0.821				
0.019	0.814	0.112	0.055				
0.972	0.002	0.021	0.005				

0.566	0.015	0.400	0.019				
0.899	0.001	0.099	0.001				
0.022	0.002	0.970	0.007				
0.013	0.008	0.857	0.122				
0.001	0.048	0.131	0.819				
0.029	0.818	0.080	0.073				
0.905	0.014	0.061	0.021				
>GGTCACGTGA USF1(HLH)/GM12878-Usf1-ChIP-Seq/Homer 6.629484 -3.903157e+03 0							
49999.3,1770.0,2858.4,2055.0,0.00e+00							
0.169	0.281	0.417	0.133				
0.229	0.156	0.582	0.033				
0.037	0.312	0.141	0.510				
0.001	0.996	0.002	0.001				
0.997	0.001	0.001	0.001				
0.001	0.946	0.001	0.052				
0.063	0.005	0.931	0.001				
0.001	0.001	0.002	0.996				
0.001	0.001	0.997	0.001				
0.447	0.144	0.347	0.062				
>NGAGGTCANNAGAGTTCANN VDR(NR/DR3)/GM10855-VDR+vitD-ChIP-Seq/Homer 8.330473 -							
1.987790e+03 0 50001.0,3386.0,1163.5,849.0,0.00e+00							
0.386613386613387	0.16983016983017	0.224775224775225	0.218781218781219				
0.387	0.097	0.446	0.07				
0.612	0.001	0.385	0.002				
0.064	0.001	0.919	0.016				
0.019	0.002	0.743	0.236				
0.01	0.046	0.149	0.795				
0.012	0.798	0.122	0.068				
0.935	0.013	0.022	0.03				
0.219219219219219	0.286286286286286	0.263263263263263	0.231231231231231				
0.282	0.184	0.155	0.379				
0.20979020979021	0.0809190809190809	0.679320679320679	0.02997002997003				
0.591	0.001	0.407	0.001				
0.001	0.001	0.994	0.004				
0.001	0.001	0.361	0.637				
0.0879120879120879	0.0709290709290709	0.185814185814186	0.655344655344655				
0.023	0.783	0.036	0.158				
0.762	0.07	0.08	0.088				
0.222	0.209	0.234	0.335				
0.287712287712288	0.182817182817183	0.2997002997003	0.22977022977023				
0.243	0.189	0.316	0.252				
>GGTTGCCATGGCAA X-box(HTH)/NPC-H3K4me1-ChIP-Seq/Homer 9.021106 -2.712771e+03 0							
82384.0,41192.0,8738.9,6587.0,0.00e+00							
0.170	0.071	0.470	0.290				
0.036	0.018	0.921	0.025				
0.009	0.051	0.016	0.924				
0.055	0.057	0.023	0.865				
0.139	0.057	0.591	0.213				
0.005	0.863	0.025	0.108				
0.007	0.869	0.001	0.123				
0.871	0.053	0.011	0.066				
0.073	0.012	0.063	0.852				
0.120	0.001	0.875	0.003				
0.096	0.027	0.873	0.004				

0.219	0.541	0.076	0.164				
0.789	0.024	0.126	0.061				
0.892	0.031	0.054	0.023				
>CAAGATGGCGGC YY1(Zf)/Promoter/Homer				8.783400	-7.129795e+02	0	
149303.9,11385.0,2727.7,842.0,0.00e+00							
0.102	0.696	0.170	0.032				
0.997	0.001	0.001	0.001				
0.995	0.001	0.003	0.001				
0.147	0.160	0.661	0.032				
0.997	0.001	0.001	0.001				
0.001	0.001	0.001	0.997				
0.001	0.001	0.997	0.001				
0.001	0.001	0.997	0.001				
0.001	0.980	0.001	0.018				
0.057	0.075	0.815	0.053				
0.045	0.004	0.942	0.009				
0.097	0.852	0.013	0.039				
>GGNTCTCGCGAGAAC ZBTB33/GM12878-ZBTB33-ChIP-Seq/Homer				7.648064	-6.073430e+02	0	
50000.5,504.0,217.1,155.0,1.72e-264							
0.129	0.168	0.604	0.099				
0.218	0.149	0.564	0.069				
0.347	0.267	0.218	0.168				
0.03	0.069	0.04	0.861				
0.089	0.88	0.03	0.001				
0.00999000999000999	0.0989010989010989	0.04995004995005	0.841158841158841				
0.001	0.997	0.001	0.001				
0.001	0.001	0.978	0.02				
0.01	0.979	0.001	0.01				
0.001	0.001	0.997	0.001				
0.861861861861862	0.0590590590590591	0.0690690690690691	0.01001001001001				
0.001001001001001	0.0490490490490491	0.870870870870871	0.0790790790790791				
0.91	0.001	0.079	0.01				
0.406	0.188	0.208	0.198				
0.089	0.545	0.188	0.178				
>AGGCCTGG ZFX(Zf)/mES-Zfx-ChIP-Seq/Homer				6.248446	-4.819637e+03	0	
49993.3,14011.0,21123.2,12598.0,0.00e+00							
0.922	0.001	0.076	0.001				
0.001	0.001	0.997	0.001				
0.001	0.001	0.997	0.001				
0.001	0.997	0.001	0.001				
0.001	0.997	0.001	0.001				
0.001	0.113	0.001	0.885				
0.358	0.138	0.360	0.144				
0.188	0.198	0.533	0.081				
>CNGTCCTCCC Znf263(Zf)/K562-Znf263-ChIP-Seq/Homer				3.877224	-3.564870e+03	0	
62784.0,31392.0,27524.2,19041.0,0.00e+00							
0.152	0.413	0.242	0.193				
0.333	0.240	0.309	0.118				
0.005	0.334	0.660	0.001				
0.001	0.001	0.001	0.997				
0.001	0.512	0.486	0.001				
0.001	0.997	0.001	0.001				
0.001	0.001	0.001	0.997				
0.001	0.997	0.001	0.001				

0.001	0.997	0.001	0.001				
0.138	0.658	0.002	0.203				
>AGGCCTAG				ZNF711(Zf)/SH-SY5Y-ZNF711-ChIP-Seq/Homer	6.101929	-2.499934e+02	0
49999.6,1243.0,5857.4,513.0,2.69e-109							
0.588	0.220	0.130	0.063				
0.001	0.001	0.996	0.002				
0.004	0.036	0.959	0.001				
0.001	0.981	0.017	0.001				
0.001	0.997	0.001	0.001				
0.042	0.174	0.001	0.783				
0.470	0.213	0.251	0.066				
0.094	0.171	0.640	0.095				



HOMER

Software for motif discovery and next-gen sequencing analysis

Visualizing Experiments with the UCSC Genome Browser

The [UCSC Genome Browser](#) is quite possibly one of the best computational tools ever developed. Not only does it contain an incredible amount of data in a single application, it allows users to upload custom information such as data from their ChIP-Seq experiments so that they can be easily visualized and compared to other information.

Making Genome Browser Files

The basic strategy HOMER uses is to create a bedGraph formatted file that can then be uploaded as a custom track to the genome browser. This is accomplished using the **makeUCSCfile** program. To make a ucsc visualization file, type the following:

```
makeUCSCfile <tag directory> -o auto
```

i.e. **makeUCSCfile PU.1-ChIP-Seq/ -o auto**

(output file will be in the PU.1-ChIP-Seq/ folder named PU.1-ChIP-Seq.ucsc.bedGraph.gz)

The "-o auto" with make the program automatically generate an output file name (i.e. TagDirectory.ucsc.bedGraph.gz) and place it in the tag directory which helps with the organization of all these files. The output file can be named differently by specifying "-o outputfilename" or by simply omitting "-o", which will send the output of the program to *stdout* (i.e. add " > outputfile" to capture it in the file outputfile). It is recommended that you zip the file using **gzip** and directly upload the *zipped* file when loading custom tracks at UCSC.

To visualize the experiment in the UCSC Genome Browser, go to Genome Browser [page](#) and select the appropriate genome (i.e. the genome that the sequencing tags were mapped to). Then click on the "add custom tracks" button (this will read "manage custom tracks" once at least one custom track is loaded). Enter the file created earlier in the "Paste URLs or data" section and click "Submit".

Problems Loading UCSC Files

The most common problem encountered while loading UCSC files is to see "position exceeds chromosome length" or something to that effect. This is usually caused by one of two problems:

1. You are trying to load the file to the wrong genome assembly. Make sure the assembly is correct!
2. Some of your tags are mapping outside the reference chromosome - this can be caused by mapping to non-standard assemblies or by some alignment programs. To remove all reads outside of the UCSC chromosome lengths, you can run the program **removeOutOfBoundsReads.pl**.

```
removeOutOfBoundsReads.pl <tag directory> <genome>
```

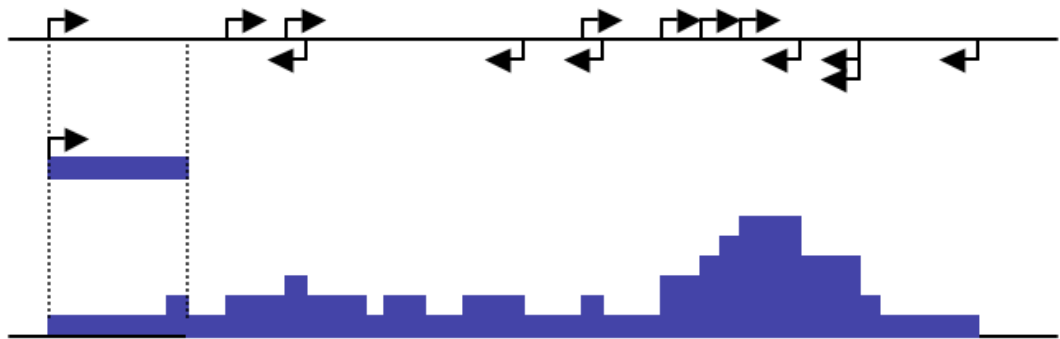
i.e. **removeOutOfBoundsReads.pl PU.1-ChIP-Seq/ mm9**

After running the program, you can rerun **makeUCSCfile**.

What does makeUCSCfile do?

The program works by approximating the ChIP-fragment density at each position in the genome. This is

done by starting with each tag and extending it by the estimated fragment length (determined by [autocorrelation](#), or it can be manually specified using "-fragLength <#>"). The ChIP-fragment density is then defined as the total number of overlapping fragments at each position in the genome. Below is a diagram that depicts how this works:



As great as the UCSC Genome Browser is, the large size of recent ChIP-Seq experiments results in custom track files that are *very* large. In addition to taking a long time to upload, the genome browser has trouble loading excessively large files. To help cope with this, the **makeUCSCfile** program works by specifying a target file size when zipped (default 50MB). In order to meet the specified target file size, makeUCSCfile merges adjacent regions of tag density levels by their weighted average to reduce the total number lines in the final bedGraph file. If you have trouble loading getting your file to load, try reducing the size of the file using the "-fsize <#>" option (i.e. "-fsize 2e7"). To force the creation of larger files, use a very large file size (i.e. "-fsize 1e50") - this will create a file that does not merge any regions and displays a "native" view of the data.

Tags can be visualized separately for each strand using the "-strand separate" option.

Changing the Resolution

By default, makeUCSCfile uses the "-fsize <#>" option to determine how many reads to essentially "skip" when making the output file. You can also manually set the resolution.

In an effort to reduce the size of large UCSC files, one attractive option is to reduce the overall resolution of the file. By default, **makeUCSCfile** will make full resolution (i.e. 1 bp) files, but this can be changed by specifying the "-res <#>" option. For example, "-res 10" will cause changes in ChIP-fragment density to be reported only every 10 bp.

Normalization of UCSC files

In order to easily compare ChIP-fragment densities between different experiments, **makeUCSCfile** will normalize density profiles based on the total number of mapped tags for each experiment. As with other programs apart of HOMER, the total number of tags is normalized to 10 million. This means that tags from an experiment with only 5 million mapped tags will count for 2 tags apiece. The total of tags to normalize to can be changed using the "-norm <#>" option.

Separating data from different strands / RNA-Seq

You can specify that HOMER separate the data based on the strand by using the "-strand <...>" option. This is useful when looking at strand-specific RNA-Seq/GRO-Seq experiments. The following options are available:

- strand both : default behavior, for ChIP-Seq/MNase-Seq etc.
- strand separate : separate data by strand, for RNA-Seq/GRO-Seq
- strand + : only show the positive strand (i.e. Watson strand) data
- strand - : only show the negative strand (i.e. crick strand) data

Creating bigWig files with HOMER

Some data sets of very large, but you still want to see all of the details from your sequencing in the UCSC Genome Browser. HOMER can produce [bigWig files](#) by running the conversion program for you (**bedGraphToBigWig**). The only catch is that [you must have access to a webserver](#) where you can post the resulting bigWig file - this is because instead of uploading the whole file to UCSC, the browser actually looks for the data file on YOUR webserver and grabs only the parts it needs. Slick, eh. Chuck uses this all the time for big experiments.

Before even trying to make bigWigs, you must download the [bedGraphToBigWig program from UCSC](#) and place it somewhere in your executable path (i.e. the /path-to-homer/bin/ folder). This called directly by HOMER to create the BigWig files.

Using the makeBigWig.pl Script

To make bigWig files easier to generate, HOMER includes a program creatively named "makeBigWig.pl" that automates all of the steps below.

```
makeBigWig.pl <tag directory> <genome> [special options] [makeUCSC file options] -
webDir /path-to-web-fold/ -url http://webserverURL/bigwigFold/
i.e. makeBigWig.pl PU.1-ChIP-Seq/ mm9 -webDir /var/www/bigWigs/ -url
http://ChuckNorrisU.edu/bigWigs/
```

If you are visualizing strand specific data (i.e. RNA-Seq), specify "-strand". The -url and -webDir are the directories are the web URL directory and file system directory where the bigWigs will be stored, respectively. Recent changes to UCSC require that the chromosome sizes be specified exactly. If having trouble, the current version of HOMER has the option "-chromSizes <filename>" so that you can specify the sizes explicitly.

Making bigWigs from scratch

This is a quick description of what HOMER is doing. To make a bigWig, add the "-bigWig <chrom.sizes file> -fsize 1e20" parameters to your makeUCSCfile command. When making a bigWig, you usually want to see all of the tag information, so make sure the "-fsize" options is large. You also need to specify an output file using "-o <bigwigfilename>" and also capture the stdout stream using "> trackfileoutput.txt". You can also use "-o auto". The "trackfileoutput.txt" will contain the header information that is uploaded as a custom track to UCSC. Recently, changes to UCSC require that HOMER know the exact size of the chromosomes when making the file - these should be placed in a file (<chrom.sizes> file). **makeBigWig.pl** and **makeMultiWigHub.pl** will generate these files automatically by analyzing the sequences in the genome directory.

After running the makeUCSCfile program with the bigWig options, you need to do the following:

1. Copy the *.bigWig file to your webserver location and make sure it is viewable over the internet.
2. Need to edit the "trackfileoutput.txt" file and enter the URL of your bigWig file (... bigDataUrl=http://server/path/bigWigFilename ...)
3. Upload the "trackfileoutput.txt" file to UCSC as a custom track to view your data.

For example:

```
makeUCSCfile <tag directory> -o auto -bigWig <chrom.sizes file> -fsize 1e20 >
trackInfo.txt
```

i.e.

```
makeUCSCfile PU.1-ChIP-Seq/ -o auto -bigWig chrom.sizes -fsize 1e20 > PU.1-
bigWig.trackInfo.txt
cp PU.1-ChIP-Seq/PU.1-ChIP-Seq.ucsc.bigWig /Web/Server/Root/Path/
** edit PU.1-bigWig.trackInfo.txt to have the right URL **
```

NOTE: As of now, a bigWig file can only be composed of a single track - if you want to separate the data by strands, do the following:

```
makeUCSCfile PU.1-ChIP-Seq/ -o PU.1.positiveStrand.bigWig -bigWig chrom.sizes -fsize
1e20 -strand + > PU.1-bigWig.trackInfo.positiveStrand.txt
makeUCSCfile PU.1-ChIP-Seq/ -o PU.1.negativeStrand.bigWig -bigWig chrom.sizes -
fsize 1e20 -strand - > PU.1-bigWig.trackInfo.negativeStrand.txt
cp PU.1.positiveStrand.bigWig PU.1.negativeStrand.bigWig /Web/Server/Root/Path/
cat PU.1-bigWig.trackInfo.positiveStrand.txt PU.1-bigWig.trackInfo.negativeStrand.txt >
PU.1-bigWig.trackInfo.both.txt
** edit PU.1-bigWig.trackInfo.both.txt to have the right URLs for both the negative and positive
strands **
```

Creating Multi-Experiment Overlay Tracks

UCSC has recently added the option to create overlay tracks, where several bigWig files can be viewed in the same space with the help of transparent colors. The first example of this was the Encode Regulation Track, which showed H3K4me1/3 data from several cell types at the same time. This is very useful for large-scale data sets with many different experiments. In these cases it is just about impossible to get them on the screen together.

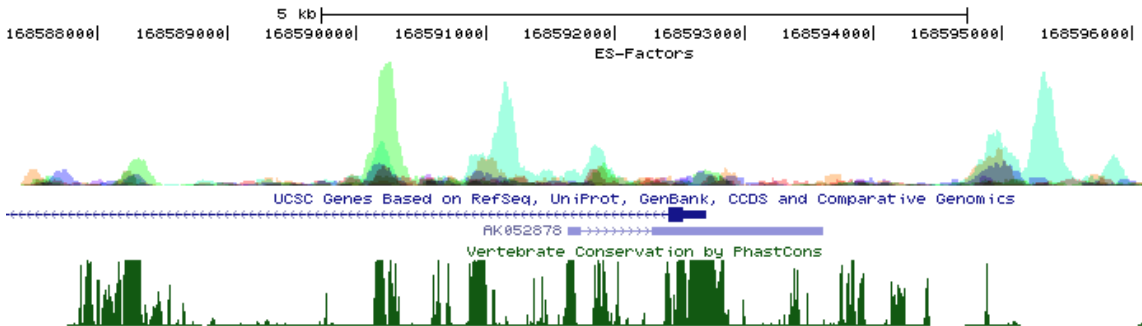
To make a "multi-wig hub", as we will refer to them, you need to make sure you have the [bedGraphToBigWig program from UCSC](#), and a working webserver to host your files. If you can handle bigWigs in the section above, you can make multi-wig hubs.

The HOMER program to handle multi-wig hubs is called **makeMultiWigHub.pl**. It works essentially the same way as the **makeBigWig.pl** script, however, the syntax is a little different. The basic usage is:

```
makeMultiWigHub.pl <hub name> <genome> [options] -d <tag directory1> <tag directory2> ...
i.e. makeMultiWigHub.pl ES-Factors mm9 -d mES-Oct4/ mES-Sox2/ mES-Nanog/ mES-Klf4/
mES-Esrrb/ mES-cMyc/ mES-Stat3/
```

NOTE: make sure you use the UCSC genome (e.g. mm9) and not the masked, bastardized HOMER version (mm9r).

The above example will produce a hub called "ES-Factors", composed of configuration files and bigWig files, and place it on your server in the directory specified by "**-webDir <directory>**". It will also provide you with a URL to the hub (dependent on the value of "**-url <base url>**"). To load the Hub, click on "Track Hubs" on the UCSC browser (next to custom tracks button), and paste the URL in to the dialog box. The example above will look something like this:



To figure out which factors correspond to which colors, click on the Blue Heading for the Hub in the settings area below the UCSC picture. Something like this should pop up:

ES-Factors Track Settings

ES-Factors

Display mode: Full Submit [Reset to defaults](#)

Overlay method: transparent

Type of graph: bar

Track height: 75 pixels (range: 11 to 100)

Vertical viewing range: min: 0 max: 127 (range: 0 to 127)

Data view scaling: auto-scale to data view Always include zero: OFF

Transform function: Transform data points by: NONE

Windowing function: maximum Smoothing window: OFF pixels

Draw y indicator lines: at y = 0.0: OFF at y = 0 OFF

[Graph configuration help](#)

List subtracks: only selected/visible all (7 of 7 selected)

☒

mES-cMyc

mES-cMyc

☒

mES-Esrrb

mES-Esrrb

☒

mES-Klf4

mES-Klf4

☒

mES-Nanog

mES-Nanog

☒

mES-Oct4

mES-Oct4

☒

mES-Sox2

mES-Sox2

☒

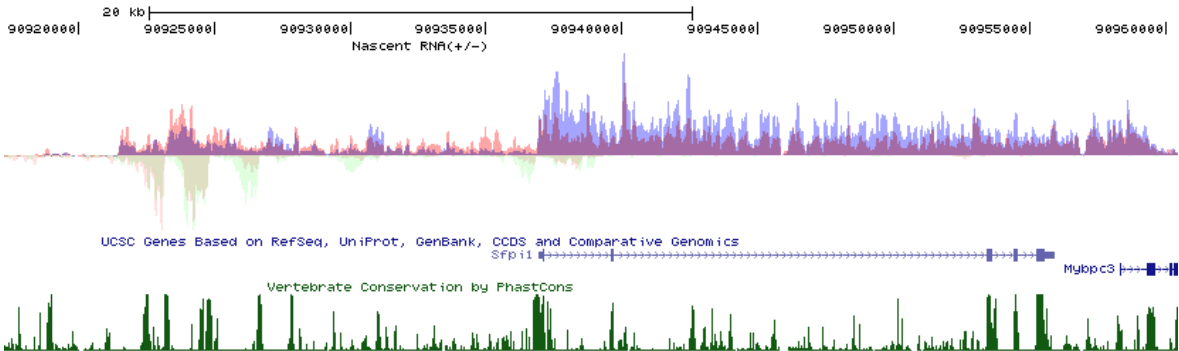
mES-Stat3

mES-Stat3

7 of 7 selected

Unfortunately, as of now editing hub information can only be done by directly modifying the hub files on the server. For example, to edit to colors, you must edit the `"/webserver/directory/hubName/genome/trackDB.txt"` file.

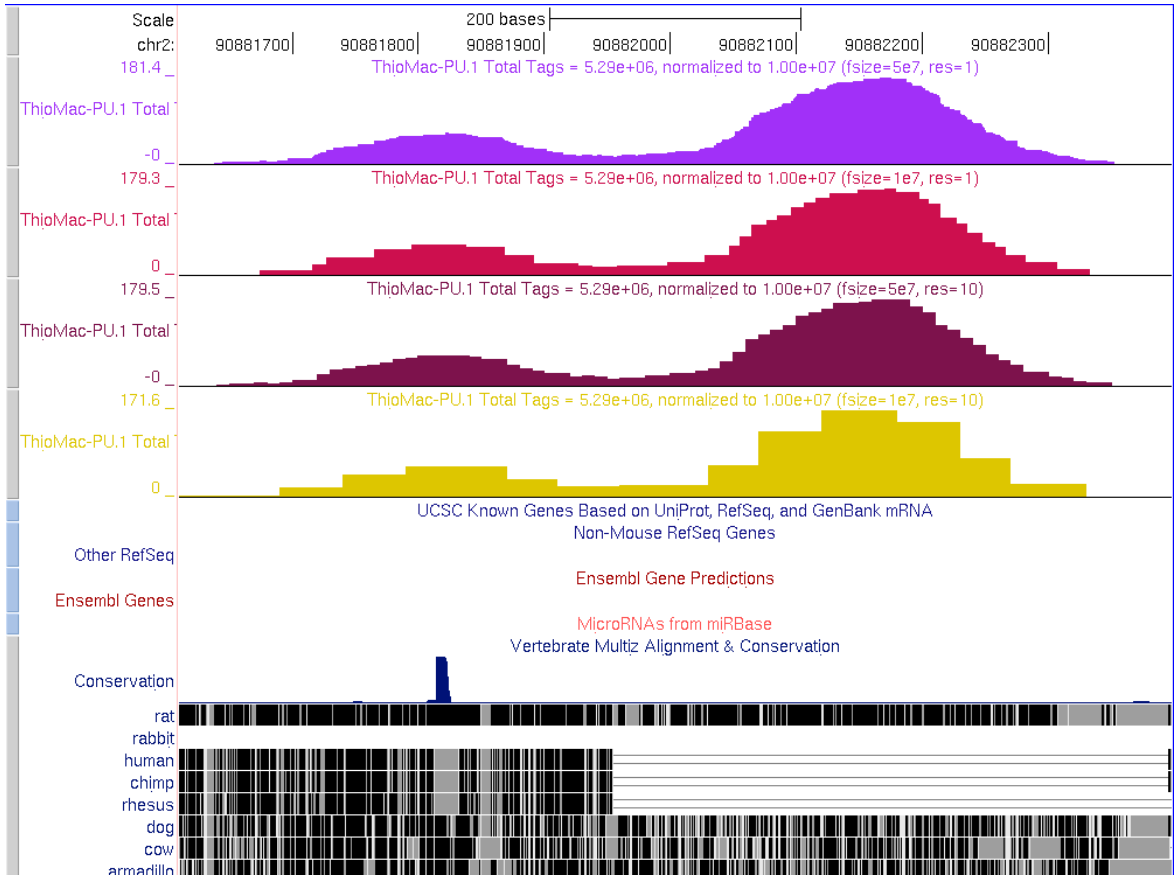
Because Hubs are so cool, HOMER will also do +/- strand RNA data right. Unfortunately, for now you can't mix stranded and non-stranded data in the same hub with the `makeMultiWigHub.pl` program. To visualize stranded information, add `"-strand"`. Below is an example:



Examples of UCSC bedGraph files

The following shows what the same data set looks like changing options for file size (`-fsize`) and resolution (`-res`). Usually it's best to use one or the other.

1. `-fsize 5e7 -res 1`
2. `-fsize 1e7 -res 1`
3. `-fsize 5e7 -res 10`
4. `-fsize 1e7 -res 10`



Command line options for makeUCSCfile

Usage: makeUCSCfile <tag directory> [options]

Creates a bedgraph file for visualization using the UCSC Genome Browser

General Options:

- fsize <#> (Size of file, when gzipped, default: 5e7)
- strand <both|separate|+|-> (control if reads are separated by strand, default: both)
- fragLength <# | auto | given> (Approximate fragment length, default: auto)
- adjust <#> (Adjust edge of tag 3' by # bp, negative for 5', default: none[good for dnase])
- tbp <#> (Maximum tags per bp to count, default: no limit)
- res <#> (Resolution, in bp, of file, default: 1)
- lastTag (To keep ucsc happy, last mapped tag is NOT extended by default
Using this option will allow extending of data past the last tag position)
- norm <#> (Total number of tags to normalize experiment to, default: 1e7)
- noadj (Do not normalize tag counts)
- neg (plot negative values, i.e. for - strand transcription)
- CpG (Show unmethylated CpG ratios)
- color <(0-255),(0-255),(0-255)> (no spaces, rgb color for UCSC track, default: random)
- bigWig <chrom.sizes> (creates a full resolution bigWig file and track line file
This requires bedGraphToBigWig to be available in your executable path
Also, because how bigWig files work, use "-strand -" and "-strand +"
in separate runs to make strand specific files: "-strand separate" will not work
- o <filename|auto> (send output to this file - will be gzipped, default: prints to stdout)
auto: this will place an appropriately named file in the tag directory
- name <...> (Name of UCSC track, default: auto generated)
- style <option> (See options below:)
 - chipseq (standard, default)
 - rnaseq (strand specific)
 - tss (strand specific, single bp fragment length)

- dnase (fragments centered on tag position instead of downstream)
- methyalted (single bp resolution of cytosine methylation)
- unmethyalted (single bp resolution of unmethyalted cytosines)
- circos <chrN:XXX-YYY|genome> (output only a specific region for circos[no header])

Command line options for makeBigWig.pl

Script for automating the process of creating bigWigs

Usage: makeBigWig.pl <tag directory> <genome> [special options] [options]

Special Options for bigWigs [choose one, don't combine]:

- normal (ChIP-Seq style, default)
- strand (Strand specific, for RNA-Seq and GRO-Seq)
- dnase (Special options for Crawford-lab style DNase-Seq)
- cage (Special options for CAGE/TSS-Seq)
- cpg (Special options for mCpG/CpG)

Other options:

Whatever options you want to pass to makeUCSCfile
!!Warning!!: do not try to specify "-strand separate" - use the special option above.

File options:

- fsize <#> (Use to limit the size of the bigwig files)
- url <URL> (URL directory -no filename- to tell UCSC where to look)
- webdir <directory> (name of directory to place resulting bigWig file)
- update (overwrite bigwigs in the webDir directory, otherwise random numbers are added to make the file unique.

Current url target (-url): <http://biowhat.ucsd.edu/bigWig/>

Current web directory (-webDir): /data/www/bigWig/

You're going to want to modify the \$wwwDir and \$httpDir variables at the top of the makeBigWig.pl program file to accomodate your system so you don't have to specify -url and -webdir all the time.

Command line options for makeMultiWigHub.pl

Script for automating the process of creating multiWig tracks

Usage: makeMultiWigHub.pl <hubname> <genome> [options] -d <tag directory1> [tag directory2]...

Special Options for bigWigs [choose one, don't combine]:

- normal (ChIP-Seq style, default)
- strand (Strand specific, for RNA-Seq and GRO-Seq)
- dnase (Special options for Crawford-lab style DNase-Seq)
- cage (Special options for CAGE/TSS-Seq)
- cpg (Special options for mCpG/CpG)

Other options:

Whatever options you want to pass to makeUCSCfile
!!Warning!!: do not try to specify "-strand separate" - use the special option above.
Also, for the genome, do NOT use repeat version (mm9r) - use mm9 instead

File options:

- force (overwrite existing hub)
- fsize <#> (limit the file size of the bigwig files to this value)

- url <URL> (URL directory -no filename- to tell UCSC where to look)
- webdir <directory> (name of directory to place resulting hub directory)

Current url target (-url): <http://biowhat.ucsd.edu/hubs/>
Current web directory (-webDir): </data/www/hubs/>

You're going to want to modify the \$wwwDir and \$httpDir variables at the top of the makeMultiWigHub.pl program file to accomidate your system so you don't have to specify -url and -webdir all the time.

Next: [Finding Peaks \(ChIP-enriched regions\) in the genome](#)



Can't figure something out? Questions, comments, concerns, or other feedback:
cbenner@ucsd.edu

Homer Update File

#

This file is updated from the Homer website and contains information about data available for
use with the program.

#

Each section has the same format, which is <tab> separated values specifying:

package name <tab> version <tab> description <tab> url <tab> directory <tab> additional parameters

SOFTWARE

homer v3.12 Code/Executables, ontologies, motifs for HOMER <http://biowhat.ucsd.edu/homer/homer.v3.12.zip> ./

PROMOTERS

human v3.0 Human RefSeq Promoters (hg18) <http://biowhat.ucsd.edu/homer/data/promoters/human.v3.0.zip>
data/promoters/ human,hg18,hg18r,refseq,-2000,2000

mouse v3.0 Mouse RefSeq Promoters (mm9) <http://biowhat.ucsd.edu/homer/data/promoters/mouse.v3.0.zip>
data/promoters/ mouse,mm9,mm9r,refseq,-2000,2000

rat v3.0 Rat RefSeq Promoters (rn4) <http://biowhat.ucsd.edu/homer/data/promoters/rat.v3.0.zip>
data/promoters/ rat,rn4,rn4r,refseq,-2000,2000

fly v3.0 Fly RefSeq Promoters (dm3) <http://biowhat.ucsd.edu/homer/data/promoters/fly.v3.0.zip>
data/promoters/ fly,dm3,dm3r,refseq,-2000,2000

worm v3.0 Worm RefSeq Promoters (ce6) <http://biowhat.ucsd.edu/homer/data/promoters/worm.v3.0.zip>
data/promoters/ worm,ce6,ce6r,refseq,-2000,2000

zebrafish v3.0 Zebrafish RefSeq Promoters (danRer7)
<http://biowhat.ucsd.edu/homer/data/promoters/zebrafish.v3.0.zip> data/promoters/ zebrafish,danRer7,danRer7r,refseq,-
2000,2000

yeast v3.1 Yeast RefSeq Promoters (sacCer2) <http://biowhat.ucsd.edu/homer/data/promoters/yeast.v3.1.zip>
data/promoters/ yeast,sacCer2,sacCer2,unigene,-2000,2000

arabidopsis v1.1 Arabidopsis RefSeq Promoters (tair10)

<http://biowhat.ucsd.edu/homer/data/promoters/arabidopsis.v1.1.zip> data/promoters/ arabidopsis,tair10,tair10,refseq,-
2000,2000

frog v1.0 Frog RefSeq Promoters (xenTro2) <http://biowhat.ucsd.edu/homer/data/promoters/frog.v1.0.zip>
data/promoters/ frog,xenTro2,xenTro2r,refseq,-2000,2000

human-mRNA v3.0 Human RefSeq mRNA (hg18) <http://biowhat.ucsd.edu/homer/data/promoters/human-mRNA.v3.0.zip>
data/promoters/ human,hg18,hg18r,refseq,-2000,2000

human-mRNA-3UTR v3.0 Human RefSeq mRNA 3'-UTRs (hg18)

<http://biowhat.ucsd.edu/homer/data/promoters/human-mRNA-3UTR.v3.0.zip> data/promoters/
human,hg18,hg18r,refseq,-2000,2000

human-mRNA-5UTR v3.0 Human RefSeq mRNA 5'-UTRs (hg18)

<http://biowhat.ucsd.edu/homer/data/promoters/human-mRNA-5UTR.v3.0.zip> data/promoters/
human,hg18,hg18r,refseq,-2000,2000

mouse-mRNA v3.0 Mouse RefSeq mRNA (mm9) <http://biowhat.ucsd.edu/homer/data/promoters/mouse-mRNA.v3.0.zip>
data/promoters/ mouse,mm9,mm9r,refseq,-2000,2000

mouse-mRNA-3UTR v3.0 Mouse RefSeq mRNA 3'-UTRs (mm9)

<http://biowhat.ucsd.edu/homer/data/promoters/mouse-mRNA-3UTR.v3.0.zip> data/promoters/
mouse,mm9,mm9r,refseq,-2000,2000

mouse-mRNA-5UTR v3.0 Mouse RefSeq mRNA 5'-UTRs (mm9)

<http://biowhat.ucsd.edu/homer/data/promoters/mouse-mRNA-5UTR.v3.0.zip> data/promoters/
mouse,mm9,mm9r,refseq,-2000,2000

rat-mRNA v3.0 Rat RefSeq mRNA (rn4) <http://biowhat.ucsd.edu/homer/data/promoters/rat-mRNA.v3.0.zip>
data/promoters/ rat,rn4,rn4r,refseq,-2000,2000

rat-mRNA-3UTR v3.0 Rat RefSeq mRNA 3'-UTRs (rn4) <http://biowhat.ucsd.edu/homer/data/promoters/rat-mRNA-3UTR.v3.0.zip>
data/promoters/ rat,rn4,rn4r,refseq,-2000,2000

rat-mRNA-5UTR v3.0 Rat RefSeq mRNA 5'-UTRs (rn4) <http://biowhat.ucsd.edu/homer/data/promoters/rat-mRNA-5UTR.v3.0.zip>

mRNA-5UTR.v3.0.zip data/promoters/ rat,rn4,rn4r,refseq,-2000,2000
 fly-mRNA v3.0 Fly RefSeq mRNA (dm3) <http://biowhat.ucsd.edu/homer/data/promoters/fly-mRNA.v3.0.zip>
 data/promoters/ fly,dm3,dm3r,refseq,-2000,2000
 fly-mRNA-3UTR v3.0 Fly RefSeq mRNA 3'-UTRs (dm3) <http://biowhat.ucsd.edu/homer/data/promoters/fly-mRNA-3UTR.v3.0.zip>
 data/promoters/ fly,dm3,dm3r,refseq,-2000,2000
 fly-mRNA-5UTR v3.0 Fly RefSeq mRNA 5'-UTRs (dm3) <http://biowhat.ucsd.edu/homer/data/promoters/fly-mRNA-5UTR.v3.0.zip>
 data/promoters/ fly,dm3,dm3r,refseq,-2000,2000
 worm-mRNA v3.0 Worm RefSeq mRNA (ce6) <http://biowhat.ucsd.edu/homer/data/promoters/worm-mRNA.v3.0.zip>
 data/promoters/ worm,ce6,ce6r,refseq,-2000,2000
 worm-mRNA-3UTR v3.0 Worm RefSeq mRNA 3'-UTRs (ce6)
<http://biowhat.ucsd.edu/homer/data/promoters/worm-mRNA-3UTR.v3.0.zip> data/promoters/ worm,ce6,ce6r,refseq,-2000,2000
 worm-mRNA-5UTR v3.0 Worm RefSeq mRNA 5'-UTRs (ce6)
<http://biowhat.ucsd.edu/homer/data/promoters/worm-mRNA-5UTR.v3.0.zip> data/promoters/ worm,ce6,ce6r,refseq,-2000,2000
 zebrafish-mRNA v3.0 Zebrafish RefSeq mRNA (danRer7) <http://biowhat.ucsd.edu/homer/data/promoters/zebrafish-mRNA.v3.0.zip>
 data/promoters/ zebrafish,danRer7,danRer7r,refseq,-2000,2000
 zebrafish-mRNA-3UTR v3.0 Zebrafish RefSeq mRNA 3'-UTRs (danRer7)
<http://biowhat.ucsd.edu/homer/data/promoters/zebrafish-mRNA-3UTR.v3.0.zip> data/promoters/
 zebrafish,danRer7,danRer7r,refseq,-2000,2000
 zebrafish-mRNA-5UTR v3.0 Zebrafish RefSeq mRNA 5'-UTRs (danRer7)
<http://biowhat.ucsd.edu/homer/data/promoters/zebrafish-mRNA-5UTR.v3.0.zip> data/promoters/
 zebrafish,danRer7,danRer7r,refseq,-2000,2000

GENOMES

hg19 v3.0 Human genome and conservation information for UCSC hg19
<http://biowhat.ucsd.edu/homer/data/genomes/hg19.v3.0.zip> data/genomes/hg19/ human,default
 hg19r v3.0 Human genome and conservation information for UCSC hg19 (masked)
<http://biowhat.ucsd.edu/homer/data/genomes/hg19r.v3.0.zip> data/genomes/hg19r/ human,default
 hg18 v3.0 Human genome and conservation information for UCSC hg18
<http://biowhat.ucsd.edu/homer/data/genomes/hg18.v3.0.zip> data/genomes/hg18/ human,default
 hg18r v3.0 Human genome and conservation information for UCSC hg18 (masked)
<http://biowhat.ucsd.edu/homer/data/genomes/hg18r.v3.0.zip> data/genomes/hg18r/ human,default
 hg17 v3.0 Human genome and conservation information for UCSC hg17
<http://biowhat.ucsd.edu/homer/data/genomes/hg17.v3.0.zip> data/genomes/hg17/ human,default
 hg17r v3.0 Human genome and conservation information for UCSC hg17 (masked)
<http://biowhat.ucsd.edu/homer/data/genomes/hg17r.v3.0.zip> data/genomes/hg17r/ human,default
 mm8 v3.0 Mouse genome and conservation information for UCSC mm8
<http://biowhat.ucsd.edu/homer/data/genomes/mm8.v3.0.zip> data/genomes/mm8/ mouse,default
 mm8r v3.0 Mouse genome and conservation information for UCSC mm8 (masked)
<http://biowhat.ucsd.edu/homer/data/genomes/mm8r.v3.0.zip> data/genomes/mm8r/ mouse,default
 mm9 v3.0 Mouse genome and conservation information for UCSC mm9
<http://biowhat.ucsd.edu/homer/data/genomes/mm9.v3.0.zip> data/genomes/mm9/ mouse,default
 mm9r v3.0 Mouse genome and conservation information for UCSC mm9 (masked)
<http://biowhat.ucsd.edu/homer/data/genomes/mm9r.v3.0.zip> data/genomes/mm9r/ mouse,default
 rn4 v3.0 Rat genome and conservation information for UCSC rn4
<http://biowhat.ucsd.edu/homer/data/genomes/rn4.v3.0.zip> data/genomes/rn4/ rat,default
 rn4r v3.0 Rat genome and conservation information for UCSC rn4 (masked)
<http://biowhat.ucsd.edu/homer/data/genomes/rn4r.v3.0.zip> data/genomes/rn4r/ rat,default
 dm3 v3.0 Fly genome and conservation information for UCSC dm3
<http://biowhat.ucsd.edu/homer/data/genomes/dm3.v3.0.zip> data/genomes/dm3/ fly,default
 dm3r v3.0 Fly genome and conservation information for UCSC dm3 (masked)
<http://biowhat.ucsd.edu/homer/data/genomes/dm3r.v3.0.zip> data/genomes/dm3r/ fly,default
 ce6 v3.0 Worm genome and conservation information for UCSC ce6

<http://biowhat.ucsd.edu/homer/data/genomes/ce6.v3.0.zip> data/genomes/ce6/ worm,default
ce6r v3.0 Worm genome and conservation information for UCSC ce6 (masked)
<http://biowhat.ucsd.edu/homer/data/genomes/ce6r.v3.0.zip> data/genomes/ce6r/ worm,default
danRer7 v3.0 Zebrafish genome and conservation information for UCSC danRer7
<http://biowhat.ucsd.edu/homer/data/genomes/danRer7.v3.0.zip> data/genomes/danRer7/ zebrafish,default
danRer7r v3.0 Zebrafish genome and conservation information for UCSC danRer7r (masked)
<http://biowhat.ucsd.edu/homer/data/genomes/danRer7r.v3.0.zip> data/genomes/danRer7r/ zebrafish,default
sacCer2 v3.1 Yeast genome and conservation information for UCSC sacCer2
<http://biowhat.ucsd.edu/homer/data/genomes/sacCer2.v3.1.zip> data/genomes/sacCer2/ yeast,default
tair10 v1.1 Arabidopsis genome tair10 <http://biowhat.ucsd.edu/homer/data/genomes/tair10.v1.1.zip>
data/genomes/tair10/ arabidopsis,default
xenTro2 v1.0 Frog genome and conservation information for UCSC xenTro2
<http://biowhat.ucsd.edu/homer/data/genomes/xenTro2.v1.0.zip> data/genomes/xenTro2/ frog,default
xenTro2r v1.0 Frog genome and conservation information for UCSC xenTro2 (masked)
<http://biowhat.ucsd.edu/homer/data/genomes/xenTro2r.v1.0.zip> data/genomes/xenTro2r/ frog,default



HOMER

Software for motif discovery and ChIP-Seq analysis

Gene Based Analysis (Microarray/RNA-Seq etc.)

There are 3 basic ways to run HOMER - with [FASTA files](#), with Gene Identifiers, or from [Genomic Positions](#). This section will outline gene-based analysis. Gene-based analysis is handled by the program **findMotifs.pl**. This program does more than just finding motifs, including [gene ontology analysis](#). By default, this is a promoter-based motif finding analysis, but can also be used to look for [RNA motifs in mRNAs](#).

Input Files:

findMotifs.pl will analyze the promoters of genes and look for motifs that are enriched in your target gene promoters relative to other promoters. The idea is to provide a list of genes that you believe should contain the same elements, such as genes that are co-regulated. For example, you may want to analyze the genes that are up-regulated by a stimulus, or genes that are specific to a certain cell type, or genes that appear in the same gene-expression cluster when doing clustering analysis. Alternatively, the gene IDs could come from a promoter ChIP-Chip experiment where each of the promoters are bound by the same transcription factor.

The primary input data is a list of gene identifiers, placed in single text file where each line contains a gene ID. This can be a gene expression spreadsheet, but findMotifs.pl will expect the first column to contain a gene ID. Below are examples of acceptable input file formats:

	A	B	C
1	Acc		
2	NM_010510		
3	NM_008361		
4	NM_008176		
5	NM_009140		
6	NM_013693		
7	AK152177		
8	L38281		
9	NM_009404		
10	NM_030612		
11	NM_010554		
12	NM_013652		
13	NM_011337		
14	NM_138648		
15	NM_021274		
16	NM_008331		
17	NM_010907		
18	NM_010276		
19	NM_007707		

	A	B	C	D
1	Acc	TGM 1h Notx	TGM 1h Kdo2	Log2 Fold
2	NM_010510	5.675	13.865	8.19
3	NM_008361	7.865	16.015	8.15
4	NM_008176	8.47	15.975	7.505
5	NM_009140	6.785	14.055	7.27
6	NM_013693	9.775	16.77	6.995
7	AK152177	9.09	16.06	6.97
8	L38281	9.1	15.94	6.84
9	NM_009404	6.085	12.765	6.68
10	NM_030612	7.95	14.485	6.535
11	NM_010554	7.715	13.98	6.265
12	NM_013652	11.255	17.215	5.96
13	NM_011337	10.305	15.65	5.345
14	NM_138648	7.26	12.26	5
15	NM_021274	8.99	13.93	4.94
16	NM_008331	7.775	12.415	4.64
17	NM_010907	12.6	17.17	4.57
18	NM_010276	6.29	10.725	4.435
19	NM_007707	11.895	16.23	4.335

If you're having trouble with the program, 9 out of 10 times it is due to an incorrectly formatted input file. If using EXCEL (especially on the Mac), make sure to save input files as "Text (Windows)". HOMER will choke on binary XLS files. HOMER accepts a broad range of different types of gene identifiers:

- NCBI Entrez Gene IDs
- NCBI Unigene IDs
- NCBI Refseq IDs (mRNA, protein)
- Ensembl Gene IDs
- Gene Symbols (i.e. Official Gene names, like "Nfkb1")
- popular affymetrix probe IDs (MOE430, U133plus, U95, U75A)

If your favorite ID isn't listed above, then you will have to covert to one of these before using HOMER. Your input file can have a mix of different IDs too. HOMER will let you know how many of the IDs it was able to "understand", so you can give it a try.

Running findMotifs.pl

findMotifs.pl takes 3 mandatory arguments: A gene ID input file, the name of the promoter set (which is tied to an organism), and an output directory for all of the output files. For now, HOMER only supports 7 organisms, although you can contact me (cbenner@ucsd.edu) if you think it would be good to add more. It is also possible to [add support for an organism and/or a custom promoter set yourself](#). For each of these organisms, a default "Promoter Set" was constructed based on RefSeq Genes:

- human (*Homo sapiens*)
- mouse (*Mus musculus*)
- rat (*Rattus norvegicus*)
- fly (*Drosophila melanogaster*)
- worm (*Caenorhabditis elegans*)
- zebrafish (*Danio rerio*)
- yeast (*Saccharomyces cerevisiae*)

To run **findMotifs.pl**, type the following:

```
findMotifs.pl <inputfile.txt> <promoter set> <output directory> [options]
```

i.e. `findMotifs.pl lpsInducedGenes.pl mouse LPSMotifResults/ -start -400 -end 100 -len 8,10`

This will search for motifs of length 8 and 10 from -400 to +100 relative to the TSS.

`findMotifs.pl` will produce a number of output files in the "output directory". The primary output will be in HTML files that should be opened with you favorite web browser.

What does findMotifs.pl do?

This program performs a number of operations en route to providing a basic analysis of motif and functional enrichment. The various steps are outlined below:

- 1. Convert Gene IDs to consistent gene identifier (usually Entrez Gene ID)
- 2. Select appropriate background IDs (usually all confident genes, i.e. not olfactory genes), or take user supplied list (see below)
- 3. Perform Gene Ontology enrichment calculation (for details, see [here](#)).
- 4. Assign weights to background promoters based on the distribution of CpG content in the target gene promoters such that comparable numbers of low and high-CpG promoters are analyzed.
- 5. Perform [de novo motif analysis](#)
- 6. Create output HTML pages for *de novo* analysis.
- 7. Perform known motif enrichment analysis and corresponding output pages.

Output files




The first output page created is for *de novo* results (beside the GO analysis). This page contains a sorted list of non-redundant motifs ranked by their enrichment p-values. Below is an example generate using genes up-regulated after 1 hour of treatment with LPS in murine macrophages (sample gene list):

`findMotifs.pl upLPS.mouse.txt mouse outputDirectory/ -len 8,10,12`

This should produce:

Homer de novo Motif Results

[Known Motif Enrichment Results](#)
[Gene Ontology Enrichment Results](#)
If Homer is having trouble matching a motif to a known motif, try copy/pasting the matrix file into [STAMP](#)
* - possible false positive

Rank	Motif	P-value	log P-value	Best Match/Details	Motif File
1		1.407e-35	-8.025e+01	NFkB-p65/GM12787-p65-ChIP-Seq/Homer More Information Similar Motifs Found	motif file (matrix)
2		1.371e-14	-3.192e+01	ISRE/ThioMac-LPS-exp/HOMER More Information Similar Motifs Found	motif file (matrix)
3		3.312e-14	-3.104e+01	CRE/Promoter/Homer More Information Similar Motifs Found	motif file (matrix)

Information is provided for each motif. One thing I've been debating is whether

or not to include my best guess for the motif identity ("Best Match/Details"). The problem is that people take this result too literally and it can be a huge problem and a source of misunderstanding. In many cases the best match doesn't not look very convincing, but few people seem to pay attention to that (see below).

[More information about HOMER motif finding output](#)

Important motif finding parameters

Repeat Masked vs. Unmasked Sequences

Actually, this usually doesn't matter *that* much. Since HOMER is a differential motif discovery algorithm, common repeats are usually in both the target and background sequences. However, it is not uncommon that a transcription factor binds to a certain class of repeats, which may cause several large stretches of similar sequence to be processed, biasing the results. Usually it's safer to go with the masked version. To use the unmasked version, use "**-nomask**".

Promoter Region ("**-start <#>**" and "**-end <#>**", default: -300, 50)

Different parts of the promoter can be used for motif finding. In the "old days", everyone would search 1kb upstream and look for motifs there. As it turns out, most of the action is within 200 bp of the promoter, with the motif density dropping of considerably after that. The maximum sizes handled by HOMER are -2000 and 2000.

Motif length ("**-len <#>**" or "**-len <#>,<#>,...**", default 8,10,12)

Specifies the length of motifs to be found. HOMER will find motifs of each size separately and then combine the results at the end. The length of time it takes to find motifs increases greatly with increasing size. In general, it's best to try out enrichment with shorter lengths (i.e. less than 15) before trying longer lengths. Much longer motifs can be found with HOMER, but it's best to use smaller sets of sequence when trying to find long motifs (i.e. use "**-len 20 -start -150 -end 50**"), otherwise it may take way too long (or take too much memory). The other trick to reduce the total resource consumption is to reduce the number of background sequences.

Mismatches allowed in global optimization phase ("**-mis <#>**", default: 2)

HOMER looks for promising candidates by initially checking ordinary oligos for enrichment, allowing mismatches. The more mismatches you allow, the more sensitive the algorithm, particularly for longer motifs. However, this also slows down the algorithm a bit. If searching for motifs longer than 12-15 bp, it's best to increase this value to at least 3 or even 4.

Number of CPUs to use ("**-p <#>**", default 1)

HOMER is now multicore compliant. It's not perfectly parallelized, however, certain types of analysis can benefit. In general, the longer the length of the motif, the better the speed-up you'll see.

Number of motifs to find ("**-S <#>**", default 25)

Specifies the number of motifs of each length to find. 25 is already quite a bit.

If anything, I'd recommend reducing this number, particularly for long motifs to reduce the total execution time.

Normalize CpG% content instead of GC% content ("**-cpg**")

Consider trying if HOMER is stuck finding "CGCGCGCG"-like motifs. You can also play around with disabling GC/CpG normalization ("**-noweight**").

Region level autonormalization ("**-nlen <#>**", default 3, "**-nlen 0**" to disable)

Motif level autonormalization ("**-olen <#>**", default 0 i.e. disabled)

Autonormalization attempts to remove sequence bias from lower order oligos (1-mers, 2-mers ... up to <#>). Region level autonormalization, which is for 1/2/3 mers by default, attempts to normalize background regions by adjusting their weights. If this isn't getting the job done (autonormalization is not guaranteed to remove all sequence bias), you can try the more aggressive motif level autonormalization ("**-olen <#>**"). This performs the autonormalization routine on the oligo table during de novo motif discovery. (see [here](#) for more info)

User defined background genes ("**-bg <file of Gene IDs to use as background>**")

By default HOMER uses all other promoters as the background set. You can choose a specific set of background promoters by placing the gene identifiers in a file (just like the target genes) and using the "**-bg <file>**" option. These will still be normalized for CpG% or GC% content just like normal and autonormalized unless these options are turned off (i.e. "**-nlen 0 -noweight**"). This can be very useful since HOMER is a differential motif discovery algorithm.

Binomial enrichment scoring ("**-b**")

By default, **findMotifs.pl** uses the hypergeometric distribution to score motifs. If the set of sequences you are analyzing is very large, you may want to use the binomial to speed things up. In general, it is recommended to use the hypergeometric since it does a better job of describing biological enrichment.

Find enrichment of individual oligos ("**-oligo**").

This creates output files in the output directory named *oligo.length.txt*.

Only search for motifs on + strand ("**-norevopp**")

By default, HOMER looks for transcription factor-like motifs on both strands. This will force it to only look at the + strand (relative to the TSS, so - strand if the TSS is on the - strand).

Mask motifs ("**-mask <motif file>**")

Mask the motif(s) in the supplied motif file before starting motif finding. Multiple motifs can be in the motif file.

Optimize motifs ("**-opt <motif file>**")

Instead of looking for novel de novo motifs, HOMER will instead try to optimize the motif supplied. This is cool when trying to change the length of a motif, or find a very long version of a given motif. For example, if you specify "**-opt <file>**" and "**-len 50**", it will try to expand the motif to 50bp and optimize it.

Dump FASTA files ("**-dumpFasta**")

Like the fact that HOMER organizes and extracts your sequence files, but don't

care for HOMER as a motif finding algorithm? That's cool, just specify "-dumpFasta" and the files "target.fa" and "background.fa" will show up in your output directory. You can then use them with MEME or whatever. Just remember, Chuck knows where you live...

Removing redundant promoters ("-noredun")

By default, HOMER only keeps one promoter if it is shared by two genes (i.e. bidirectional promoter) so that the sequence isn't duplicated. If the duplicated promoter is found in both the target promoter group and the background group, the background instance is removed.

Finding Instances of Specific Motifs

By default, HOMER does not return the locations of each motif found in the motif discovery process. To recover the motif locations, you must first select the motifs you're interested in by getting the "motif file" output by HOMER. You can combine multiple motifs in single file if you like to form a "motif library". To identify motif locations, you have two options:

1. Run **findMotifs.pl** with the "**-find <motif file>**" option. This will output a tab-delimited text file with each line containing an instance of the motif in the target peaks. The output is sent to *stdout*.

For example: **findMotifs.pl lpsGenes.txt mouse MotifOutputDirectory/ -find motif1.motif > outputfile.txt**

The output file will contain the columns:

1. Peak/Region ID
2. Offset from the TSS
3. Sequence of the site
4. Name of the Motif
5. Strand
6. Motif Score (log odds score of the motif matrix, higher scores are better matches)

2. Run **annotatePeaks.pl** with the "**-m <motif file>**" option in **tss mode** (see the [here](#) for more info). To use this option, you must [install the proper genome](#). Chuck prefers doing it this way. This will output a tab-delimited text file with each line containing a peak/region and a column containing instance of each motif separated by commas to stdout

For example: **annotatePeaks.pl tss mm9 -size -300,50 -m motif1.motif > outputfile.txt**

The output file will contain columns:

1. Peak/Region ID
2. Chromosome
3. Start of TSS region
4. End of TSS region

5. Strand of Peaks

- 6-18: annotation information
- 19. CpG%
- 20. GC%
- 21. Motif Instances
- ...

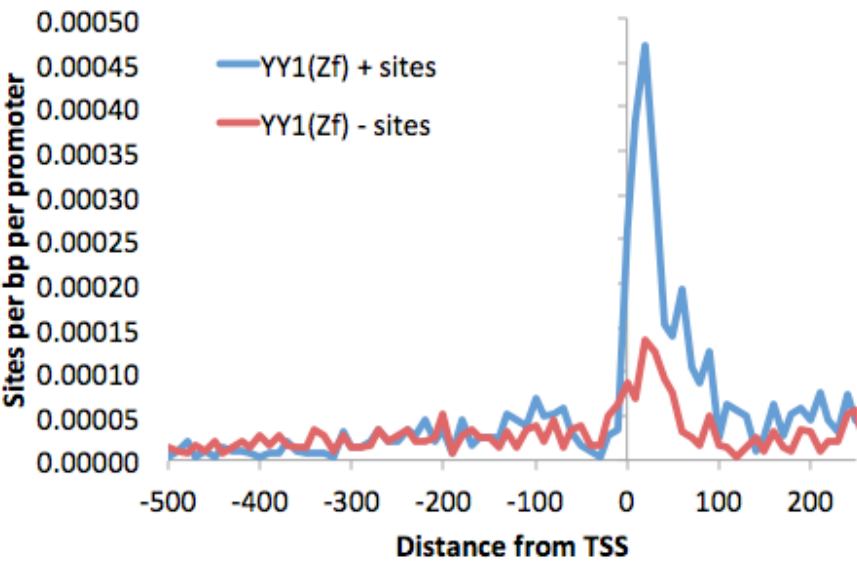
Motif Instances have the following format:
<distance from TSS>(<sequence>,<strand>,<conservation>)
i.e -29(TAAATCAACA,+,0.00)

To limit the search to only the target set of genes (or any subset of genes), use the option "**-list <gene id file>**".

This can also be used to find histograms of motif density relative to the TSS - just add the "**-hist <#>**" option.

For example: **annotatePeaks.pl tss mm9 -size -500,250 -hist 10 -m yy1.motif > outputfile.txt**

Graphing the output file with EXCEL, we can see the distribution of TSS-associated motif YY1:



Add "**-list <gene id list>**" to make a histogram on a specific subset of genes.

Practical Tips for Motif Finding

Can't figure something out? Questions, comments, concerns, or other feedback:
cbenner@ucsd.edu





HOMER

Software for motif discovery and next-sequencing analysis

Practical Tips to Motif Finding with HOMER

Below are some general tips for getting the most out of you motif analysis when using HOMER. Be sure to look over [this section about judging motif quality!](#)

What to do if motif finding takes too long...

Ctrl+C... If you are using reasonable parameters (see next section), it shouldn't take more than an hour or so, and in most cases much less.

Choosing the length of motifs to find

It's almost always a good idea to start with the default parameters. Resist the urge to find motifs larger than 12 bp the first time around. Longer motifs will show up as different short motifs when finding shorter motifs. If there aren't any truly significant motifs when looking at short motifs, it is unlikely that you will find good long motifs either. And it doesn't take much time to check for short motifs.

i.e. **-S 25 -len 8,10,12**

Once you do find motifs that look promising, try looking for longer motifs.

Finding Long Motifs

The new version of HOMER (v3.0+) is better at looking for long motifs. However, it can be tricky looking for long motifs because the search space gets very large. Also, the running time on longer motifs increases and may break your patience.

Since HOMER is an empirical motif finding program, it starts from actual oligos present in the sequence and attempts to figure out if they are enriched. If you are looking at 20 bp sequences, there is a good chance that they are all more-or-less unique in your data set with only 1 instance in either the target or background sequences. HOMER normally allows mismatches in the original oligo to see if the oligo together with similar oligos are collectively enriched. The problem is that this technique starts to break down at long lengths. It takes many mismatches to find enough related sequences to assess enrichment, and it is computationally expensive to find them.

To maintain sensitivity for longer motifs:

Increase the "-mis <#>" option to allow more mismatches. In practice, I would use at least "-mis 4" or "-mis 5" for sensitive detection of 20 bp motifs. If the data set is for a strong motif (i.e. CTCF ChIP-Seq peaks), then you don't have to worry about this so much since the motif signal is very strong.

To find longer version of a given motif:

The local optimization phase handles long motifs pretty well - long motifs cause more of a problem with the global search phase. Usually long motifs show enrichment for parts at shorter motif lengths. Another strategy is to first find a short version of the motif (i.e. -len 12), and then rerun HOMER and tell it to optimize the motif at a longer motif length with the "-opt <motif file>". To do this with a motif named "motif1.motif":

findMotifsGenome.pl peaks.txt hg18r OutputDirectory -opt motif1.motif -len 30

This will enlarge the motif(s) in the motif1.motif to 30 bp and optimize them.

Other things to try:

- try to reduce the number of target sequences to include only high quality sequences (such as "focused" peaks or peak with the highest peak scores).
- try limiting the length of sequences used (i.e. "-size 50" when using **findMotifsGenome.pl**)
- try limiting the total number of background sequences (i.e. "-N 20000" when using **findMotifsGenome.pl**)

In a practical sense, you should be able to search for motifs of length 20 or 30 when analyzing ~10k peaks with parameters "-len 20,30 -size 50 -N 25000 -mis 5". HOMER wasn't really designed to find really long motifs; since it is an empirical motif finder, the sequence "space" gets a bit sparse at lengths >16, but in practice it still works.

How many sequences can HOMER handle?

In theory, a lot (i.e. millions). It has been designed to work well with ~10k target sequences and 50k background sequences. If you are using a large number of sequences with **findMotifs.pl**, you may want to use the "-b" option, which switches to the [cumulative binomial distribution](#) for motif scoring, which is faster to calculate and gives essentially the same results when using large numbers of sequences. The binomial is used by default in **findMotifsGenome.pl**. (I guess it should be called BOMER !?).

Choosing background sequences

Most of the methods in HOMER attempt to select the proper background for you, but in some cases this doesn't work. Normally, HOMER attempts to normalize the GC content in target and background sequences. If you believe normalizing the CpG content is better, use the option "-cpg" when performing motif finding with either **findMotifs.pl** or **findMotifsGenome.pl**.

In some cases the user may have a better idea of what the background should be, so HOMER offers the following options:

Promoters: When using analyzing promoters with **findMotifs.pl**, if you wish to use a specific set of promoters as background, place them in a text file (1st column is the ID) and use the "-bg <background IDs file>" option. Genes found in the target and background will be removed from the background set so that they don't cancel out each other. Examples:

- Use expressed genes from a microarray as background
- Use only genes represented on the microarray as background

Genomic Regions: When analyzing peaks/regions with **findMotifsGenome.pl**, you can specify the genomic regions of appropriate background regions by placing them in their own peak file and using the "-bg <background peak file>". Examples:

- Specify peaks common to two cell types as background when trying to find motifs specific to a set of cell-type specific peaks - this will help cancel out the primary motif and reveal the co-enriched motifs
- If peaks are near Exons, specify regions on Exons as background to remove triplet bias.

FASTA Files: Here you have (the necessary) freedom to specify whatever you want!

Please note, that if the number of background sequences is small, or similar in number to the number of target sequences, you should consider switching to the hypergeometric distribution to improve accuracy when using **findMotifsGenome.pl** ("-h").

You may also want to disable CpG/GC normalization depending on how you selected your background, which can be done with "-noweight".


Sequence Bias, GC/CpG normalization, and Autonormalization

By default, homer performs several normalization steps to make sure the sequences that are being analyzed look reasonable (details [here](#)). Since GC% differences are the largest source of bias, these are dealt with during the background selection stage to minimize any issues.

Other types of sequence bias may be present in your data. The purpose of the autonormalization routines ("-nlen <#>" and "-olen <#>") are there to help deal with this type of bias. If your results have strong enrichment for simple nucleotide repeats, you may want to try "-olen <#>" which will more aggressively normalize the data.

How to Judge the Quality of the Motifs Found

WARNING: Because this is the hardest thing for people to understand, I'll say it again here. HOMER will print the best guess for the motif next to the motif results, but before you tell your adviser that your factor is enriched for that motif, it is highly recommended that you look at the alignment!!! Here is an example of what might be going on:

8		3.941e-24	-5.389e+01	YY1_01 More Information Similar Motifs Found
---	---	-----------	------------	---

In this case, HOMER has identified YY1 as the "best guess" match for this *de novo* motif. Well, lets click on "More Information" and see what's up:

YY1(M00069)


Match Rank: 2

Score: 6.11

Offset: 0

Orientation: reverse strand

Alignment: AGCAGGCA-----
ANCAGNCAAGATGGCCGNGN



As you can see in this case, the motif aligns to the edge of the known YY1 motif, and not to the core of the YY1 motif (CAAGATGGC). This doesn't mean that the YY1 motif is not enriched in your data, but unless there are other motif results that show enrichment of the other parts of the YY1 motif, it is not likely that the YY1 motif is enriched in your data set.

And as always, remember that HOMER is a *de novo* motif tool!!! Even though HOMER will guess the best match, if it is a novel motif, your don't want to trust that match anyway. Hence, the you can see the importance of viewing the alignment and getting a feel for what evidence exists either for or against this assignment.

There are many cases where HOMER will find motifs with very low p-values, but the motifs might look "suspicious". Poor quality motifs can be loosely classified into the following groups:

Low Complexity Motifs:

(less of a problem with the v3.0+) These types of motifs tend to show preference for same collection of 1, 2, 3, or 4 nucleotides in each position and are typically very degenerate. For example:



These motifs typically arise when a systematic bias exists between target and background sequence sets. Commonly they will be very high in GC-content, in which case you may want to try adding "-gc" to your motif finding command to normalize by total GC-content instead of CpG-content.

Other times this will come up when analyzing sequences for various genomic features that have not been controlled for in the background - for example, comparing sequences from promoters to random genomic background sequences in some organisms will show preferences for purines or pyrimidines. HOMER is very sensitive, so if there is a bias in the composition of the sequences, HOMER will likely pick it up. Autonormalization in the new version minimizes this problem.

Simple Repeat Motifs:

(less of a problem with the v3.0+) Some times motifs will show repeats of certain patterns:



Usually motifs like this will be accompanied by several other motifs looking highly similar. Unless there is a good reason to believe these may be real, it's best to assume there is likely a problem with the background. These can arise if your target sequences are highly enriched on exons (think triplets) and other types of sequences, and if "-gc" doesn't help, you may have to think hard about the types of sequences that you are trying to analyze and try to match them. (i.e. Promoters vs. Exons etc.) You can also try upping the ante by using "-olen <#>" to autonormalize sequence bias at the oligo level.

Small Quantity Motifs / Repeats:

These are a little harder to explain. These look like real motifs but are found in an incredibly low percentage of targets - i.e. like an oligo or part of a repeat that is in a couple of the target sequences that appears as a significant motif. Statistically speaking they are enriched, but likely not real. These are the biggest problem when looking for motifs in promoters from a small list of regulated genes. In principle, in a motif is present in **less than 5% of the targets sequences**, there may be a problem.

Leftover Junk:

These are motifs that appear in your lower in your results list after you've discovered high quality motifs. If an element is highly enriched in your sequences, HOMER will (hopefully) find it, mask it, and then continue to look for motifs. In this case, many of the other motifs that HOMER finds will be offsets or degenerate versions of highly enriched motif(s) found at the beginning. For example (another PU.1 example):

The top motif identified:

Rank	Motif	P-value	log P-pvalue	Best Match/Details
1		0.000e+00	-2.938e+04	PU.1/ThioMac-PU.1-ChIP-Seq/Homer More Information

Examples further down the list:

8		0.000e+00	-3.711e+03	PU.1/ThioMac-PU.1-ChIP-Seq/Homer More Information
22		0.000e+00	-1.692e+03	STAT1/Stat More Information
74		6.727e-190	-4.356e+02	MAF(M00648) More Information

This are not necessarily negative results, but they should be place in context. This commonly happens in ChIP-Seq data sets where the immunoprecipitated protein is highly expressed and binds strongly a ton of binding sites. These "other" motifs are likely also capable of binding PU.1 and probably represent low affinity binding sites, but giving them too much individual attention is not recommended in this context given they are motifs that have been constructed using leftover oligos in the motif finding process that didn't make it into the most highly enrichment motifs. A safer way to approach these elements is to repeat the motif finding procedure with regions lacking the top motif, or by adding "-mask <motif file>" to the motif finding command to cleanly mask the top motif from the motif finding procedure.



Can't figure something out? Questions, comments, concerns, or other feedback:
cbenner@ucsd.edu



HOMER

Software for motif discovery and ChIP-Seq analysis

Motif Finding with HOMER from FASTA files

Most of HOMER's functionality is built around either promoter or genomic position based analysis, and aims to manage the sequence manipulation, hiding it from the user. However, if you have some sequences that you would like HOMER to analyze, the program **findMotifs.pl** accepts **FASTA** formatted files for analysis. Alternatively you could use the **homer2** executable which also accepts FASTA files as input.

HOMER is designed to analyze high-throughput data using differential motif discovery, which means you **MUST** have both target and background sequences, and in each case you should have several (preferably thousands) of sequences in each set that are roughly the same length.

A quick note about FASTA files - Each sequence should have a unique identifier. In theory, HOMER should be flexible with what is in the header line, but if you're having trouble please just keep it simple with minimal quite-space, especially tabs. For example:

```
>NM_003456
AAGGCCTGAGATAGCTAGAGCTGAGAGTTTTCCACACG
```

Running findMotifs.pl with FASTA files:

To find motifs from FASTA files, run findMotifs.pl with the target sequence FASTA file as the first command-line argument, and use the option "**-fasta <file>**" to specify the background FASTA file. You CANNOT NOT specify a background file - that would defeat the purpose of differential motif finding.

```
findMotifs.pl <targetSequences.fa> fasta <output directory> -fasta  
<background.fa> [options]
```

NOTE: you must choose an "organism" for the 2nd argument to keep with the structure of the command, even though this isn't actually relevant for FASTA based analysis. Organism doesn't have to match the data in the FASTA files. You can use a valid organism or just put "**fasta**" as a place holder. i.e.:

```
findMotifs.pl chuckNorrisGenes.fa human analysis_output/ -fasta  
normalHumanGenes.fa
```

Many other options are available to control motif finding parameters.

findMotifs.pl will perform GC normalization and autonormalization by default (see [here for more details](#)).

Finding instances of motifs with FASTA files:

To find instance of a motif, run the same command used for motif discovery above but add the option "**-find <motif file>**". Motif results will be sent to stdout, so to capture the results in a file Add "**> outputfile**" to the end of the command.

```
findMotifs.pl <targetSequences.fa> fasta <output directory> -fasta  
<background.fa> [options] -find motif1.motif > outputfile.txt
```

For more information on the output file format, see [here](#).

Using homer2 directly with FASTA files:

homer2 is the motif finding executable, and it can choke down FASTA files if you want to avoid all the nonsense above. Running the **homer2** command will also give you access to other options for optimizing the motif finding process. **homer2** works by first specifying a command, and then the appropriate options:

```
homer2 <command> [options]  
i.e. homer2 denovo -i input.fa -b background.fa > outputfile.txt
```

To find instances of the output motifs, use "**homer2 find**". To see other commands, just type "**homer2**".



Can't figure something out? Questions, comments, concerns, or other feedback:
cbenner@ucsd.edu



HOMER

Software for motif discovery and ChIP-Seq analysis

Functional Enrichment Analysis

HOMER contains a program for performing functional enrichment analysis from a list of Entrez Gene IDs (**findGO.pl**). Normally, you don't need to know how to use **findGO.pl** because it is called internally by **findMotifs.pl** and **annotatePeaks.pl**. **findGO.pl** assesses the enrichment of various categories of gene function, biological pathways, domain structure, chromosome location, etc., in your gene list relative to a set of background gene IDs. Enrichment is calculated assuming the cumulative [hypergeometric distribution](#), much in the same way that HOMER scores motif enrichment. HOMER does not attempt to deal with the multiple-hypothesis testing problem, although it does record the number tests made in each output file.

There are several different "ontologies", or libraries of gene groupings, that HOMER will check for enrichment. Below is the list:

- Biological Process: Functional groupings of proteins (Gene Ontology)
- Molecular Function: Mechanistic actions of proteins (Gene Ontology)
- Cellular Component: Protein localization (Gene Ontology)
- KEGG Pathways: Groups of proteins in the same pathways (From KEGG)
- Interactions: Groups of proteins interacting with the same protein (From NCBI Gene)
- Interpro: Proteins with similar domains and features (Interpro)
- Pfam: Proteins with similar domains and features (Pfam)
- SMART: Proteins with similar domains and features (SMART)
- Gene3D: Proteins with similar domains and features (Gene3D Database)
- Prosite: Proteins with similar domains and features (Prosite Database)
- PRINTS: Proteins with similar domains and features (PRINTS Database)
- Chromosome Location: Genes with similar chromosome localization
- miRNA Targets: Genes targeted by similar miRNAs (miRBase) [[Has not been updated in a while](#)].

Some of these ontologies are pretty standard (i.e. Gene Ontology), while others were simply easy to provide since I had to parse through uniprot flat files anyway.

To run **findGO.pl** on its own, type:

findGO.pl <input file of Entrez Gene IDs> <organism> <output directory> [background ID file(optional)]

Normally **findGO.pl** will use a default set of gene ids for that organism. The program produces one HTML file, containing a mixture of different enriched categories, as well tab-delimited text files for each of the ontologies analyzed. An example of the GO output is show below:

P-value	LogP	Term	GO Tree	GO ID	# of Genes in Term	# of Target Genes in Term	# of Total Genes	# of Target Genes	Common Genes
2.912e-26	-5.880e+01	immune response	biological process	GO:0006955	349	35	18091	168	Il10,Cd14,Malt1,Ccl2,Ccl7,Ifih1
3.912e-26	-5.850e+01	immune system process	biological process	GO:0002376	679	45	18091	168	S100a9,Egr1,Il10,Cd14,Malt1,C
1.823e-25	-5.696e+01	cytokine activity	molecular function	GO:0005125	178	27	18371	167	Gdf15,Il10,Csf2,Ccl9,Ccl2,Ccl7
3.372e-23	-5.174e+01	defense response	biological process	GO:0006952	430	35	18091	168	Il10,Cd14,Malt1,Nupr1,Ccl2,Cc

In the HTML page, findGO.pl will convert the gene IDs into gene symbols so that it is easy to read. In the text files the IDs are kept as gene IDs.

Gene Ontology and GO slims

My favorite topic in the world of Gene Ontology analysis is the use of [GO slims](#). HOMER does not contain GO slims libraries. As a result, you may find that many of your gene ontology results contain terms such as "*metabolism*" and "*cellular process*" when other tools may not reveal these terms. GO slims are great because they delete terms that you don't generally want to see. Another way to do this is to look through your list and just use the terms you want. There really isn't much of a difference between that and using GO slims - but at least you're being honest with yourself with one of the techniques.



Can't figure something out? Questions, comments, concerns, or other feedback:
cbenner@ucsd.edu



HOMER

Software for motif discovery and ChIP-Seq analysis

Next-gen Sequencing Analysis: Creating a "Tag Directory" with makeTagDirectory

To facilitate the analysis of ChIP-Seq (or any other type of short read re-sequencing data), it is useful to first transform the sequence alignment into platform independent data structure representing the experiment, analogous to loading the data into a database. HOMER does this by placing all relevant information about the experiment into a "Tag Directory", which is essentially a directory on your computer that contains several files describing your experiment.

During the creation of tag directories, several quality control routines are run to help provide information and feedback about the quality of the experiment. During this phase several important parameters are estimated that are later used for downstream analysis, such as the estimated length of ChIP-Seq fragments.

Input Alignment Files

During this part of the analysis, any type of sequencing data can be use (ChIP-Seq/RNA-Seq/DNase-Seq, etc.). If these files are zipped (*.gz, *.zip, *.bz2), HOMER will automatically unzip, process, and re-zip them (if applicable) so you don't have to worry about this step.

To create a "Tag Directory", you must have alignment files in one of the following formats:

- BED format
- SAM format
- BAM format (HOMER will use "samtools view file.BAM > file.SAM" to covert to a SAM file, so "samtools" must be available)
- default bowtie output format
- *.eland_result.txt or *_export.txt format from the Illumina pipeline

If your alignment is in a different format, it is recommended that you convert it into a BED file format:

Column1: chromosome
 Column2: start position
 Column3: end position
 Column4: Name (or strand +/-)

****Column5: Number of reads at this position****

Column6: Strand +/-

****Unfortunately, BED files are used in different ways by different groups. By default, HOMER assumes the 5th column of a BED file is the number of reads at that position. If this is NOT the case, add the option "-forceBED" in your command to tell HOMER to ignore this value.**

Alternatively (or in combination), you can make tag directories from existing tag directories or from tag files (explained below).

If your input files are named "s_1_sequence.txt", or have a suffix such as *.fq or *.fastq, then you probably have raw sequence files without raw sequence alignment information. You need to align these sequences to the genome first. [Learn about aligning data to the genome here.](#)

Paired-End Reads

Unfortunately, HOMER does NOT explicitly support paired-end reads... yet. This is primarily due to the fact that our group hasn't used paired-end sequencing for any of these applications (nor is there much data in the literature using paired-end reads for applications like ChIP-Seq, with the exception of RNA-Seq), but this is changing and support for them will be available in the near future.

For now, the best option is to separate the paired-end reads and treat them as separate single-end runs (or just use one of the two reads).

Creating Tag Directories

To make a tag directory, run the following command:

```
makeTagDirectory <Output Directory Name> [options] <alignment file1>
[alignment file 2] ...
```

Where the first argument must be the output directory (required). If it does not exist, it will be created. If it does exist, it will be overwritten.

An example:

```
makeTagDirectory Macrophage-PU.1-ChIP-Seq/ pu1.lane1.bed
pu1.lane2.bed pu1.lane3.bed
```

Several additional options exist for **makeTagDirectory**. The program attempts to guess the format of your alignment files, but if it is unsuccessful, you can force the format with "**-format <X>**".

```
makeTagDirectory Macrophage-PU.1-ChIP-Seq/ pu1.lane1.bed
pu1.lane2.bed pu1.lane3.bed -format bed
makeTagDirectory Macrophage-H3K4me1-ChIP-Seq/
s_1_sequence.align.sam -format sam
```

Sometimes BED file alignments contain stupid values in the 5th column, such as

quality information etc. HOMER will treat this value as the number of reads aligning to the same location. If this is not how the value is used, add "-forceBED" to ignore the value found in the 5th column of a BED file.

**makeTagDirectory Macrophage-PU.1-ChIP-Seq/ pu1.lane1.bed
pu1.lane2.bed pu1.lane3.bed -format bed -forceBED**

To combine tag directories, for example when combining two separate experiments into one, do the following:

makeTagDirectory Combined-PU.1-ChIP-Seq/ -d Exp1-ChIP-Seq/ Exp2-ChIP-Seq/ Exp3-ChIP-Seq/

What does makeTagDirectory do?

makeTagDirectory basically parses through the alignment file and splits the tags into separate files based on their chromosome. As a result, several *.tags.tsv files are created in the output directory. These are made to very efficiently return to the data during downstream analysis. This also helps speed up the analysis of very large data sets without running out of memory.

In the end, your output directory will contain several *.tags.tsv files, as well as a file named "**tagInfo.txt**". This file contains information about your sequencing run, including the total number of tags considered. This file is used by later programs to quickly reference information about the experiment, and can be manually modified to set certain parameters for analysis.

makeTagDirectory also performs several quality control steps shown below.

Basic Quality Control Analysis:

The following 4 basic quality control results are produced by default and are relatively inexpensive in terms of processing power to create while parsing alignment files into Tag Directories, and do not require any extra information to produce. These files are meant to be opened with a text editor or graphed using EXCEL or similar program. More detailed descriptions of these files and how to interpret them are found in the sequencing technique-specific tutorials.

Basic Tag information

tagInfo.txt - Contains basic configuration information, such as the total number of reads, the total number of unique positions with aligned reads (by genome and chromosome), and various other statistics. One of the more important parameters is "fragmentLengthEstimate=##", which provides an estimate of the length of fragments used for sequencing.

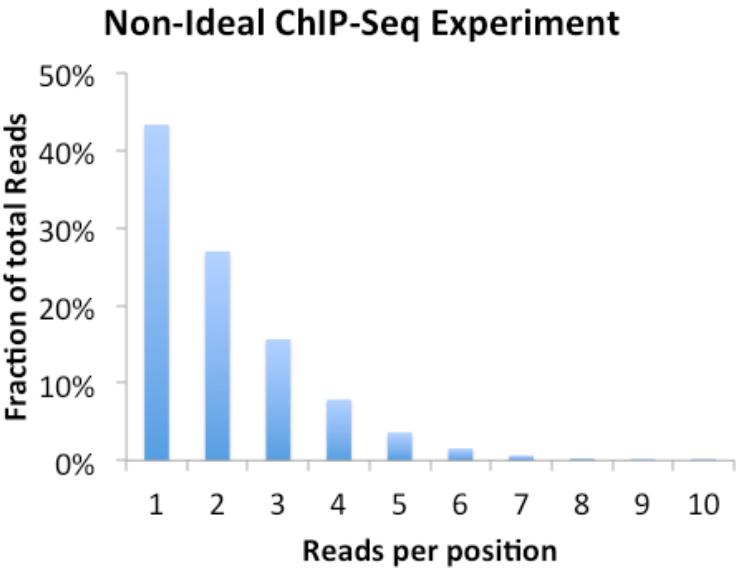
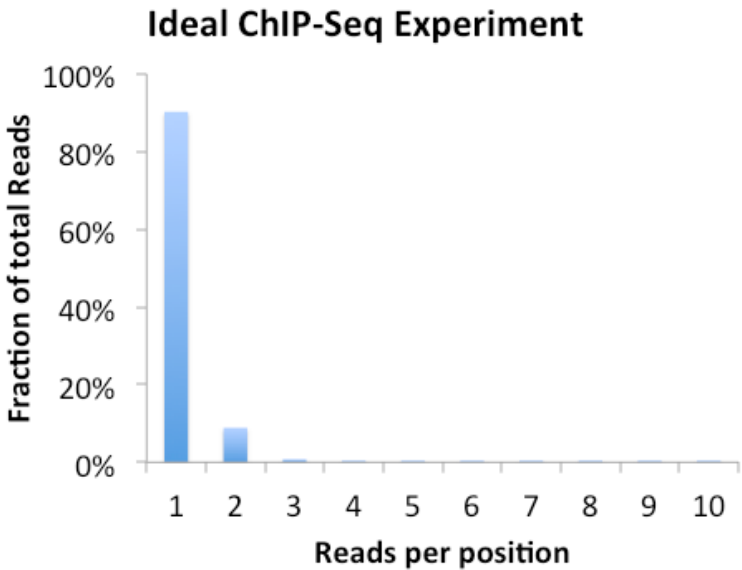
Read Length Distribution

tagLengthDistribution.txt - File contains a histogram of read lengths used

for alignment.

Clonal Tag Distribution

tagCountDistribution.txt - File contains a histogram of clonal read depth, showing the number of reads per unique position. If an experiment is "over-sequenced", you start seeing the same reads over and over instead of unique reads. Sometimes this is a sign there was not enough starting material for sequencing library preparation. Below are examples of ideal and non-ideal results - in the case of the non-ideal experiment, you probably don't want to sequence that library anymore.



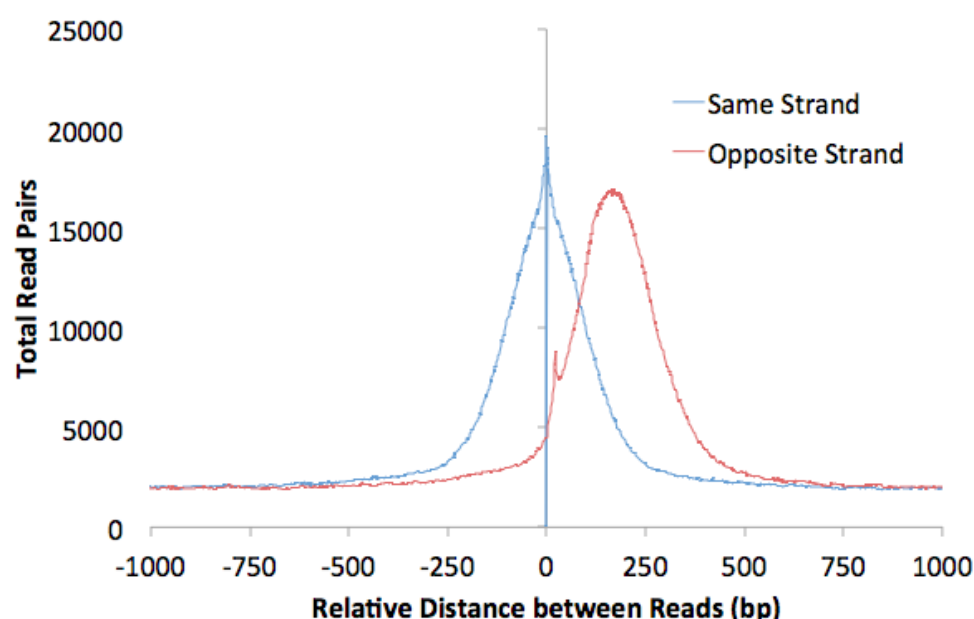
If the experiment is highly clonal and not expected to be, it might help to clean up the downstream analysis by forcing tag counts at each position to be no greater than x , where x is usually 1. To do this rerun the `makeTagDirectory` command and add `"-tbp <#>"` where $\#$ is the maximum tags per bp.

Autocorrelation Analysis

tagAutocorrelation.txt - The autocorrelation routine creates a distribution of

distances between adjacent reads in the genome. If reads are mapped to the same strand, they are added to the first column. If adjacent reads map to different strands, they are added to the 2nd column. The results from autocorrelation analysis are very useful for troubleshooting problems with the experiment, and are used to estimate the fragment length for ChIP-Seq and MNase-Seq. The fragment length is estimated by finding the position where the "opposite strand" distribution is maximum. HOMER will use this value as the fragment length unless overridden with the option **"-fragLength <#>"**.

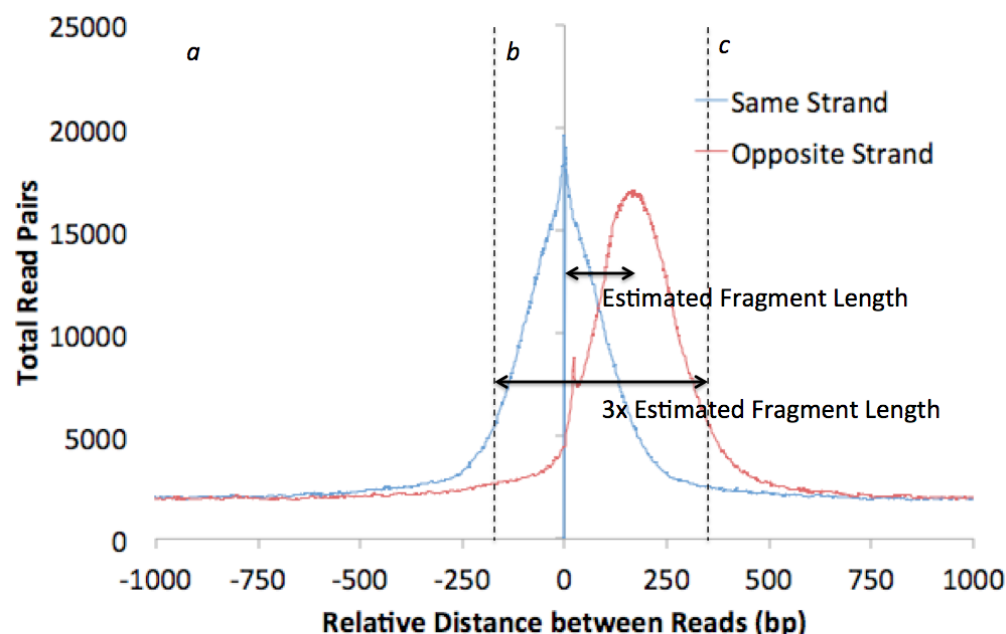
Different types of experiments (i.e. ChIP-Seq vs. DNase-Seq) produce different looking autocorrelation plots, and more detailed discussion of these differences can be found in the individual tutorials. Below is an example from a successful ChIP-Seq experiment:



HOMER also uses the autocorrelation results to guess what type of experiment you conducted. It computes 3 statistics:

- Same strand fold enrichment: Enrichment of reads on the same strand within 3x the estimated fragment length
- Diff strand fold enrichment: Enrichment of reads on different strands within 3x the estimated fragment length
- Same / Diff fold enrichment: Difference between enrichment of reads on the same strand or different strands

Below is a schematic to visualize how these are calculated (keep in mind that the background is calculated out to +/- 2kb):



Same strand and Diff Strand Fold Enrichment: $\frac{\text{Density } b}{\text{Density } a \text{ \& } c}$

Same/Diff Fold Enrichment: $\frac{\text{Density } b \text{ (same strand)}}{\text{Density } b \text{ (Opposite Strand)}}$

Depending on the value of the autocorrelation quality statistics, HOMER will guess what your experiment is:

- If the Same/Diff Fold Enrichment is > 8 fold, it's a great chance the sample is strand-specific RNA. If HOMER decides the sample is probably RNA due to the difference between same strand and different strand numbers, it will automatically set the estimated fragment length to 75 bp. This is because it is difficult to estimate the fragment length for RNA/GRO-seq. To manually set the fragment length, use "**-fragLength <#>**"
- If both the "Same Strand Fold Enrichment" and "Diff Strand Fold Enrichment" are both greater than 1.5 fold, there is good chance you're looking at a working ChIP-Seq experiment.

This is meant as immediate feedback, and not a definitive declaration of the quality or type of your data. We've found it useful to help identify wrong annotations in sample IDs, for example, and helped give us an idea about how well an experiment worked. Good ChIP-Seq antibodies give "Same Strand Fold Enrichment" values well above 5.0 for transcription factors. However, in may cases good results can be extracted out of experiments with lower enrichments, even those that yield a "Guessing sample is ChIP-Seq - may have low enrichment with lots of background" message.

Sequence Bias Analysis:

Invaluable information about a sequencing experiment can be found by examining the relationship between sequencing reads and the genomic sequence the came from. Sequence bias analysis is not performed by default. To perform this analysis, you must provide the *genome* and the "-checkGC" option.

**makeTagDirectory <Output Directory Name> [options] -genome
<genome> -checkGC <alignment file1> ...**
i.e. **makeTagDirectory Macrophage-PU.1-ChIP-Seq -genome mm9 -
checkGC pu1.alignment.bed**

This analysis will produce several output files in addition to the basic quality control analysis described above.

An important prerequisite for analyzing sequence bias is that the [appropriate genome must be configured for use with HOMER](#). In version v3.1, HOMER now handles custom/arbitrary genomes. Instead of installing/configuring a genome, you can specify the path to a file or directory containing the genomic sequence in FASTA format. The genome can be in a single FASTA file, or you specify a directory where each chromosome is in a separate file (named chrXXX.fa or chrXXX.fa.masked). In either case, the FASTA headers must contain the chromosome names followed by white space, i.e. ">chr blahblahblah", not ">chr1-blahblahblah", or preferably only ">chr1".

NOTE: If using a sequencing type other than ChIP-Seq or MNase-Seq, you may want to add "-fragLength <#>" since it may be difficult for HOMER to automatically determine the size of fragments used for sequencing.

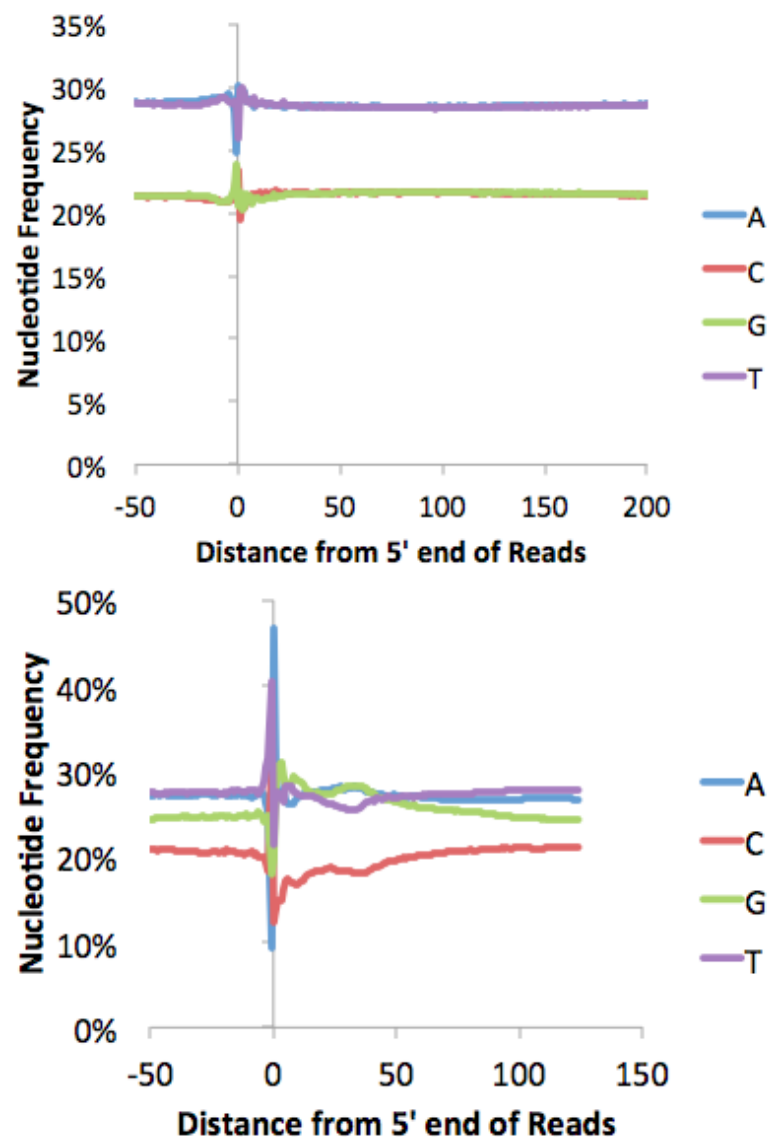
Genomic Nucleotide Frequency relative to read positions

tagFreq.txt - Calculates the nucleotide and dinucleotide frequencies as a function of distance from the 5' end of all reads.

tagFreqUniq.txt - Same as tagFreq.txt, however individual genomic positions are only counted once. If 10 reads mapped to the same position, those nucleotide counts would be added 10 times for "tagFreq.txt", but only once for "tagFreqUniq.txt".

By default, HOMER calculates this frequency from -50 bp to +50 bp relative to the end of the estimated fragment (e.g. not +50bp relative to the 36 bp read, but +50 bp relative to the 200 bp ChIP-fragment). This can be changed using the options "**-freqStart <#>**" and "**-freqEnd <#>**"

Below are some examples of this file graphed with EXCEL. One is a pretty typical ChIP-Seq file, the other Chuck wouldn't say where it came from, but he's pretty sure there might be a problem with how they performed their experiment... Quite a bit can be learned from looking at these types of plots (see individual tutorials of additional details)

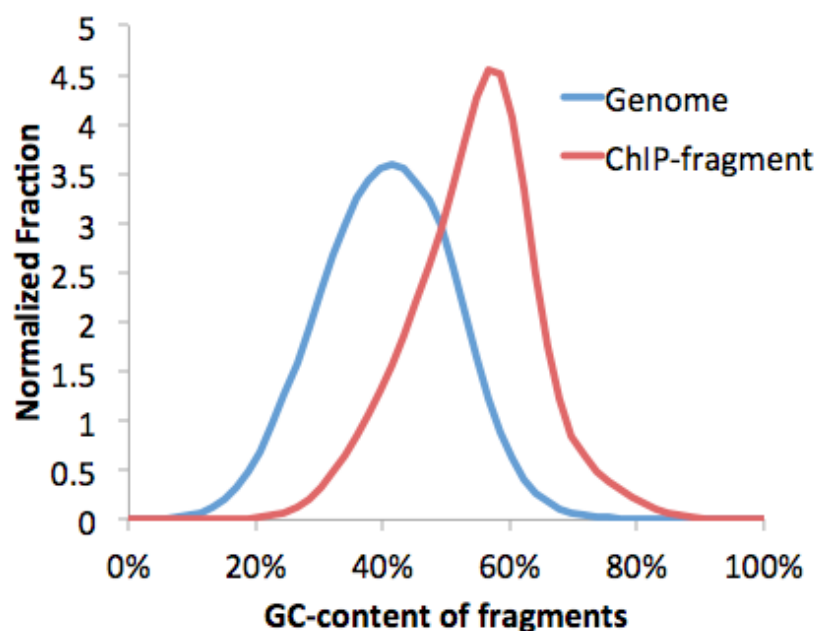
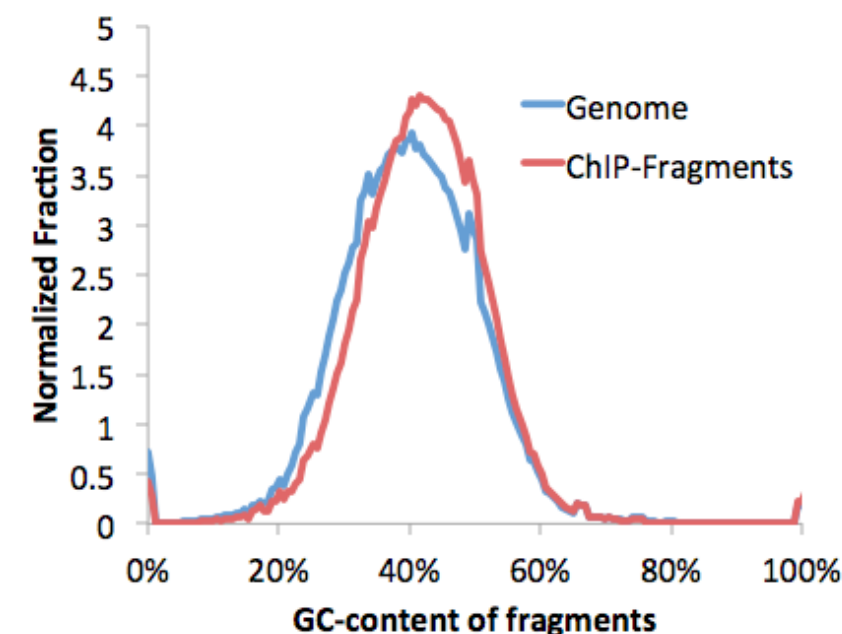


Fragment GC% Distribution

tagGCcontent.txt - Distribution of fragment GC%-content. GC% is calculated from the 5' end position of the read to end of the "fragment", not the end of the read. For ChIP-Seq and MNase-Seq, the fragment length is automatically estimated with pretty high confidence. If you are using another type of sequencing or want to set this manually, be sure to specify the desired fragment length manually ("**-fragLength** <#>").

genomeGCcontent.txt - Distribution of fragment GC%-content at each location in the genome (i.e. expected distribution)

These are very important files to check for any sequencing experiment. Due to the numerous steps of library preparation involving amplification, size selection and gel extraction, it is very easy for the average GC%-content of the sequencing library to "shift". This can be a disastrous problem. Consider the following:



The problem with a GC% shifted sample is that even if the sample is random sequence, you will start to show "enrichment" at places with high GC-content in the genome, such as at CpG Islands. This is unfortunate because most GC-rich areas are at transcription start sites, which might make you think the experiment worked, when in reality the sample was boiled instead of placed in the freezer. See below to learn about GC-normalization.

Sequence Bias GC normalization:

HOMER has built in routines for normalizing GC-bias in a sequencing experiment. In general, normalizing for GC-content is tricky. First, you need to decide if it is worth normalizing at all.

When should I normalize for GC-content in my sequencing experiment?

First you must be able to identify that there is a GC-bias problem. If the average GC% is within 5% of the expected genomic average, you're probably ok. If you are comparing several experiments of similar type, and they all have the same approximate GC-bias, you're also probably better off just comparing the experiments as they are. However, if one of your replicates is much more GC-rich than another, similar sample, you may want to consider normalizing it.

For some experiments, it's tough to know what the expected GC-content should be. For example, if sequencing H3K4me3 ChIP-Seq data, which is typically found near CpG Islands, you may expect the GC% to be higher. However, even with H3K4me3 ChIP-Seq, most ChIP experiments are not very efficient, and most of the DNA being sequenced is background. As a result, the average GC content should not be too far off the expected genomic average. You can always "try" normalizing.

If the GC-bias is severe, you might be better off repeating the experiment. Normalization essentially destroys information, so repeating the experiment (or at least the sequencing library preparation) may be a good move. When re-prepping the sample, try amplifying less and when extracting DNA from gels, dissolve the gels at room temperature.

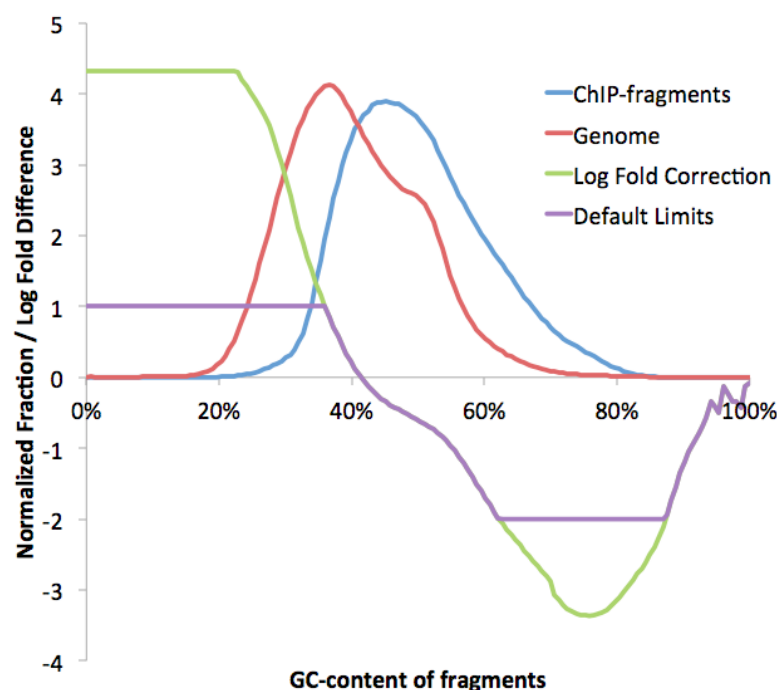
Normalizing tags for GC-bias

To normalize for GC-bias, run the same **makeTagDirectory** command, but you must specify the genome (i.e. "**-genome hg18**") and a target normalization file ("**-normGC <filename>**"). To normalize to the expected genomic average, use "**-normGC default**".

makeTagDirectory <Tag Directory> -genome <genome> -normGC <GC profile file profile> <alignment file 1> [alignment file 2] ...

i.e. makeTagDirectory Macrophage-PU.1-GC -genome mm9 -normGC default aligned.reads.bed

The normalization procedure is actually very simple. HOMER calculates the distribution of fragment GC%-contents, then for each range of GC%, normalizes the tag counts to the expected distribution. If your sample is more GC-rich than expected, reads in GC-rich regions will be reduced to a fractional value (limited by "**-minNormRatio <#>**"). Likewise, if reads are in AT-rich regions, their values will be increased (limited by "**-maxNormRatio <#>**"). HOMER is happy to deal with fractional tag values - there is no reason why a sequencing read can't be counted as a "half-a-read", for example. The biggest problem with **increasing** the value of a read is that it is akin to creating information that doesn't exist, so HOMER puts a tight cap on that adjustment to only 2-fold (change with "**-maxNormRatio <#>**"). The resulting normalization information is recorded in the **tagGCnormalization.txt** file.



Often, the expected genome GC-content is not appropriate. To normalize to a custom GC-profile, provide a file after **-normGC** option ("**-normGC <filename>**"). This file should be formatted just like the **tagGCcontent.txt** files found in the directories. In fact, the idea is that you can normalize one experiment to another by providing the experiments **tagGCcontent.txt** file as the argument for **-normGC**. One potential strategy is to combine several experimental files across different experiments into a single tag directory, and then use the tagGCcontent.txt file from this directory to normalize each of the other experiment, normalizing each experiment to the average from all of the experiments.

Command line options of makeTagDirectory command:

Usage: parseAlignment <directory> <alignment file 1> [file 2] ... [options]

Creates a platform-independent 'tag directory' for later analysis.

Currently BED, eland, bowtie, and sam files are accepted. The program will try to automatically detect the alignment format if not specified.

Existing tag directories can be added or combined to make a new one using -d/-t

If more than one format is needed and the program cannot auto-detect it properly, make separate tag directories by running the program separately, then combine them.

Options:

-fragLength <# | given> (Set estimated fragment length - given: use read lengths)

By default treats the sample as a single read ChIP-Seq experiment

-format <X> where X can be: (with column specifications underneath)

bed - BED format files:
(1:chr,2:start,3:end,4:+/- or read name,5:# tags,6:+/-)
bowtie - output from bowtie (run with --best -k 2 options)
(1:read name,2:+/-,3:chr,4:position,5:seq,6:quality,
7:NA,8:mismatch info)
eland_result - output from basic eland
(1:read name,2:seq,3:code,4:#zeroMM,5:#oneMM,6:#twoMM,7:chr,
8:position,9:F/R,10:-mismatches
eland_export - output from illumina pipeline (22 columns total)
(1-5:read name
info,9:sequence,10:quality,11:chr,13:position,14:strand)
eland_extended - output from illumina pipeline (4 columns total)
(1:read name,2:sequence,3:match stats,4:positions[,])
sam - SAM formatted files (use samTools to covert BAMs into SAM if you
have BAM)
-keep (keep one mapping of each read regardless if multiple equal mappings
exist)
-forceBED (if 5th column of BED file contains stupid values, like mapping quality
instead of number of tags, then ignore this column)
-d <tag directory> [tag directory 2] ... (add Tag directory to new tag directory)
-t <tag file> [tag file 2] ... (add tag file i.e. *.tags.tsv to new tag directory)
-single (Create a single tags.tsv file for all "chromosomes" - i.e. if >100
chromosomes)
-tbp <#> (Maximum tags per bp, default: no maximum)

GC-bias options:
-genome <genome version> (To see available genomes, use "-genome list")
-checkGC (check Sequence bias, requires "-genome")
-freqStart <#> (offset to start calculating frequency, default: -50)
-freqEnd <#> (distance past fragment length to calculate frequency, default:
+50)
-normGC <target GC profile file> (i.e. tagGCcontent.txt file from control
experiment)
Use "-normGC default" to match the genomic GC distribution
-minNormRatio <#> (Minimum deflation ratio of tag counts, default: 0.25)
-maxNormRatio <#> (Maximum inflation ratio of tag counts, default: 2.0)

Next: [Creating UCSC Genome Browser visualization files](#)



Can't figure something out? Questions, comments, concerns, or other feedback:
cbenner@ucsd.edu



HOMER

Software for motif discovery and next-gen sequencing analysis

Finding Enriched Motifs in Genomic Regions (*findMotifsGenome.pl*)

HOMER was initially developed to automate the process of finding enriched motifs in ChIP-Seq peaks. More generally, HOMER analyzes genomic positions, not limited to only ChIP-Seq peaks, for enriched motifs. The main idea is that all the user really needs is a file containing genomic coordinates (i.e. a HOMER peak file or BED file), and HOMER will generally take care of the rest. To analyze a peak file for motifs, run the following command:

findMotifsGenome.pl <peak/BED file> <genome> <output directory> [options]

i.e. **findMotifsGenome.pl ERpeaks.txt hg18 ER_MotifOutput/ -mask**

A variety of output files will be placed in the <output directory>, including html pages showing the results. The "-mask" is optional and tells the program to use the repeat-masked sequence. (The old shorthand hg18r will also work)

The findMotifsGenome.pl program is a wrapper that helps set up the data for analysis using the HOMER motif discovery algorithm. By default this will perform *de novo* motif discovery as well as check the enrichment of known motifs. If you have not done so already, please look over [this page](#) describing how HOMER analyzes sequences for enriched motifs.

An important prerequisite for analyzing genomic motifs is that the appropriate genome [must be configured for use with HOMER](#). In version v3.1, HOMER now handles custom/arbitrary genomes. Instead of installing/configuring a genome, you can specify the path to a file or directory containing the genomic sequence in FASTA format. The genome can be in a single FASTA file, or you specify a directory where where each chromosome can be in a separate file (named chrXXX.fa or chrXXX.fa.masked). In either case, the FASTA headers must contain the chromosome names followed by white space, i.e. ">chr blahblahblah", not ">chr1-blahblahblah", or preferably only ">chr1". (also note that homer will create a "prepared/" directory where the genome is, so make sure you have write permissions in the genomic directory.

Acceptable Input files

findMotifsGenome.pl accepts [HOMER peak files](#) or [BED files](#):

HOMER peak files should have at minimum 5 columns (separated by TABs, additional columns will be ignored):

- Column1: Unique Peak ID
- Column2: chromosome
- Column3: starting position
- Column4: ending position
- Column5: Strand (+/- or 0/1, where 0="+", 1="-")

BED files should have at minimum 6 columns (separated by TABs, additional columns will be ignored)

- Column1: chromosome
- Column2: starting position
- Column3: ending position
- Column4: Unique Peak ID
- Column5: not used
- Column6: Strand (+/- or 0/1, where 0="+", 1="-")

In theory, HOMER will accept BED files with only 4 columns (+/- in the 4th column), and files without unique IDs, but this is NOT recommended. For one, if you don't have unique IDs for your regions, it's hard to go back and figure out which region contains which peak.

Mac Users: If using a EXCEL to prepare input files, make sure to save files as a "Text (Windows)" if running MacOS - saving as "Tab delimited text" in Mac produces problems for the software. Otherwise, you can run the script "**changeNewLine.pl <filename>**" to convert the Mac-formatted text file to a Windows/Dos/Unix formatted text file.

If errors occur, it is likely that the file is not in the correct format, or the first column is not actually populated with unique identifiers.

Custom Background Regions

Since HOMER uses a differential motif discovery algorithm, different types of background sequences can be chosen to produce different results. For example, you may want to compare the ChIP-Seq peaks specific in one cell type versus the peaks that are specific to another. To do this, create a second peak/BED file and use it with the argument "**-bg <peak/BED file>**". HOMER will still try to normalize the background to remove GC-bias and will also perform autonormalization (see below). You can turn off the normalization with ("**-noweight**" and/or "**-nlen 0**").

How findMotifsGenome.pl works

There are a series of steps that the program goes through to find quality motifs:

1. Verify peak/BED file

HOMER makes sure you have valid peaks, and checks to make sure you have unique peak identifiers. If there are replicates, it will inform you, and will add numbers to peak names to ensure they are unique for downstream analysis.

2. Extract sequences from the genome corresponding to the regions in the input file, filtering sequences that are >70% "N"

This step is pretty self explanatory. If you wish to extract sequences from a genome for any reason, check out [homerTools](#). HOMER will also trash sequences that are predominately "N". If you feel you are throwing away too many sequences, try running **findMotifsGenome.pl** on an unmasked genome.

3. Calculate GC/CpG content of peak sequences.

CpG Islands are the single biggest source of sequence content bias in mammalian genomes, and are unfortunately found near transcription start sites, where all the

action is! By default, HOMER tracks GC% (use **"-cpg"** to use CpG%).

4. Prepare the genomic sequences of the selected size to serve as background sequences.

This step is only done the first time you find motifs from regions of a given size (**"-size <#>"**). HOMER takes regions near the TSS of genes (+/- 50kb) and splits them into regions of the indicated size. It then calculates their GC/CpG% and stores them for later use to speed up execution the next time you search for motifs from similar sized regions.

5. Randomly select background regions for motif discovery.

Since HOMER is a differential motif discovery algorithm, it must use background sequence regions as a control. By default, HOMER selects enough random background regions such that the total number of regions is 50000 or 2x the total number of peaks, whichever is larger (to change use **"-N <#>"**). The more total sequence that is used, the slower the program will run, but you want to make sure there is enough background regions to reliably estimate motif frequency. HOMER attempts to select background regions that match the GC-content distribution of the input sequences (in 5% increments). For example, if your input regions are extremely GC-rich, HOMER will select random regions from GC-rich regions of the genome as a control.

If custom background regions are provided (**"-bg <peak/BED file>"**), HOMER will automatically ensure that these regions do NOT overlap with the target regions (using **mergePeaks**). Custom regions will still be normalized for GC-content.

6. Autonormalization of sequence bias.

Autonormalization is a unique procedure provided by HOMER that attempts to remove bias introduced by lower-order oligo sequences. It works by assuming your targets regions and background regions should not have an imbalance in 1-mers, 2-mers, 3-mers, etc. The maximum length of oligo that is autonormalized is specified by **"-nlen <#>"** (default is 3, to disable use **"-nlen 0"**). For example, there should not be significantly more A's in the target sequences than in the background. After calculating the imbalances for each oligo, it adjusts the weights of each background sequence by a small amount to help normalize any imbalance. If target sequences are rich in A, then background sequences that contain many A's will be assigned higher weights while those with very few A's will be assigned lower weights. The weights are incremented by only small amounts and the procedure repeated many times in a hill climbing optimization. This procedure helps remove some of the sequence bias associated with certain genomic regions, or bias that may have been introduced by biased experimental results such as biased sequencing.

7. Check enrichment of known motifs

HOMER screens its library of reliable motifs against the target and background sequences for enrichment, returning motifs enriched with a p-value less than 0.05. The known motif enrichment is performed first since it is usually faster, and gives a faster look at what's enriched in your target regions. Known motif enrichment will be reported to the "knownResults.html" file in the output directory.

8. *de novo* motif finding

Best saved for last. By default, HOMER will search for motifs of len 8, 10, and 12 bp (change using **-len <#,#,>** with no spaces between the numbers, i.e. **"-len**

6,10,15,20"). For a more detail description of the motif discovery algorithm, see [here](#). Output from the de novo motif finding will be displayed in the "homerResults.html" file.

findMotifsGenome.pl Output

A full description of motif finding output and the output can be found [here](#).

Several files are produced in the output directory:

- homerMotifs.motifs<#> : these are the output files from the de novo motif finding, separated by motif length, and represent separate runs of the algorithm.
- homerMotifs.all.motifs : Simply the concatenated file composed of all the homerMotifs.motifs<#> files.
- motifFindingParameters.txt : command used to execute findMotifsGenome.pl
- knownResults.txt : text file containing statistics about known motif enrichment (open in EXCEL).
- seq.autonorm.tsv : autonormalization statistics for lower-order oligo normalization.
- homerResults.html : formatted output of *de novo* motif finding.

Homer de novo Motif Results

[Known Motif Enrichment Results](#)
[Gene Ontology Enrichment Results](#)
If Homer is having trouble matching a motif to a known motif, try copy/pasting the matrix file into [STAMP](#)
More information on motif finding results: [HOMER](#) | [Description of Results](#) | [Tips](#)
Total target sequences = 37301
Total background sequences = 35962
* - possible false positive

Rank	Motif	P-value	log P-value	% of Targets	% of Background	STD(Bg STD)	Best Match/Details	Motif File
1	TGTTTACATA	1e-12661	-2.915e+04	70.91%	15.19%	40.5bp (65.1bp)	Foxa2(Forkhead)/Liver-Foxa2-ChIP-Seq/Homer More Information Similar Motifs Found	motif file (matrix)
2	CTTGGCAG	1e-578	-1.332e+03	27.14%	16.52%	54.0bp (65.5bp)	NF1-halfsite(CTF)/LNCaP-NF1-ChIP-Seq/Homer More Information Similar Motifs Found	motif file (matrix)
3	TTTTATTGGC	1e-384	-8.860e+02	17.77%	10.53%	53.9bp (62.1bp)	Unknown/Homeobox/Limb-p300-ChIP-Seq/Homer More Information Similar Motifs Found	motif file (matrix)
4	CCTGTAAAT	1e-164	-3.783e+02	3.17%	1.28%	52.2bp (62.9bp)	PH0048.1_Hoxa13 More Information Similar Motifs Found	motif file (matrix)
5	ATGACTCA	1e-151	-3.485e+02	3.38%	1.47%	50.2bp (65.4bp)	NF-E2(bZIP)/K562-NFE2-ChIP-Seq/Homer More Information Similar Motifs Found	motif file (matrix)
6	GCCATCTGGTGG	1e-107	-2.485e+02	1.21%	0.35%	56.3bp (69.7bp)	CTCF(Zf)/CD4+ CTCF-ChIP-Seq/Homer More Information Similar Motifs Found	motif file (matrix)
7	AGATAAGATC	1e-72	-1.671e+02	2.10%	1.02%	55.1bp (58.5bp)	MA0029.1_Evi1 More Information Similar Motifs Found	motif file (matrix)

homerResults/ directory: contains files for the homerResults.html webpage, including motif<#>.motif files for use in finding specific instance of each motif.

knownResults.html : formatted output of known motif finding.

knownResults/ directory: contains files for the knownResults.html webpage, including

known<#>.motif files for use in finding specific instance of each motif.

Interpreting motif finding results

The format of the output files generated by **findMotifsGenome.pl** are identical to those generated by the promoter-based version **findMotifs.pl** (description).

In general, when analyzing ChIP-Seq / ChIP-Chip peaks you should expect to see strong enrichment for a motif resembling the site recognized by the DNA binding domain of the factor you are studying. Enrichment p-values reported by HOMER should be very very significant (i.e. $< 1e-50$). If this is not the case, there is a strong possibility that the experiment may have failed in one way or another. For example, the peaks could be of low quality because the factor is not expressed very high.

Practical Tips for Motif finding

Important motif finding parameters

Masked vs. Unmasked Genome ("**-mask**" or hg18 vs. hg18r)

Actually, this usually doesn't matter *that* much. Since HOMER is a differential motif discovery algorithm, common repeats are usually in both the target and background sequences. However, it is not uncommon that a transcription factor binds to a certain class of repeats, which may cause several large stretches of similar sequence to be processed, biasing the results. Usually it's safer to go with the masked version.

Region Size ("**-size <#>**", default: 200)

The size of the region used for motif finding is important. If analyzing ChIP-Seq peaks from a transcription factor, Chuck would recommend 50 bp for establishing the primary motif bound by a given transcription factor and 200 bp for finding both primary and "co-enriched" motifs for a transcription factor. When looking at histone marked regions, 500-1000 bp is probably a good idea (i.e. H3K4me or H3/H4 acetylated regions). In theory, HOMER can work with very large regions (i.e. 10kb), but with the larger the regions comes more sequence and longer execution time.

Motif length ("**-len <#>**" or "**-len <#>,<#>,...**", default 8,10,12)

Specifies the length of motifs to be found. HOMER will find motifs of each size separately and then combine the results at the end. The length of time it takes to find motifs increases greatly with increasing size. In general, it's best to try out enrichment with shorter lengths (i.e. less than 15) before trying longer lengths. Much longer motifs can be found with HOMER, but it's best to use smaller sets of sequence when trying to find long motifs (i.e. use "**-len 20 -size 50**"), otherwise it may take way too long (or take too much memory). The other trick to reduce the total resource consumption is to reduce the number of background sequences (**-N <#>**).

Mismatches allowed in global optimization phase ("**-mis <#>**", default: 2)

HOMER looks for promising candidates by initially checking ordinary oligos for enrichment, allowing mismatches. The more mismatches you allow, the more sensitive the algorithm, particularly for longer motifs. However, this also slows down the algorithm a bit. If searching for motifs longer than 12-15 bp, it's best to increase this value to at least 3 or even 4.

Number of CPUs to use ("**-p <#>**", default 1)

HOMER is now multicore compliant. It's not perfectly parallelized, however, certain types of analysis can benefit. In general, the longer the length of the motif, the better

the speed-up you'll see.

Number of motifs to find ("**-S <#>**", default 25)

Specifies the number of motifs of each length to find. 25 is already quite a bit. If anything, I'd recommend reducing this number, particularly for long motifs to reduce the total execution time.

Normalize CpG% content instead of GC% content ("**-cpg**")

Consider trying if HOMER is stuck finding "CGCGCGCG"-like motifs. You can also play around with disabling GC/CpG normalization ("**-noweight**").

Region level autonormalization ("**-nlen <#>**", default 3, "**-nlen 0**" to disable)

Motif level autonormalization ("**-olen <#>**", default 0 i.e. disabled)

Autonormalization attempts to remove sequence bias from lower order oligos (1-mers, 2-mers ... up to <#>). Region level autonormalization, which is for 1/2/3 mers by default, attempts to normalize background regions by adjusting their weights. If this isn't getting the job done (autonormalization is not guaranteed to remove all sequence bias), you can try the more aggressive motif level autonormalization ("**-olen <#>**"). This performs the autonormalization routine on the oligo table during de novo motif discovery. (see [here](#) for more info)

User defined background regions ("**-bg <peak file of background regions>**")

Why let HOMER randomly pick you background regions when you can choose them yourself!! These will still be normalized for CpG% or GC% content just like randomly chosen sequences and autonormalized unless these options are turned off (i.e. "**-nlen 0 -noweight**"). This can be very useful since HOMER is a differential motif discovery algorithm. For example, you can give HOMER a set of peaks co-bound by another factor and compare them to the rest of the peaks. HOMER will automatically check if the background peaks overlap with the target peaks using **mergePeaks**, and discard overlapping regions.

Hypergeometric enrichment scoring ("**-h**")

By default, **findMotifsGenome.pl** uses the binomial distribution to score motifs. This works well when the number of background sequences greatly outnumber the target sequences - however, if you are using "**-bg**" option above, and the number of background sequences is smaller than target sequences, it is a good idea to use the hypergeometric distribution instead ("**-h**"). FYI - The binomial is faster to compute, hence its use for motif finding in large numbers of regions.

Find enrichment of individual oligos ("**-oligo**").

This creates output files in the output directory named *oligo.length.txt*.

Force **findMotifsGenome.pl** to re-prepare genome for the given region size ("**-prepare**").

In case there is a problem with the existing prepared files, force them to be remade with "**-prepare**".

Only search for motifs on + strand ("**-norevopp**")

By default, HOMER looks for transcription factor-like motifs on both strands. This will force it to only look at the + strand (relative to the peak, so - strand if the peak is on the - strand).

Search for RNA motifs ("**-rna**")

If looking at RNA data (i.e. Clip-Seq or similar), this option will restrict HOMER to only search the + strand (relative to the peak), and will output RNA motif logos (i.e. U instead of T). It will also try to compare found motifs to an RNA motif database, which sadly, only contains miRNAs right now... I guess chuck roundhouse kicked all of the

splicing and other RNA motifs into hard to find databases.

Mask motifs ("**-mask <motif file>**")

Mask the motif(s) in the supplied motif file before starting motif finding. Multiple motifs can be in the motif file.

Optimize motifs ("**-opt <motif file>**")

Instead of looking for novel de novo motifs, HOMER will instead try to optimize the motif supplied. This is cool when trying to change the length of a motif, or find a very long version of a given motif. For example, if you specify "-opt <file>" and "-len 50", it will try to expand the motif to 50bp and optimize it.

Dump FASTA files ("**-dumpFasta**")

Like the fact that HOMER organizes and extracts your sequence files, but don't care for HOMER as a motif finding algorithm? That's cool, just specify "-dumpFasta" and the files "target.fa" and "background.fa" will show up in your output directory. You can then use them with MEME or whatever. Just remember, Chuck knows where you live...

Finding Instance of Specific Motifs

By default, HOMER does not return the locations of each motif found in the motif discovery process. To recover the motif locations, you must first select the motifs you're interested in by getting the "motif file" output by HOMER. You can combine multiple motifs in single file if you like to form a "motif library". To identify motif locations, you have two options:

1. Run **findMotifsGenome.pl** with the "**-find <motif file>**" option. This will output a tab-delimited text file with each line containing an instance of the motif in the target peaks. The output is sent to *stdout*.

For example: **findMotifsGenome.pl ERalpha.peaks hg18 MotifOutputDirectory/ -find motif1.motif > outputfile.txt**

The output file will contain the columns:

1. Peak/Region ID
2. Offset from the center of the region
3. Sequence of the site
4. Name of the Motif
5. Strand
6. Motif Score (log odds score of the motif matrix, higher scores are better matches)

2. Run **annotatePeaks.pl** with the "**-m <motif file>**" option (see the [annotation section](#) for more info). Chuck prefers doing it this way. This will output a tab-delimited text file with each line containing a peak/region and a column containing instance of each motif separated by commas to stdout

For example: **annotatePeaks.pl ERalpha.peaks hg18 -m motif1.motif > outputfile.txt**

The output file will contain columns:

1. Peak/Region ID
2. Chromosome

- 3. Start
- 4. End
- 5. Strand of Peaks

6-18: annotation information

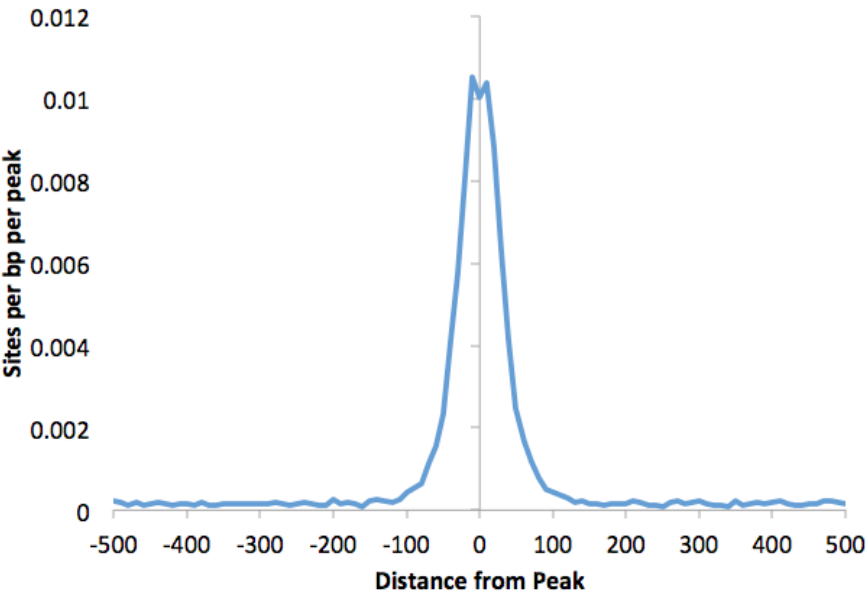
- 19. CpG%
- 20. GC%
- 21. Motif Instances
- ...

Motif Instances have the following format:
<distance from center of region>(<sequence>,<strand>,<conservation>)
i.e -29(TAAATCAACA,+,0.00)

You can also find histogram of motif density this way by adding "-hist <#>" to the command. For example:

```
annotatePeaks.pl ERalpha.peaks hg18 -m ere.motif foxa1.motif -size 1000 -hist 10 > outputfile.txt
```

Graphing the output with EXCEL:



Command-line options for findMotifsGenome.pl

Program will find de novo and known motifs in regions in the genome

Usage: findMotifsGenome.pl <pos file> <genome> <output directory> [additional options]
Example: findMotifsGenome.pl peaks.txt mm8r peakAnalysis -size 200 -len 8

Possible Genomes:

- ...
- Custom: provide the path to genome FASTA files (directory or single file)
Heads up: will create the directory "preparsed/" in same location.

Basic options:

- bg <background position file> (genomic positions to be used as background, default=automatic)
 - removes background positions overlapping with target positions
- chopify (chop up large background regions to the avg size of target regions)
- len <#>[,<#>,<#>...] (motif length, default=8,10,12) [NOTE: values greater 12 may cause the program to run out of memory - in these cases decrease the number of sequences analyzed (-N), or try analyzing shorter sequence regions (i.e. -size 100)]
- size <#> (fragment size to use for motif finding, default=200)
 - size <#,#> (i.e. -size -100,50 will get sequences from -100 to +50 relative from center)
 - size given (uses the exact regions you give it)
- S <#> (Number of motifs to optimize, default: 25)
- mis <#> (global optimization: searches for strings with # mismatches, default: 2)
- norevopp (don't search reverse strand for motifs)
- nomotif (don't search for de novo motif enrichment)
- rna (output RNA motif logos and compare to RNA motif database, automatically sets -norevopp)

Scanning sequence for motifs

- find <motif file> (This will cause the program to only scan for motifs)

Known Motif Options/Visualization

- bits (scale sequence logos by information content, default: doesn't scale)
- nocheck (don't search for de novo vs. known motif similarity)
- mcheck <motif file> (known motifs to check against de novo motifs, default: /bioinformatics/homer/data/knownTFs/all.motifs)
- float (allow adjustment of the degeneracy threshold for known motifs to improve p-value[dangerous])
- noknown (don't search for known motif enrichment, default: -known)
- mknown <motif file> (known motifs to check for enrichment, default: /bioinformatics/homer/data/knownTFs/known.motifs)

Sequence normalization options:

- gc (use GC% for sequence content normalization, now the default)
- cpg (use CpG% instead of GC% for sequence content normalization)
- noweight (no CG correction)

Advanced options:

- h (use hypergeometric for p-values, binomial is default)
- N <#> (Number of sequences to use for motif finding, default=max(50k, 2x input))
- noforce (will attempt to reuse sequence files etc. that are already in output directory)
- local <#> (use local background, # of equal size regions around peaks to use i.e. 2)
- redundant <#> (Remove redundant sequences matching greater than # percent, i.e. -redundant 0.5)
- mask <motif file1> [motif file 2]... (motifs to mask before motif finding)
- opt <motif file1> [motif file 2]... (motifs to optimize or change length of)
- refine <motif file1> (motif to optimize)
- rand (randomize target and background sequences labels)
- ref <peak file> (use file for target and background - first argument is list of peak ids for targets)
- oligo (perform analysis of individual oligo enrichment)
- dumpFasta (Dump fasta files for target and background sequences for use with other programs)
- preparse (force new background files to be created)
- keepFiles (keep temporary files)

homer2 specific options:

- homer2 (use homer2 instead of original homer, default)
- nlen <#> (length of lower-order oligos to normalize in background, default: -nlen 3)
 - nmax <#> (Max normalization iterations, default: 160)
- olen <#> (lower-order oligo normalization for oligo table, use if -nlen isn't working well)
- p <#> (Number of processors to use, default: 1)
- e <#> (Maximum expected motif instance per bp in random sequence, default: 0.01)
- cache <#> (size in MB for statistics cache, default: 500)
- quickMask (skip full masking after finding motifs, similar to original homer)

Original homer specific options:

- homer1 (to force the use of the original homer)
- depth [low|med|high|allnight] (time spent on local optimization default: med)



Can't figure something out? Questions, comments, concerns, or other feedback:
cbenner@ucsd.edu



HOMER

Software for motif discovery and next-gen sequencing analysis

ChIP-Seq Analysis: Analyzing a ChIP-Seq experiment with one command

Even I don't like typing the same commands over and over again. The following command performs the standard set of analysis commands so that you can do better things while your data is processed.

analyzeChIP-Seq.pl <Tag Directory> <genome> [general options] [-A | B | C | D sub-program options]

i.e. a common use: **analyzeChIP-Seq.pl** Factor-ChIP-Seq/ hg18r -i Input-ChIP-Seq -focus -A factor_alignment_file.bed factor_alignment_file2.bed

This command performs 4 separate tasks labeled as A,B,C & D:

- A. Runs **makeTagDirectory** to parse alignment files, set up the tag directory, and performs basic QC such as tag auto correlation and checks for sequence bias.
- B. Runs **makeUCSCfile** and **findPeaks** to generate UCSC Genome Browser files and peak files for the experiment.
- C. Runs **findMotifsGenome.pl** to determine enriched motifs in your ChIP-Seq peaks.
- D. Runs **annotatePeaks.pl** to generate an annotated peak file and performs GO analysis on genes found near the peaks.

As output, this program will create standard files in the "Tag Directory" including an "index.html" file that links you to each of the output files.

There are a couple of general options that can be used with **analyzeChIP-Seq.pl**:

- i <input tag directory> : "Tag directory" to use as a control for peak finding.
- size <#> : Force this peak size for analysis (default is "auto" for peak finding, 200 bp for motif analysis and 50 bp for focused peak analysis)
- focus : This will find enriched motifs in 85% focused peaks using only +/- 25 bp of sequence, useful for identifying the primary motif bound by the factor.
- enhancer : This will set the peak size to "-size 1000" and only perform motif analysis on peaks > 3kb from the TSS.

To specifically tailor the options used by the sub-programs, first enter the "general options" you want above, then enter "-A" followed by the options you want passed to the sub-programs. For example, the most used sub-program option I use is the

following:

"-A s_1_eland_result.txt" - this passes the alignment file to the **makeTagDirectory** program so that it will be used to make the Tag Directory.

If you've already made a tag directory, no need to use the "-A blah blah" option. In similar fashion, to tell the motif finding to check motifs of length 10, 11, and 12, add "**C -len 10,11,12**" to the end of the command.

[Back to ChIP-Seq Analysis](#)



Can't figure something out? Questions, comments, concerns, or other feedback:
cbenner@ucsd.edu



HOMER

Software for motif discovery and next-gen sequencing analysis

Annotating Regions in the Genome (*annotatePeaks.pl*)

Homer contains a useful, all-in-one program for performing peak annotation called **annotatePeaks.pl**. In addition to associating peaks with nearby genes, **annotatePeaks.pl** can perform Gene Ontology Analysis, genomic feature association analysis (Genome Ontology), associate peaks with gene expression data, calculate ChIP-Seq Tag densities from different experiments, and find motif occurrences in peaks. **annotatePeaks.pl** can also be used to create histograms and heatmaps. Description of the annotation functions are covered below, while quantification of tags, motifs, histograms, etc. are covered [here](#).

NOTE: If you're running *annotatePeaks.pl* on your laptop, you may want to use **"-noann"** to skip the full annotation routines, which use a bit of memory (up to 4Gb)

Basic usage:

```
annotatePeaks.pl <peak/BED file> <genome> > <output file>
```

```
i.e. annotatePeaks.pl ERpeaks.txt hg18 > outfile.txt
```

The first two arguments, the *<peak file>* and *<genome>*, are required, and must be the first two arguments. Other optional command line arguments can be placed in any order after the first two. By default, **annotatePeaks.pl** prints the program output to *stdout*, which can be captured in a file by appending " > filename" to the command. With most uses of **annotatePeaks.pl**, the output is a data table that is meant to be opened with EXCEL or similar program. An example of the output can be seen below:

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R
1	PeakID	Chr	Start	End	Strand	Peak Sco	Focus Rg	Annotation	Detailed Anno	Distance to T	Nearest Pror	PromoterID	Nearest Unig	Nearest Refs	Nearest Ense	Gene Name	Gene Alias	Gene Descrip
2	chr18-1	chr18	69007968	69008268	+	593	0.939	intron (NR_03)	intron (NR_03)	74595	NR_034133	400655	Hs.579378	NR_034133		LOC400655	-	hypothetical
3	chr9-1	chr9	88209966	88210266	+	531.9	0.946	Intergenic	Intergenic	-50894	NM_001185	79670	Hs.597057	NM_001185	ENSG000000000000	ZCCHC6	DKFZp666B1	zinc finger, C
4	chr14-1	chr14	62337073	62337373	+	505.4	0.918	intron (NM_17)	intron (NM_17)	244485	NM_172375	27133	Hs.27043	NM_139318	ENSG000000000000	KCNH5	EAG2 H-EAG	potassium vc
5	chr17-1	chr17	5076243	5076543	+	492.1	0.936	intron (NR_03)	intron (NR_03)	2414	NM_207103	388325	Hs.462080	NM_207103	ENSG000000000000	C17orf87	FLJ32580 Mi	chromosome
6	chr17-2	chr17	47851714	47852014	+	476.2	0.824	Intergenic	Intergenic	-259488	NM_001082	56934	Hs.463466	NM_001082	ENSG000000000000	CA10	CA-RPX CAR	carbonic anh
7	chr10-1	chr10	98420680	98420980	+	474.9	0.967	intron (NM_15)	intron (NM_15)	49439	NM_152309	118788	Hs.310456	NM_152309	ENSG000000000000	PIK3AP1	BCAP RP11-	phosphoinos
8	chr9-2	chr9	81294389	81294689	+	456.3	0.957	Intergenic	Intergenic	-82159	NM_007005	7091	Hs.444213	NM_007005	ENSG000000000000	TLE4	BCE-1 BCE1	transducin-II
9	chr14-2	chr14	36817736	36818036	+	452.3	0.757	intron (NM_13)	intron (NM_13)	81017	NM_001195	145282	Hs.660396	NM_001195	ENSG000000000000	MIPOL1	DKFZp313M	mirror-image
10	chr18-2	chr18	20049825	20050125	+	449.7	0.853	intron (NM_06)	intron (NM_06)	56219	NM_018030	114876	Hs.370725	NM_018030	ENSG000000000000	OSBP1A	FLJ10217 OF	oxysterol bin
11	chr7-1	chr7	12226829	12227129	+	445.7	0.901	intron (NM_01)	intron (NM_01)	9606	NM_001134	54664	Hs.396358	NM_001134	ENSG000000000000	TMEM1068	FLJ11273 Mi	transmembri
12	chr14-3	chr14	88712188	88712488	+	443.1	0.844	intron (NM_0C)	intron (NM_0C)	240869	NM_005197	1112	Hs.621371	NM_001085	ENSG000000000000	FOXN3	C14orf116 C	forkhead box
13	chr18-3	chr18	62951924	62952224	+	443.1	0.947	Intergenic	Intergenic	-382689	NR_033921	643542	Hs.652901	NR_033921		LOC643542	-	hypothetical
14	chr3-1	chr3	32196769	32197069	+	443.1	0.87	Intergenic	Intergenic	-58256	NM_178868	152189	Hs.154986	NM_178868	ENSG000000000000	CMTM8	CKLFSF8 CKL	CKLF-like MA
15	chr11-1	chr11	110685448	110685748	+	425.8	0.907	Intergenic	Intergenic	-9849	NR_034154	399948	Hs.729225	NR_034154		C11orf92	DKFZp781P1	chromosome
16	chr4-1	chr4	81755366	81755666	+	423.2	0.908	intron (NM_15)	intron (NM_15)	279618	NM_152770	255119	Hs.527104	NM_152770	ENSG000000000000	C4orf22	MGC35043	chromosome

Acceptable Input files

annotatePeaks.pl accepts [HOMER peak files](#) or [BED files](#):

HOMER peak files should have at minimum 5 columns (separated by TABs, additional columns will be ignored):

- Column1: Unique Peak ID
- Column2: chromosome
- Column3: starting position
- Column4: ending position
- Column5: Strand (+/- or 0/1, where 0="+", 1="-")

BED files should have at minimum 6 columns (separated by TABs, additional columns will be ignored)

- Column1: chromosome
- Column2: starting position
- Column3: ending position
- Column4: Unique Peak ID
- Column5: not used

- Column6: Strand (+/- or 0/1, where 0="+", 1="-")

In theory, HOMER will accept BED files with only 4 columns (+/- in the 4th column), and files without unique IDs, but this is NOT recommended.

Mac Users: If using a EXCEL to prepare input files, make sure to save files as a "Text (Windows)" if running MacOS - saving as "Tab delimited text" in Mac produces problems for the software. Otherwise, you can run the script "**changeNewLine.pl** <filename>" to convert the Mac-formatted text file to a Windows/Dos/Unix formatted text file.

If errors occur, it is likely that the file is not in the correct format, or the first column is not actually populated with unique identifiers.

How Basic Annotation Works

The process of annotating peaks/regions is divided into two primary parts. The first determines the distance to the nearest TSS and assigns the peak to that gene. The second determines the genomic annotation of the region occupied by the center of the peak/region.

Distance to the nearest TSS

By default, **annotatePeaks.pl** loads a file in the *"/path-to-homer/data/genomes/<genome>/<genome>.tss"* that contains the positions of RefSeq transcription start sites. It uses these positions to determine the closest TSS, reporting the distance (negative values mean upstream of the TSS, positive values mean downstream), and various annotation information linked to locus including alternative identifiers (unigene, entrez gene, ensembl, gene symbol etc.). This information is also used to link gene-specific information (see below) to a peak/region, such as gene expression.

Genomic Annotation

To annotate the location of a given peak in terms of important genomic features, **annotatePeaks.pl** calls a separate program (**assignGenomeAnnotation**) to efficiently assign peaks to one of millions of possible annotations genome wide. Two types of output are provided. The first is "Basic Annotation" that includes whether a peak is in the TSS (transcription start site), TTS (transcription termination site), Exon (Coding), 5' UTR Exon, 3' UTR Exon, Intronic, or Intergenic, which are common annotations that many researchers are interested in. A second round of "Detailed Annotation" also includes more detailed annotation, also considering repeat elements and CpG islands. Since some annotation overlap, a priority is assign based on the following (in case of ties it's random [i.e. if there are two overlapping repeat element annotations]):

1. TSS (by default defined from -1kb to +100bp)
2. TTS (by default defined from -100 bp to +1kb)
3. CDS Exons
4. 5' UTR Exons
5. 3' UTR Exons
6. **CpG Islands
7. **Repeats
8. Introns
9. Intergenic

** Only applicable for the "Detailed Annotation".

Although HOMER doesn't allow you to explicitly change the definition of the region that is the TSS (-1kb to +100bp), you can "do it yourself" by sorting the annotation output in EXCEL by the "Distance to nearest TSS" column, and selecting those within the range you are interested in.

Using Custom Annotations

Custom Gene Annotation Definition using GTF files

RefSeq doesn't do it for everyone. There are many other quality gene annotations out there, including UCSC genes, Ensembl, and Gencode to name a couple. Even more important, as RNA-Seq methods develop, the locations of exons etc. can be defined based on your own experimental RNA data rather than using a static database to define transcripts. You can download GTF files for various gene definitions from the UCSC Genome Browser in the "[Table Browser](#)" section. Custom GTF files can be created from RNA-Seq data using

tools like [Cufflinks](#).

HOMER can process GTF (Gene Transfer Format) files and use them for annotation purposes ("**-gtf <gtf filename>**"). If a GTF file is specified, HOMER will parse it and use the TSS from the GTF file for determining the distance to the nearest TSS. It will also use the GTF file's definition of TSS/TTS/exons/Introns for Basic Genome Annotation. The original HOMER annotation files are still used for the "Detailed Annotation" since the repeats will not be define in the GTF files.

i.e. **annotatePeaks.pl ERpeaks.txt hg18 -gtf gencode.gtf > outputfile.txt**

HOMER will try it's best to take the "transcript_id" from the GTF definition and translate it into a known gene identifier. If it can't match it to a known gene, many of the annotation columns corresponding to Unigene etc. will be empty.

HOMER can also work with GFF and GFF3 files - to a degree. The problem with these is that the formats do not have the same strict naming convention enforced in GTF files. To use them, substitute "**-gtf <GTF file>**" with "**-gff <GFF file>**" or "**-gff3 <GFF3 file>**"

Custom Promoter Locations

By default, **annotatePeaks.pl** assigns peaks to the nearest TSS. If you have a custom peak/pos file of these locations, you can supply that with the option "**-cTSS <peak/pos file>**". This will override the default, which is RefSeq TSS.

Using Custom Genomes/Annotations

For organisms with relatively incomplete genomes, annotatePeaks.pl can still provide some functionality. If the genome is not available as a pre-configured genome in HOMER, then you can supply the path to the full genome FASTA file or path to directory containing chromosome FASTA files as the 2nd argument. For example, lets say you were sequencing the [banana slug](#) genome, and had downloaded the current draft of the genome into the file "bananaSlug.fa". You could then run annotatePeaks.pl like this:

annotatePeaks.pl chip-seq-peaks.txt bananaSlug.fa > output.txt

If no genome sequence is available, you can also specify "**none**":

annotatePeaks.pl chip-seq-peaks.txt none > output.txt

You may also find a custom annotation file for the organism, such as banana_slug_genes.gtf, or banana_slug_genes.gff from the community website. This can then be used to help annotate your data:

annotatePeaks.pl chip-seq-peaks.txt bananaSlug.fa -gtf banana_slug_genes.gtf > output.txt
or
annotatePeaks.pl chip-seq-peaks.txt bananaSlug.fa -gff banana_slug_genes.gff > output.txt

In the case of custom genomes, HOMER will not be able to convert gene IDs to names - you may have to do this yourself, but many of the other features, including motif finding etc., are available as long as you have the sequence.

Basic Annotation File Output

Description of Columns:

1. Peak ID
2. Chromosome
3. Peak start position
4. Peak end position
5. Strand
6. Peak Score
7. FDR/Peak Focus Ratio/Region Size
8. Annotation (i.e. Exon, Intron, ...)
9. Detailed Annotation (Exon, Intron etc. + CpG Islands, repeats, etc.)

10. Distance to nearest RefSeq TSS
11. Nearest TSS: Native ID of annotation file
12. Nearest TSS: Entrez Gene ID
13. Nearest TSS: Unigene ID
14. Nearest TSS: RefSeq ID
15. Nearest TSS: Ensembl ID
16. Nearest TSS: Gene Symbol
17. Nearest TSS: Gene Aliases
18. Nearest TSS: Gene description
19. *Additional columns depend on options selected when running the program.*

As of now, basic annotation is based on alignments of RefSeq transcripts to the UCSC hosted genomes.

Adding Gene Expression Data

annotatePeaks.pl can add gene-specific information to peaks based on each peak's nearest annotated TSS. To add gene expression or other data types, first create a gene data file (tab delimited text file) where the first column contains gene identifiers, and the first row is a header describing the contents of each column. In principle, the contents of these columns doesn't matter. To add this information to the annotation result, use the "**-gene <gene data file>**".

```
annotation.pl <peak file> <genome> -gene <gene data file> > output.txt
```

For peaks that are near genes with associated data in the "gene data file", this data will be appended to the end of the row for each peak.

Peak Annotation Enrichment

Gene Ontology Analysis of Associated Genes

annotatePeaks.pl offers two types of annotation enrichment analysis. The first is based on Gene Ontology classifications for genes. The idea is that your regions (i.e. ChIP-Seq peaks) might be preferentially found near genes with specific biological functions. By specifying "**-go <GO output directory>**", HOMER will take the list of genes associated with your regions and search for enriched functional categories, placing the output in the indicated directory. This is just like looking for GO enrichment in a set of regulated genes ([covered in greater detail here](#)).

There are some caveats to this type of analysis - for example, some genes are simply more likely to be considered in the analysis if there are isolated along the genome where they are likely to be the "closest gene" for peaks in the region. But... since gene density roughly correlates with transcription factor density, it works itself out to some degree. Other tools, such as [GREAT](#) attempt to deal with this problem to some degree, and are worth trying. My experience is that the results are not that different once you factor in the fact that most other tools use "GO slims".

Genome Ontology: Looking for Enriched Genomic Annotations

The **Genome Ontology** looks for enrichment of various genomic annotations in your list of peaks/regions. Just as the Gene Ontology contains groups of genes sharing a biological function, the Genome Ontology contains groups of genomic regions sharing an annotation, such as TSS, or LINE repeats, coding exons, Transcription factor peaks, etc. To run the Genome Ontology, add the option "**-genomeOntology <output directory>**". A much more detailed description of the Genome Ontology will be available [here](#) (coming soon).

Command line options for annotatePeaks.pl

Usage: `annotatePeaks.pl <peak file | tss> <genome version> [additional options...]`

Available Genomes (required argument): (name,org,directory,default promoter set)

-- or --

Custom: provide the path to genome FASTA files (directory or single file)

User defined annotation files (default is UCSC refGene annotation):

annotatePeaks.pl accepts GTF (gene transfer formatted) files to annotate positions relative to custom annotations, such as those from de novo transcript discovery or Gencode.
-gtf <gtf format file> (-gff and -gff3 can work for those files, but GTF is better)

Peak vs. tss/tts/rna mode (works with custom GTF file):

If the first argument is "tss" (i.e. annotatePeaks.pl tss hg18 ...) then a TSS centric analysis will be carried out. Tag counts and motifs will be found relative to the TSS. (no position file needed) ["tts" now works too - e.g. 3' end of gene]
["rna" specifies gene bodies, will automaticall set "-size given"]
NOTE: The default TSS peak size is 4000 bp, i.e. +/- 2kb (change with -size option)
-list <gene id list> (subset of genes to perform analysis [unigene, gene id, accession, probe, etc.], default = all promoters)
-cTSS <promoter position file i.e. peak file> (should be centered on TSS)

Primary Annotation Options:

- m <motif file 1> [motif file 2] ... (list of motifs to find in peaks)
- mscore (reports the highest log-odds score within the peak)
- nmotifs (reports the number of motifs per peak)
- mdist (reports distance to closest motif)
- mfasta <filename> (reports sites in a fasta file - for building new motifs)
- fm <motif file 1> [motif file 2] (list of motifs to filter from above)
- rmrevopp <#> (only count sites found within <#> on both strands once, i.e. palindromic)
- matrix <prefix> (outputs a motif co-occurrence files:
 - prefix.count.matrix.txt - number of peaks with motif co-occurrence
 - prefix.ratio.matrix.txt - ratio of observed vs. expected co-occurrence
 - prefix.logPvalue.matrix.txt - co-occurrence enrichment
 - prefix.stats.txt - table of pair-wise motif co-occurrence statisticsadditional options:
 - matrixMinDist <#> (minimum distance between motif pairs - to avoid overlap)
 - matrixMaxDist <#> (maximum distance between motif pairs)
- mbed <filename> (Output motif positions to a BED file to load at UCSC (or -mpeak))
- d <tag directory 1> [tag directory 2] ... (list of experiment directories to show tag counts for) NOTE: -dfile <file> where file is a list of directories in first column
- bedGraph <bedGraph file 1> [bedGraph file 2] ... (read coverage counts from bedGraph files)
- p <peak file> [peak file 2] ... (to find nearest peaks)
 - pdist to report only distance (-pdist2 gives directional distance)
 - pcount to report number of peaks within region
- vcf <VCF file> (annotate peaks with genetic variation information, one col per individual)
 - editDistance (Computes the # bp changes relative to reference)
 - individuals <name1> [name2] ... (restrict analysis to these individuals)
- gene <data file> ... (Adds additional data to result based on the closest gene. This is useful for adding gene expression data. The file must have a header, and the first column must be a GeneID, Accession number, etc. If the peak cannot be mapped to data in the file then the entry will be left empty.
- go <output directory> (perform GO analysis using genes near peaks)
- genomeOntology <output directory> (perform genomeOntology analysis on peaks)
 - gsize <#> (Genome size for genomeOntology analysis, default: 2e9)

Annotation vs. Histogram mode:

- hist <bin size in bp> (i.e 1, 2, 5, 10, 20, 50, 100 etc.)

The -hist option can be used to generate histograms of position dependent features relative to the center of peaks. This is primarily meant to be used with -d and -m options to map distribution of motifs and ChIP-Seq tags. For ChIP-Seq peaks for a Transcription factor you might want to use the -center option (below) to center peaks on the known motif
** If using "-size given", histogram will be scaled to each region (i.e. 0-100%), with the -hist parameter being the number of bins to divide each region into.

Histogram Mode specific Options:

- nuc (calculated mononucleotide frequencies at each position,
 - Will report by default if extracting sequence for other purposes like motifs)
- di (calculated dinucleotide frequencies at each position)
- histNorm <#> (normalize the total tag count for each region to 1, where <#> is the minimum tag total per region - use to avoid tag spikes from low coverage)
- ghist (outputs profiles for each gene, for peak shape clustering)

-rm <#> (remove occurrences of same motif that occur within # bp)

Peak Centering: (other options are ignored)

- center <motif file> (This will re-center peaks on the specified motif, or remove peak if there is no motif in the peak. ONLY recentering will be performed, and all other options will be ignored. This will output a new peak file that can then be reanalyzed to reveal fine-grain structure in peaks (It is advised to use -size < 200) with this to keep peaks from moving too far (-mirror flips the position)
- multi (returns genomic positions of all sites instead of just the closest to center)

Advanced Options:

- len <#> / -fragLength <#> (Fragment length, default=auto, might want to set to 0 for RNA)
- size <#> (Peak size[from center of peak], default=inferred from peak file)
 - size #,# (i.e. -size -10,50 count tags from -10 bp to +50 bp from center)
 - size "given" (count tags etc. using the actual regions - for variable length regions)
- log (output tag counts as $\log_2(x+1+rand)$ values - for scatter plots)
- sqrt (output tag counts as $\sqrt{x+rand}$ values - for scatter plots)
- strand <+|-|both> (Count tags on specific strands relative to peak, default: both)
- pc <#> (maximum number of tags to count per bp, default=0 [no maximum])
- cons (Retrieve conservation information for peaks/sites)
- CpG (Calculate CpG/GC content)
- ratio (process tag values as ratios - i.e. chip-seq, or mCpG/CpG)
- nfr (report nucleosome free region scores instead of tag counts, also -nfrSize <#>)
- norevopp (do not search for motifs on the opposite strand [works with -center too])
- noadj (do not adjust the tag counts based on total tags sequenced)
- norm <#> (normalize tags to this tag count, default=1e7, 0=average tag count in all directories)
- pdist (only report distance to nearest peak using -p, not peak name)
- map <mapping file> (mapping between peak IDs and promoter IDs, overrides closest assignment)
- noann, -nogene (skip genome annotation step, skip TSS annotation)
- homer1/-homer2 (by default, the new version of homer [-homer2] is used for finding motifs)



Can't figure something out? Questions, comments, concerns, or other feedback:
cbenner@ucsd.edu



HOMER

Software for motif discovery and next-gen sequencing analysis

Quantifying Data and Motifs and Comparing Peaks/Regions in the Genome

Homer contains a useful, all-in-one program for performing peak annotation called **annotatePeaks.pl**. In addition to associating peaks with nearby genes, **annotatePeaks.pl** can perform Gene Ontology Analysis, genomic feature association analysis (Genome Ontology), associate peaks with gene expression data, calculate ChIP-Seq Tag densities from different experiments, and find motif occurrences in peaks. **annotatePeaks.pl** can also be used to create histograms and heatmaps. Description of the annotation functions are covered [here](#), while quantification of tags, motifs, histograms, etc. are covered below.

Basic usage (see [Annotation](#)):

```
annotatePeaks.pl <peak/BED file> <genome> [options] > <output file>
```

i.e. `annotatePeaks.pl ERpeaks.txt hg18 > outputfile.txt`

Everything packed into one program

The great thing about `annotatePeaks.pl` is that it combines many features into a single location. It is the primary program to investigate how sequencing reads, sequence motifs, and other information and annotations interact.

Three primary options are available to specify types of data that can be processed by `annotatePeaks.pl`:

```
-d <tag directory1> [tag directory 2] ...
-m <motif file 1> [motif file 2] ...
-p <peak/BED file 1> [peak/BED file 2] ...
```

By default, these data types are processed relative to each peak/region provided in the primary input file. There are a bunch of options that help fine tune how each type of data is considered by the program covered below.

However, `annotatePeaks.pl` can take the same input data and do other things, such as make histograms and heatmaps, allowing you to explore the data in a different way.

```
-hist <# bin size>
```

Acceptable Input files

annotatePeaks.pl accepts [HOMER peak files](#) or [BED files](#):

HOMER peak files should have at minimum 5 columns (separated by TABs, additional columns will be ignored):

- Column1: Unique Peak ID
- Column2: chromosome
- Column3: starting position
- Column4: ending position
- Column5: Strand (+/- or 0/1, where 0="+", 1="-")

BED files should have at minimum 6 columns (separated by TABs, additional columns will be ignored)

- Column1: chromosome
- Column2: starting position
- Column3: ending position
- Column4: Unique Peak ID
- Column5: not used
- Column6: Strand (+/- or 0/1, where 0="+", 1="-")

In theory, HOMER will accept BED files with only 4 columns (+/- in the 4th column), and files without unique IDs, but this is NOT recommended. For one, if you don't have unique IDs for your regions, it's hard to go back and figure out which region contains which peak.

Mac Users: If using a EXCEL to prepare input files, make sure to save files as a "Text (Windows)" if running MacOS - saving as "Tab delimited text" in Mac produces problems for the software. Otherwise, you can run the script "**changeNewLine.pl** <filename>" to convert the Mac-formatted text file to a Windows/Dos/Unix formatted text file.

If errors occur, it is likely that the file is not in the correct format, or the first column is not actually populated with unique identifiers.

TSS Mode

Instead of supplying a peak file, HOMER makes it easy to do analysis of features near TSS. To analyze transcription start sites as if they were peaks, use **"tss"** as the first argument.

```
annotatePeaks.pl tss hg18 ...
```

To restrict the analysis to a subset of TSS promoters, add the option **"-list <file>"** where the file is a Tab-delimited text file with the first column containing gene identifiers. TSS mode also works with custom gene definitions specified with **"-gtf <GTF file>"**.

Specifying the Peak Size - the most important parameter

Pretty much everything explained in the following sections depends heavily on the **"-size <#>"** parameter. A couple of quick notes:

- size <#> : will perform analysis on the # bp surrounding the peak centers [example: -size 1000]
- size <#,#> : will perform analysis from # to # relative to peak center [example: -size -200,50]
- size given : will perform analysis on different sized peaks - size given by actual coordinates in peak/BED file [example: -size given]

For example, if you peaks are actually transcription start sites, you might want to specify **"-size -500,100"** to perform the analysis upstream -500 bp to +100 bp downstream. If your peaks/regions are actually "transcript" regions, specifying **"-size given"** will count reads along the entire transcript. If it doesn't make sense, watch Delta Force I and II back to back. That should numb the brain enough to get it.

Annotating Individual Peaks

Calculating ChIP-Seq Tag Densities across different experiments

annotatePeaks.pl is useful program for cross-referencing data from multiple experiments. In order to count the number of tags from different sequencing experiments, you must first [create tag directories](#) for each of these experiments. Once created, tag counts from these directories in the vicinity of your peaks can be added by specifying **"-d <tag directory 1> <tag directory 2> ..."**. You can specify as many tag directories as you like. Tag totals for each directory will be placed in new columns starting on column 18. For example:

```
annotatePeaks.pl pu1peaks.txt mm8 -size 400 -d Macrophage-PU.1/ Bcell-PU.1/ > output.txt
```

output.txt, when opened in EXCEL, will look like this:

N	O	P	Q	R	S	T	U	V	W	X	Y
est Refs	Nearest Ense	Gene Name	Gene Alias	Gene Descrip	Macrophage-PU.1/ Tag C	Bcell-PU.1/ Tag	Count in 400 bp	(8861792 Total, normalization factor = 1.13, effe			
010133	ENSMUSG000	Gm11487	OTTMUSG000	predicted ger	3268.14		8209.4				
010455	ENSMUSG000	Zfp868	AI449175	AV zinc finger pr	4919.21		11197.51				
025423	ENSMUSG000	1110059E24f	-	RIKEN cDNA 1	3.78		1005.44				
010924	ENSMUSG000	Nnmt	-	nicotinamide	935.1		1751.34				
011412	ENSMUSG000	Slit3	Slit2	slit homolog	3.78		804.58				
077662	ENSMUSG000	Ctso	A330105D01	cathepsin O	5.67		920.81				
028810	ENSMUSG000	Rnd3	2610017M01	Rho family G	846.32		1663.32				
011518	ENSMUSG000	Syk	-	spleen tyrosi	627.18		1649.78				

HOMER automatically normalizes each directory by the total number of mapped tags such that each directory contains **10 million tags**. This total can be changed by specifying **"-norm <#>"** or by specifying **"-noadj"**, which will skip this normalization step.

The other important parameter when counting tags is to specify the size of the region you would like to count tags in with **"-size <#>"**. For example, **"-size 1000"** will count tags in the 1kb region centered on each peak, while **"-size 50"** will count tags in the 50 bp region centered on the peak (default is 200). The number of tags is not normalized by the size of the region.

One last thing to keep in mind is that in order to fairly count tags, HOMER will automatically center tags based on their estimated ChIP-fragment lengths. This is can be overridden by specifying a fixed ChIP-fragment length using **"-len <#>"** or **"-fragLength <#>"**. This is important to consider when trying to count RNA tags, or things such as 5' RNA CAGE/TSS-Seq, where you may want to specify **"-len 0"** so that HOMER doesn't try to move the tags before counting them.

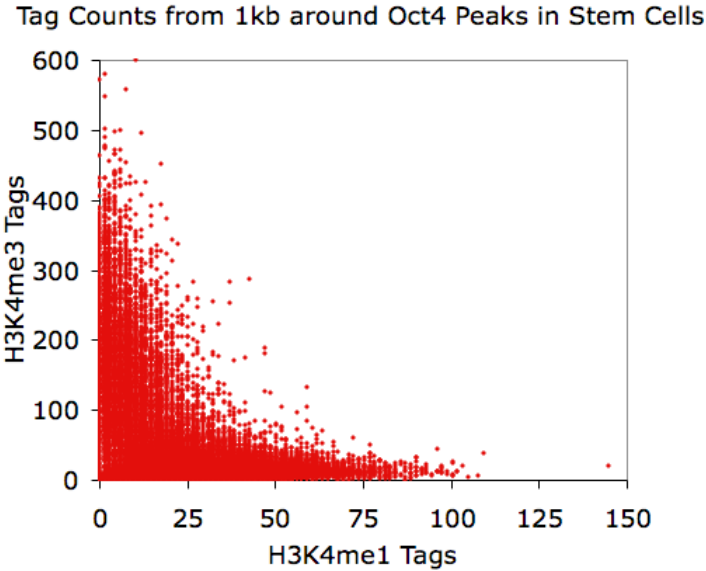
Making Scatter Plots

X-Y scatter plots are a great way to present information, and by counting tag densities from different tag directories, you can visualize the relative levels of different sequencing experiments. Because of the large range of values, it can be difficult to appreciate the relationship between data sets without log transforming the data (or sqrt to stay Poisson friendly). Also, due to the digital nature of tag counting, it can be hard to properly assess the data from a X-Y scatter plot since may of the data points will have the same values and overlap. To assist with these issues, you can specify **"-log"** or **"-sqrt"** to transform the data. These functions will actually report "log(value+1+rand)" and "sqrt(value+rand)", respectively, where rand is a random

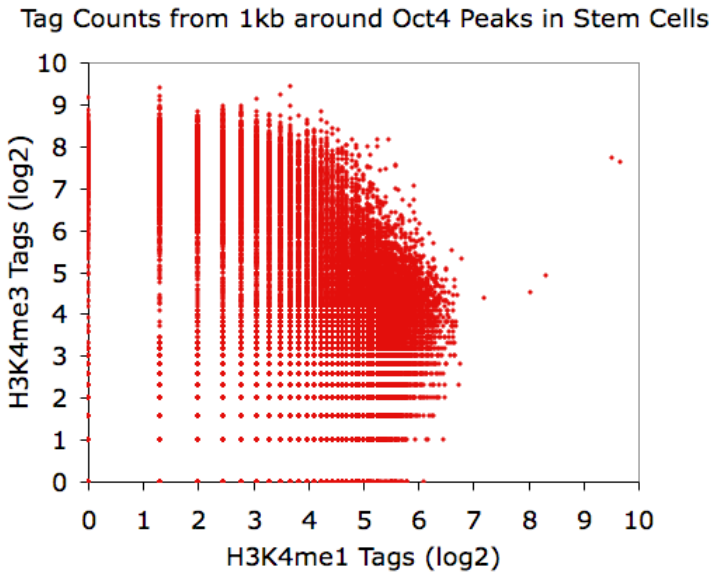
"fraction of a tag" that adds *jitter* to your data so that data points with low tag counts will not have exactly the same value. For example, lets look at the distribution of H3K4me1 and H3K4me3 near Oct4 peaks in mouse embryonic stem cells:

```
annotatePeaks.pl Oct4.peaks.txt mm8 -size 1000 -d H3K4me1-ChIP-Seq/ H3K4me3-ChIP-Seq/ > output.txt
```

Opening output.txt with EXCEL and plotting the last two columns:



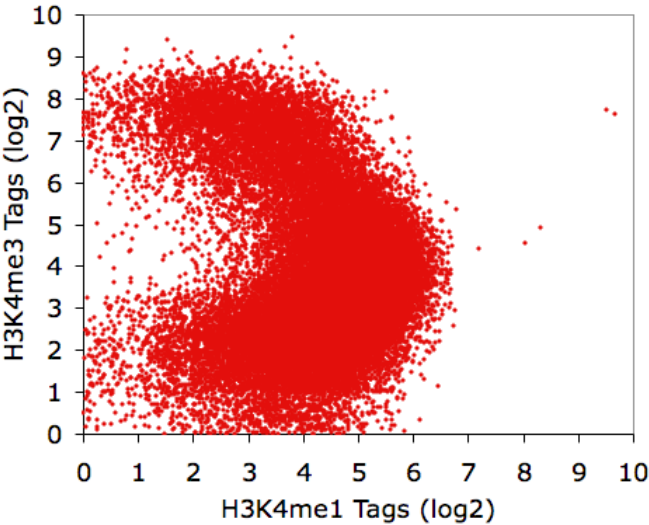
Using EXCEL to take the log(base 2) of the data:



Now using the "-log" option:

```
annotatePeaks.pl Oct4.peaks.txt mm8 -size 1000 -log -d H3K4me1-ChIP-Seq/ H3K4me3-ChIP-Seq/ > output.txt
```

Tag Counts from 1kb around Oct4 Peaks in Stem Cells



Believe it or not, all of these X-Y plots show the same data. Interesting, eh?

Finding instances of motifs near peaks

Figuring out which peaks have instances of [motifs found with findMotifsGenome.pl](#) is very easy. Simply use **"-m <motif file1> <motif file 2>..."** with **annotatePeaks.pl**. (Motif files can be concatenated into a single file for ease of use) This will search for each of these motifs near each peak in your peak file. Use **"-size <#>"** to specify the size of the region around the peak center you wish to search. Found instances of each motif will be reported in additional columns of the output file. For example:

```
annotatePeaks.pl pu1peaks.txt mm8 -size 200 -m pu1.motif cebp.motif > output.txt
```

Opening output.txt with EXCEL:

O	P	Q	R	S	T	U	V
arest	Ense	Gene Name	Gene Alias	Gene Descrip	CpG%	GC%	PU.1/ThioMac PU.1-ChIP-Seq/Homer Distance From Peak(sequence, strand, conservation)
3MUSG000	Dsm1	1700022L091DSN1	MIND 1	0.015	0.621891	-55(GGAGGAAAGTG, +0.00), -26(TGGGGAAAGTG, +0.00), 35(GCAGGAAAGTG, +0.00)	CEBP/CEBPb-ChIP-Seq/Homer Dist
3MUSG000	Cd53	AI3236591	CD53 antigen	0.01	0.432836	34(TGAGGAAAGTG, +0.00)	
3MUSG000	Atg4c	App4-C	App4 autophagy-re	0.01	0.378109	8(AGAGGAAAGTG, +0.00), 54(CACTTCCTCT, -0.00)	
3MUSG000	Tax1bp1	1200003J11F	Tax1 (humar	0.019608	0.307692		-28(ATTTCATAC, +0.00)
3MUSG000	Plekho2	AI840980	plekstrin ho	0.005	0.532338	-11(TGGGGAAAGTG, +0.00), 33(TGAGGAAAGTG, +0.00)	
3MUSG000	Cttnbp2nl	AA552995	CTTNBP2 N-b	0.01	0.462687	-97(GGAGGAAAGTG, +0.00), 48(ACAGGAAAGTG, +0.00)	
3MUSG000	Pitp	OD107	phospholipid	0.005	0.467662	29(CAGTTCTCTT, -0.00)	
3MUSG000	Ifngr2	Ifgr2	interferon ga	0.01	0.462687	-24(CACTTCCTCA, -0.00), 49(CACTTCCTCA, -0.00)	
3MUSG000	Aff2	Fmr2	Ox19(AFA/FMR2 fai	0	0.512438		
3MUSG000	Syk	-	spleen tyrosi	0.01	0.497512	40(AGGGGAAAGTG, +0.00), 60(GGAGGAAAGTG, +0.00)	
3MUSG000	Exoc6	4833405E05	exocyst comp	0.015	0.432836	-41(CAGTTCTCTCT, -0.00)	
3MUSG000	Ccdc109b	9030408N13	coiled-coil do	0.005	0.482587	71(GGAGGAAAGTG, +0.00)	53(6TTGTGTAAAG, -0.00)
3MUSG000	Cttnnd2	Catnd2	Npraj catenin (cad	0.021858	0.472826	-54(AGCGGAAAGTG, +0.00), 18(CAGTTCTCTGT, -0.00), 34(CACTTCCTCT, -0.00)	64(6TTTCATAAT, -0.00)
3MUSG000	Slc7a1	Slc7a1	Slc7a1	0.02	0.547564	-28(AGAGGAAAGTG, +0.00), -16(CACTTCCTCT, -0.00)	

Each instance of the motif is specified in the following format (separated by commas):

Distance from Peak Center(Sequence Matching Motif, Strand, Average Conservation)

The average conservation will not be reported unless you specify **"-cons"**. Also, when finding motifs, the average CpG/GC content will automatically be reported since it has to extract peak sequences from the genome anyway.

There are also a bunch of motif specific options for specialized analysis:

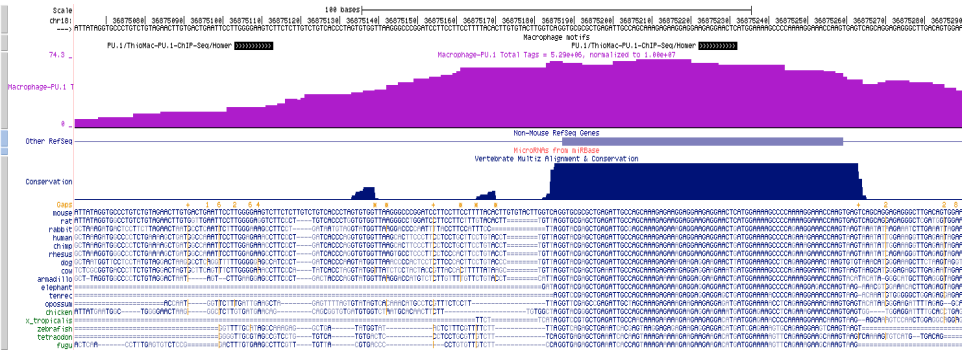
- "-norevopp" (only search + strand relative to peak strand for motifs)
- "-nmotifs" (just report the total number of motifs per peak)
- "-mscore" (report the maximum log-odds score of the motif in each peak)
- "-rmrevopp <#>" (tries to avoid double counting reverse opposites within # bp)
- "-mdist" (reports distance to closest motif)
- "-fm <motif file 1> [motif file 2]" (list of motifs to filter out of results if found)
- "-mfasta <filename>" (reports sites in a fasta file - for building new motifs)
- "-mbed <filename>" (Output motif positions to a BED file to load at UCSC - see below)
- "-matrix <filename>" (outputs a motif co-occurrence matrix with the p-value of co-occurrence assuming instance of each motif are independently distributed amongst the peaks)

Visualizing Motif positions in the UCSC Genome Browser

This feature may seem slightly out of place, but since **annotatePeaks.pl** is the workhorse of HOMER, you can add **"-mbed <filename>"** in conjunction with **"-m <motif file 1> [motif file 2] ..."** to produce a BED file describing motif positions near

peaks that can be loaded as a custom track in the UCSC genome browser. In the example below, you would load **motif.bed** as a custom track:

```
annotatePeaks.pl pu1peaks.txt mm8 -size 200 -m pu1.motif cebp.motif -mbed motif.bed > output.txt
```



Finding the distance to other sets of Peaks

In order to find the nearest peak from another set of peaks, use **"-p <peak file 1> [peak file 2] ..."**. This will add columns to the output spreadsheet that will specify the nearest peak ID and the distance to that peak. If all you want is the distance (so you can sort this column), add the option **"-pdist"** to the command. Otherwise, if you prefer to count the number of peaks in the peak file found within the indicated regions (i.e. with **"-size <#>"**), add **"-pcount"**.

Creating Histograms from High-throughput Sequencing data and Motifs

HOMER can be used to make histograms that document sequencing library and motif densities relative to specific positions in the genome. This can be done near peaks, subsets of peaks, or near promoters, exon junctions or anywhere else you find interesting. To make histograms, use the **annotatePeaks.pl** program but add the parameters **"-hist <#>"** to produce a tab delimited text file that can then be visualized using EXCEL or other data visualization software.

Basic usage:

```
annotatePeaks.pl <peak file> <genome> -size <#> -hist <#> -d <tag directory 1> [tag directory2] ... -m <motif 1> <motif 2> ... > <output matrix file>
```

i.e. **annotatePeaks.pl ERpeaks.txt hg18 -size 6000 -hist 25 -d MCF7-H3K4me1/ MCF7-H3K4me2/ MCF7-H3K4me3/ > outfile.txt**

Running this command is very similar to creating annotated peak files - in fact, most of the data can be used to make both types of files - hence the reason for combining this functionality in the same command. Be default, HOMER normalizes the output histogram such that the resulting units are **per bp per peak**, on top of the standard total mapped tag normalization of **10 million tags**.

Histograms of Tag Directories:

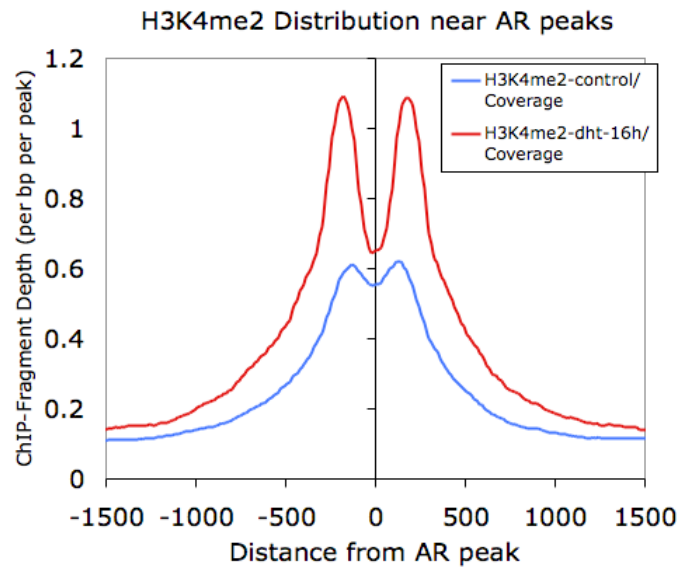
For each tag directory or motif, HOMER will output 3 columns in the histogram. In the case of tag directories, the first column will indicate **ChIP-Fragment Coverage**, which is calculated by extending tags by their estimated ChIP-fragment length, and is analogous to the profiles made for the UCSC Genome Browser. The 2nd and 3rd columns report the density of 5' and 3' independent tags, and are independent of fragment length. For example, lets look at H3K4me2 distribution near Androgen Receptor (AR) peaks before and after 16 hours of treatment with testosterone (dht):

```
annotatePeaks.pl ARpeaks.txt hg18 -size 4000 -hist 10 -d H3K4me2-control/ H3K4me2-dht-16h/ > outputfile.txt
```

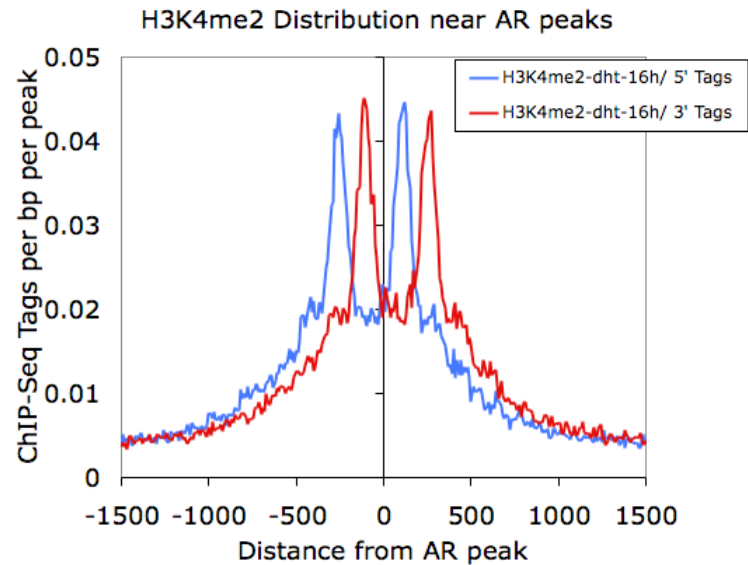
Opening outputfile.txt with EXCEL, we see:

	A	B	C	D	E	F	G
1	Distance from Center H3K4me2-control/	Coverage H3K4me2-control/	5' Tags H3K4me2-control/	3' Tags H3K4me2-dht-16h/	Coverage H3K4me2-dht-16h/	5' Tags H3K4me2-dht-16h/	3' Tags
2	-2000	0.046971	0.001564	0.001734	0.059736	0.001824	0.002136
3	-1990	0.052394	0.003621	0.003179	0.06552	0.004344	0.002928
4	-1980	0.05661	0.003689	0.003298	0.070872	0.004008	0.003528
5	-1970	0.059721	0.003026	0.003128	0.075048	0.003816	0.003912
6	-1960	0.063614	0.003638	0.002975	0.079128	0.00396	0.00372
7	-1950	0.066742	0.002958	0.003298	0.084048	0.004392	0.00372
8	-1940	0.069751	0.003315	0.003145	0.087792	0.004128	0.003432
9	-1930	0.072403	0.003281	0.00323	0.091416	0.003312	0.004632
10	-1920	0.075259	0.002924	0.002958	0.0936	0.003504	0.004224

Graphing columns B and E while using column A for the x-coordinates, we get the following:



However, if we graph only the 5' and 3' tags that come from the H3K4me2-dht-16h directory (columns F and G):



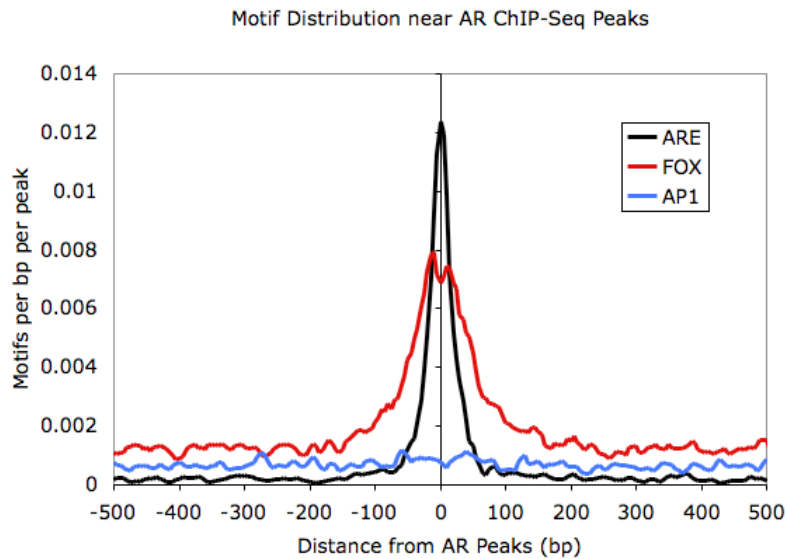
Here we can see how the 5' and 3' reads from the H3K4me2 marked nucleosomes are distributed near the AR sites.

Histograms of Motif Densities:

Making histograms out of motif occurrences is very similar to sequencing tag distributions. Run the `annotatePeaks.pl` program with `"-hist <#>"` and `"-m <motif file>"` (you can also find motif densities and tag densities at the same time):

```
annotatePeaks.pl ARpeaks.txt hg18 -size 1000 -hist 5 -m are.motif fox.motif ap1.motif > outputfile.txt
```

Graphing `outputfile.txt` with EXCEL:



Centering Peaks on Motifs

One cool analysis strategy is to center peaks on a specific motif. For example, by centering peak for the Androgen Receptor Transcription Factor on the ARE motif (GNACANNNTGTNC), you can map the spacial relationship between the motif and other sequence features and sequencing reads.

To center peaks on a motif, run **annotatePeaks.pl** with the following options:

```
annotatePeaks.pl <peak file> <genome> -size <#> -center <motif file> > newpeakfile.txt
```

i.e. **annotatePeaks.pl** ARpeaks.txt hg18r -size 200 -center are.motif > areCenteredPeaks.txt

Now the idea is to use the new peak file to perform analysis:

```
annotatePeaks.pl areCenteredPeaks.txt hg18 -size 6000 -hist 10 -d H3K4me2-notx/ ... > output.txt
```

Creating Heatmaps from High-throughput Sequencing data

HOMER is not capable of generating actual Heatmaps *per se*, but it will generate the data matrix (similar to a gene expression matrix) than can then be visualized using standard gene expression heatmap tools. For example, I will generate a heatmap data matrix file using HOMER, and then open it with [Cluster 3.0 \(Micheal Eisen/de Hoon\)](#) to cluster it and/or visualize it with [Java Tree View \(by Alok J. Saldanha\)](#). In reality, you can use any clustering and/or heatmap visualization software (i.e. R).

Basic usage (add **"-ghist"** when making a histogram):

```
annotatePeaks.pl <peak file> <genome> -size <#> -hist <#> -ghist -d <tag directory 1> [tag directory2] ... > <output matrix file>
```

i.e. **annotatePeaks.pl** ARpeaks.txt hg18 -size 6000 -hist 25 -ghist -d H3K4me2-control/ H3K4me2-dht-16h/ > outputfile.txt

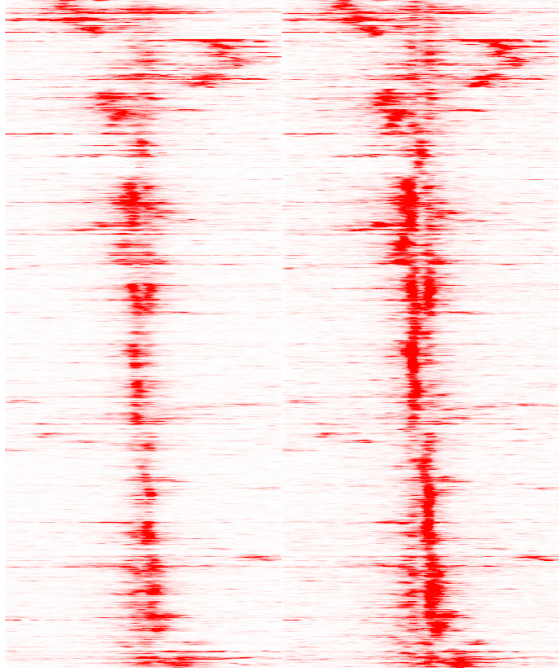
Running this command is very similar to making histograms with **annotatePeaks.pl**. In fact, a heatmap isn't really all that different from a histogram - basically, instead of averaging all of the data from each peak, we keep data from each peak separate and visualize it all together in a heatmap. The key difference when making a heat map or a histogram is that you must add **"-ghist"** when making a heatmap.

Format of Data Matrix Output File

The resulting file is a tab-delimited text file where the first row is a header file and the remaining rows represent each peak from the input peak file. The first set of columns will describe the read densities from the **first tag directory** as a function of distance from the center of the peak, with bin sizes corresponding to the parameter used with **"-hist <#>"**. After the first block of columns, a second block will start over with read densities from the **2nd tag directory**, and so on. If you would like to cluster this file to help organize the patterns, make sure you only cluster the "genes" (i.e. rows).

Example: **annotatePeaks.pl** ARpeaks.txt hg18 -size 6000 -hist 25 -ghist -d H3K4me2-control/ H3K4me2-dht-16h/ > outputfile.txt

After creating the file, I loaded into Cluster 3.0 to cluster and clustered the genes using "centered correlation" as the distance metric, then loaded that output into Java Tree View.



General Options to Control Data Analysis Behavior

- strand** <+|-|both> (only look for motif etc. on + or - strand, default both)
- fragLength** <#> (Fragment length, default=auto, might want to set to 0 for RNA)
- pc** <#> (maximum number of tags to count per bp, default=0 [no maximum])
- cons** (Retrieve conservation information for peaks/sites - creates new column for this information)
- CpG** (Calculate CpG/GC content)
- norevopp** (do not search for motifs on the opposite strand [works with -center too])
- norm** <#> (normalize tags to this tag count, default=1e7, 0=average tag count in all directories)
Use **-noadj** to disable tag normalization for sequencing depth

Command line options for annotatePeaks.pl

Usage: `annotatePeaks.pl <peak file | tss> <genome version> [additional options...]`

Available Genomes (required argument): (name,org,directory,default promoter set)

User defined annotation files (default is UCSC refGene annotation):

`annotatePeaks.pl` accepts GTF (gene transfer formatted) files to annotate positions relative to custom annotations, such as those from de novo transcript discovery or Gencode.

`-gtf <gtf format file>`

Peak vs. tss / tts mode (works with custom GTF file):

If the first argument is "tss" (i.e. `annotatePeaks.pl tss hg18 ...`) then a TSS centric analysis will be carried out. Tag counts and motifs will be found relative to the TSS. (no position file needed) ["tts" now works too - e.g. 3' end of gene]

NOTE: The default TSS peak size is 4000 bp, i.e. +/- 2kb (change with `-size` option)

`-list <gene id list>` (subset of genes to perform analysis [unigene, gene id, accession, probe, etc.], default = all promoters)

Primary Annotation Options:

- m** <motif file 1> [motif file 2] ... (list of motifs to find in peaks)
- m**score (reports the highest log-odds score within the peak)
- n**motifs (reports the number of motifs per peak)
- m**dist (reports distance to closest motif)
- m**fasta <filename> (reports sites in a fasta file - for building new motifs)
- f**m <motif file 1> [motif file 2] (list of motifs to filter from above)
- r**mrevopp <#> (only count sites found within <#> on both strands once, i.e. palindromic)
- m**atrix <filename> (outputs a motif co-occurrence matrix)
- m**bed <filename> (Output motif positions to a BED file to load at UCSC (or `-mpeak`))

- d <tag directory 1> [tag directory 2] ... (list of experiment directories to show tag counts for) NOTE: -dfile <file> where file is a list of directories in first column
- p <peak file> [peak file 2] ... (to find nearest peaks)
 - pdist to report only distance (-pdist2 gives directional distance)
 - pcount to report number of peaks within region
- gene <data file> ... (Adds additional data to result based on the closest gene. This is useful for adding gene expression data. The file must have a header, and the first column must be a GeneID, Accession number, etc. If the peak cannot be mapped to data in the file then the entry will be left empty.
- go <output directory> (perform GO analysis using genes near peaks)
- genomeOntology <output directory> (perform genomeOntology analysis on peaks)
- gsiz <#> (Genome size for genomeOntology analysis, default: 2e9)

Annotation vs. Histogram mode:

- hist <bin size in bp> (i.e 1, 2, 5, 10, 20, 50, 100 etc.)
- The -hist option can be used to generate histograms of position dependent features relative to the center of peaks. This is primarily meant to be used with -d and -m options to map distribution of motifs and ChIP-Seq tags. For ChIP-Seq peaks for a Transcription factor you might want to use the -center option (below) to center peaks on the known motif
- ** If using "-size given", histogram will be scaled to each region (i.e. 0-100%), with the -hist parameter being the number of bins to divide each region into.
- Histogram Mode specific Options:
- nuc (calculated mononucleotide frequencies at each position, Will report by default if extracting sequence for other purposes like motifs)
 - di (calculated dinucleotide frequencies at each position)
 - histNorm <#> (normalize the total tag count for each region to 1, where <#> is the minimum tag total per region - use to avoid tag spikes from low coverage)
 - ghist (outputs profiles for each gene, for peak shape clustering)
 - rm <#> (remove occurrences of same motif that occur within # bp)

Peak Centering: (other options are ignored)

- center <motif file> (This will re-center peaks on the specified motif, or remove peak if there is no motif in the peak. ONLY recentering will be performed, and all other options will be ignored. This will output a new peak file that can then be reanalyzed to reveal fine-grain structure in peaks (It is advised to use -size < 200) with this to keep peaks from moving too far (-mirror flips the position))
- multi (returns genomic positions of all sites instead of just the closest to center)

Advanced Options:

- len <#> / -fragLength <#> (Fragment length, default=auto, might want to set to 0 for RNA)
- size <#> (Peak size[from center of peak], default=inferred from peak file)
 - size #,# (i.e. -size -10,50 count tags from -10 bp to +50 bp from center)
 - size "given" (count tags etc. using the actual regions - for variable length regions)
- log (output tag counts as log2(x+1+rand) values - for scatter plots)
- sqrt (output tag counts as sqrt(x+rand) values - for scatter plots)
- strand <+|-|both> (Count tags on specific strands relative to peak, default: both)
- pc <#> (maximum number of tags to count per bp, default=0 [no maximum])
- cons (Retrieve conservation information for peaks/sites)
- CpG (Calculate CpG/GC content)
- ratio (process tag values as ratios - i.e. chip-seq, or mCpG/CpG)
- norevopp (do not search for motifs on the opposite strand [works with -center too])
- noadj (do not adjust the tag counts based on total tags sequenced)
- norm <#> (normalize tags to this tag count, default=1e7, 0=average tag count in all directories)
- pdist (only report distance to nearest peak using -p, not peak name)
- noann, -nogene (skip genome annotation step, skip TSS annotation)
- homer1/-homer2 (by default, the new version of homer [-homer2] is used for finding motifs)



Can't figure something out? Questions, comments, concerns, or other feedback:
cbenner@ucsd.edu



HOMER

Software for motif discovery and next-gen sequencing analysis

Quantifying RNA with analyzeRNA.pl

Homer contains a program (**analyzeRNA.pl**) to quantify RNA reads in genes. Of all the tools in HOMER, this one is probably the least efficient and resource hungry program in terms of memory. It needs to be rewritten but I have not found the time or pressing need to do this yet. However, it does perform useful calculations, particularly for GRO-Seq applications, that are not available or not automated for other programs.

analyzeRNA.pl DOES produces a gene expression matrix from "Tag Directories", and works in a similar manner to how annotatedPeaks.pl works for **ChIP-Seq** data. It has options to count reads in "genic" vs. "exon" regions, and attempts to automate the analysis of promoter proximal pausing seen in GRO-Seq data.

analyzeRNA.pl DOES NOT produce differential expression or differential splicing calls. It would be great if Chuck had the time to help me with that, but for now I recommend sending the output from analyzeRNA.pl to other utilities such as [edgeR](#) or [DEseq](#) or whatever your favorite next-gen differential signal finder is.

Basic usage:

```
analyzeRNA.pl <rna|repeats|gtf file> <genome> [options] -count [genes|exons|introns] -d <Tag Directory> [Tag Directory 2] ... > <output file>
```

i.e. `analyzeRNA.pl rna hg18 -count genes -d IMR90-GroSeq > outputfile.txt`

Specifying the Genes/Transcripts to Analyze

The first argument to analyzeRNA.pl specifies which "gene definition" to use for analysis. There are three options:

1. **rna** - This will direct HOMER to analyze the default RefSeq annotation for that genome.
2. **repeats** - HOMER will load repeat definitions from UCSC and assess the expression levels of different repeat classes. This can be resource hungry and is not recommended unless your computer has a bunch of memory.
3. **<GTF file>** - Specify your own custom genes. These can be an alternative annotation (i.e. Ensembl genes, UCSC genes) or it can be custom, de novo genes you found analyzing GRO-Seq or mRNAs defined using "cufflinks". If you are looking for a GTF file for your favorite annotation, check out the [UCSC Table Browser](#). You can use this tool to download GTF formatted files (with the correct genome version) for many popular annotations.

Examples include "analyzeRNA.pl rna mm9 ..." or "analyzeRNA.pl myGenes.gtf mm9 ...".

Choosing Experiments to Analyze

As with all HOMER programs, you need to create "Tag Directories" out of each experiment you want to quantify first. To quantify them with **analyzeRNA.pl** simply add one or more directory after "-d". For example:

```
analyzeRNA.pl rna hg18 -d LNCaP-RNA-Seq-notx/ LNCaP-RNA-Seq-Dht/ > outputfile.txt
```

This will produce an output file containing genes expression values for two experiments, "LNCaP-RNA-Seq-notx" and "LNCaP-RNA-Seq-Dht".

Measuring Gene Expression in Exons vs. Gene Bodies.

Depending on the type of sequencing you are analyzing, you will want to quantify RNA from different parts of the gene. The "-count [...]" option controls which regions of the gene are used for analysis (use like **"-count exons"** or **"-count genes"**)

- **exons** (default) Counts tags in exons only. Use this for most applications of RNA-Seq, such as polyA-RNA-seq or other techniques that aim to measure mRNA.
- **cds** - Counts tags in coding regions only. This could be useful for quantifying ribosome coverage on coding sequences with techniques such as Ribo-Seq
- **introns** - Counts tags on introns only.
- **5utr** or **3utr** - Count tags on 5' UTR and 3' UTR regions, respectively
- **genes** - Counts tags on the full gene body (TSS to TTS). This is useful for GRO-Seq where we expect coverage across the entire transcript. Can also be used to quantify H3K36me3 or PolII ChIP-Seq.

By default, analyzeRNA.pl assumes your RNA is strand-specific. If it is not, or you are using analyzeRNA.pl for other types of data such as ChIP-Seq, you can specify:

- "-strand +"** - default, only measure + strand (relative to gene orientation), for strand specific RNA
- "-strand -"** - only measure - strand (relative to gene orientation)
- "-strand both"** - count reads on both strands, for non-strand specific RNA or ChIP-Seq

Also, there are a couple options to slightly modify how tags are counted to avoid TSS/TTS features. By specifying **"-start <#>"**, you can adjust where (relative to the TSS) HOMER starts counting reads. This is useful for GRO-Seq when you might want to avoid the promoter-proximal paused region by add **"-start 500"** to start counting 500 bp downstream from the TSS. Negative values would start counting 500 bp upstream. The option **"-end <#>"** is similar, but applies to the TTS. Negative values will stop the tag counting upstream of the TTS (**"-end -500"**), while positive values will count tags further downstream.

Normalization of Gene Expression Values

Normalization is probably the trickiest part about RNA-Seq. There are lots of papers on it. HOMER offers a couple different normalization options depending on what the experiment is and what your needs are.

By default, HOMER normalizes each experiment to 10 million mapped reads, which is the same normalization strategy used in **annotatePeaks.pl** for ChIP-Seq data. However, RNA has the potential to contain much more "contaminates" than ChIP-Seq. In ChIP-Seq, the background is the genome, with random fragments coming down in the immunoprecipitation step. With RNA, instead of a two copies of the genome per cell, you have 99% ribosomal RNA, along with a host of other very very common short RNA species such as tRNAs, snoRNAs, and other problematic things, and then a small fraction of what you're actually interested in. Most RNA-Seq protocols contain enrichment steps, such as polyA selection, to isolate mRNAs from the rest of the crap (my appologies to those studying rRNA and tRNA). These enrichment steps can have different efficiencies from sample to sample, with some samples containing more rRNA, tRNA etc. than others. When mapping to the genome, many of these RNA species will be discarded if you only keep reads that map uniquely since the genomic elements that create them are often repeated in the genome. However, you're bound to still map many of these reads. Differential contamination of common RNA species can throw off the default normalization - if 50% of your "mapped" reads are rRNA in one sample and only 25% in another, you're likely going to have problems.

There are several things you can do to combat these problems. One is to "pre-clear" common RNA species computationally, i.e. map your reads to rRNA first to remove them, then map the rest to the genome. Another strategy is to normalize strictly with mRNA species only instead of considering the total number of mapped reads. In general, I would recommend normalizing in this strategy (**"-normMatrix <#>"**) - it's probably the safest in most situations due to the contamination problem. Below are the various normalization options in **analyzeRNA.pl**.

- **-norm <#>** - Normalize the total number of mapped reads per experiment to #, this is the default **"-norm 1e7"**.
- **-normMatrix <#>** - Normalize the total of number of reads found in the gene expression matrix to # (i.e. normalize total reads in mRNAs)
- **-noadj** - Don't perform any normalization, just report the exact number of reads found. This is useful when sending the output of HOMER to another differential expression program such as edgeR or similar that requires the raw read counts.

The options above perform the actual normalization. "Dress up" normalization/transformations options follow:

- **-rpkm** - Report normalized values as reads per kilobase per million mapped reads

- **-log** - Report log normalized values with jitter = $\log_2(x+1+\text{rand}(0-1))$
- **-sqrt** - Report square root of the normalized value with jitter = $\sqrt{x+\text{rand}(0-1)}$

Limiting the number of Tags per Position ("-pc <#>")

Another important parameter (that I almost made mandatory) is **"-pc <#>"**, which controls the maximum number of reads to consider per position. In the case of some genes, there may be high-abundance RNA species in introns, where large spikes in RNA reads may occur. If quantifying expression from gene bodies (i.e. **"-count genes"**, GRO-Seq), these reads will be counted toward the gene's total abundance. Ideally these are removed. However, as a quick and dirty solution to this problem, you can limit the number of reads that HOMER counts from each position. In many cases rRNA loci will have tens of thousands of reads per bp. On the flip side, limiting the number of reads per bp can kill your dynamic range. Usually the middle ground such as **"-pc 3"** is a good way to go - you effectively remove the super spikes but still allow genes with high RPKM to "express themselves" in your analysis.

How analyzeRNA.pl Works

The following are the steps HOMER takes to produce a Gene Expression Matrix:

1. Parses the RNA definition file to figure out where all of the genes are in the genome.
2. Queries each of the "Tag Directories" and determine where each of the tags are within gene bodies.
3. At that point it will rummage through the tag positions and count only the tags found within the desired regions (i.e. exons). Incidentally, this is also the part that makes it less efficient at the moment - it's done with perl and keeps way too much extra information around.
4. Expression values are calculated and normalized
5. Results are formatted and set to *stdout*

Description of Output Files

The output gene expression matrix with the following columns:

1. Transcript ID (RefSeq accession is the default with "rna" option, custom name is used otherwise)
2. chromosome
3. start
4. end
5. strand
6. mRNA length (e.g. exons only)
7. gene length (TSS to TTS)
8. Copies in genome (some genes map to the genome more than once...)
9. Symbol (e.g Gene Name)
10. Alias (alternative gene names)
11. Description
12. Unigene
13. Entrez Gene ID
14. Ensembl
15. Data: First Tag Directory Gene Expression information
16. [Data: Second Tag Directory Gene Expression information]
17. ...

A separate column is generated for each tag directory. The head of these columns contains the name, the region of the gene used for expression (i.e. gene, exon, 5utr..), the total number of mapped tags in the directory, and the normalization factor.

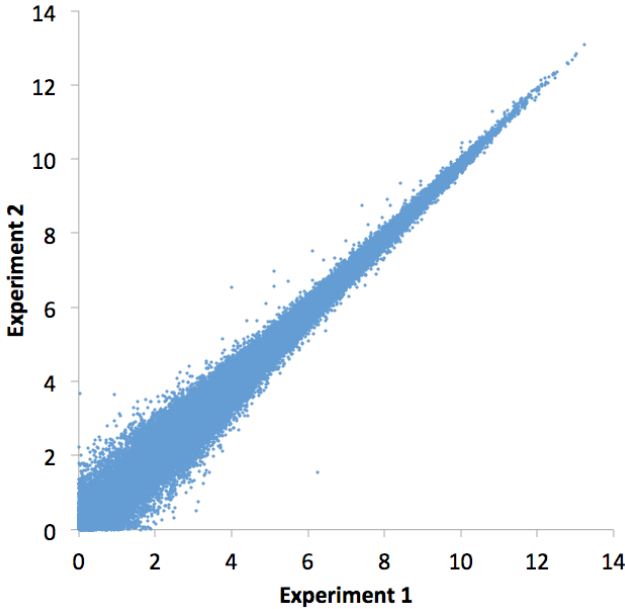
Due to the fact that some genes map to multiple places in the genome (<1% of RefSeq), only one of their locations will be reported. However, the gene length and mRNA lengths will be averaged, and their read totals will be averaged in the output.

If custom IDs are used, HOMER will attempt to link them to known gene identifiers (As of now it does not annotate their position, just treats their ID as an accession number and tries to match it with known genes). If you give analyzeRNA.pl a custom GTF file from Cufflinks, where the IDs are all "CUFF12345.1", most of the annotation columns will be blank.

Below is an example of the output opened with EXCEL:

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P
1	Genes	chr	start	end	stran	mRNA Length	Gene Length	Copies	in Symbol	Alias	Description	Unigene	GeneID	Ensembl	Macrophage-RNA-nc	Macrophage-RN
2	NM_001001130	chr13	67848736	67856071	-	2218	7335	1	Zfp85-rs1	KRAB19	Rslc zinc finger pr Mm.288396	22746	ENSMUSG		25.49128477	26.38801643
3	NM_001001144	chr9	110235796	110287450	+	4286	51654	1	Scap	9530044G19	SREBF chape Mm.288741	235623	ENSMUSG		82.8466755	160.357946
4	NM_001001152	chr13	67355853	67370004	-	3488	14151	1	Zfp458	BC062958	R zinc finger pr Mm.306358	238690	ENSMUSG		11.15243709	12.85570031
5	NM_001001160	chr6	85419571	85452880	-	6562	33309	1	Fbxo41	9630017H13	F-box proteir Mm.38777	330369	ENSMUSG		0	0
6	NM_001001176	chrX	103402212	103415181	-	2583	12969	1	Taf9b	BC066223	Ti TAF9B RNA p Mm.19440	407786	ENSMUSG		0	0
7	NM_001001177	chr17	34535764	34597679	+	1900	61915	1	BC051142	NG8	TSBP T cDNA sequer Mm.73205	407788	ENSMUSG		0.796602649	0
8	NM_001001178	chr2	58674108	58998684	-	3920	324576	1	Ccdc148	5830402J09	coiled-coil dc Mm.432280	227933	ENSMUSG		0	0
9	NM_001001179	chr6	128489839	128531624	-	4698	41785	1	BC048546	MGC58520	c cDNA sequer Mm.259234	232400	ENSMUSG		0	0
10	NM_001001180	chr7	147995575	148008077	-	3909	12502	1	Zfp941	-	zinc finger pr Mm.359154	407812	ENSMUSG		0	0.676615806
11	NM_001001181	chr18	75165553	75169587	+	1047	4034	1	BC031181	-	cDNA sequer Mm.29866	407819	ENSMUSG		157.7273245	150.2087089
12	NM_001001182	chr2	59737419	59963797	-	7980	226378	1	Baz2b	5830435C13	bromodomai Mm.436730	407823	ENSMUSG		215.8793179	158.3280986
13	NM_001001183	chr17	25194646	25218059	-	1779	23413	1	Tmem204	-	transmembri Mm.34379	407831	ENSMUSG		1.593205298	0.676615806
14	NM_001001184	chr8	47660947	47702554	+	3797	41607	1	Ccdc111	BC065112	N coiled-coil dc Mm.217385	408022	ENSMUSG		145.7782848	63.60188576
15	NM_001001185	chr13	67964273	67964854	+	581	581	1	BC048507	-	cDNA sequer Mm.177840	408058	ENSMUSG		24.69468212	33.8307903
16	NM_001001186	chr13	67464519	67476699	-	4079	12180	1	Zfp456	BC065418	N zinc finger pr Mm.461583	408065	ENSMUSG		47.79615894	33.15417449
17	NM_001001187	chr13	67768377	67784449	-	4504	16072	1	Zfp738	3830402I07R	zinc finger pr Mm.435551	408068	ENSMUSG		218.2691258	174.5668779
18	NM_001001189	chr9	64154562	64189064	-	3575	34502	1	Dis3l	AV340375	DIS3 mitotic Mm.268341	213550	ENSMUSG		45.40635099	64.27850156
19	NM_001001297	chr2	21127350	21136636	+	2327	9286	1	Thns1	AW413632	threonine syi Mm.268841	208967	ENSMUSG		23.10147682	12.17908451
20	NM_001001309	chr2	12028286	12223547	-	5782	195261	1	Itga8	AI447669	integrin alph. Mm.329997	241226	ENSMUSG		0	0
21	NM_001001319	chr4	143649028	143659221	+	2399	10193	1	Pramel4	C79430	D4E1 preferentialh Mm.271697	347710	ENSMUSG		0	0
22	NM_001001320	chr19	3992751	3999512	+	1778	6761	1	Tbx10	Dc MGC129	T-box 10 Mm.246555	109575	ENSMUSG		0	0
23	NM_001001321	chr13	64197617	64230638	-	2276	33021	1	Slc35d2	5730408I21R	solute carrieri Mm.133731	70484	ENSMUSG		42.2199404	37.21386932
24	NM_001001322	chr2	26828935	26865145	+	4580	36210	1	Adamts13	Gm710	WVF a disintegrin Mm.330084	279028	ENSMUSG		0	1.353231612
25	NM_001001326	chr7	116667424	116760661	-	4255	93237	1	St5	2010004M01	suppression Mm.252009	76954	ENSMUSG		140.9986689	59.54219092

If you specify the **"-log"** or **"-sqrt"** options, you can make a nice X-Y scatter plot of the data columns. Here is an example:



Quantifying Promoter-proximal RNA Polymerase Pausing

One of the neat observations from GRO-Seq experiments is that an actively engaged RNA Polymerases are present at the promoters of many genes, but they do not seem to elongate down the body of the gene. The idea is that they are "paused", waiting for elongation signals to give them the green light to make the gene.



HOMER can calculate the "pausing" ratio, which is the ratio of tag density at the promoter relative to the body of the gene. To use this option, specify **"-pausing <#>"**, where # the is the distance downstream of the TSS to stop counting promoter tags and start counting gene body tags. A safe value is probably 250-500.

This will que the program to produce 3 columns of output per experiment. First, it will report the pausing ratio, and then it will report the promoter and gene body read densities.

Analyzing Repeat and non-mRNA Expression

mRNA is not the only RNA species present in the cell. In fact, it makes up a very small fraction of the total. **analyzeRNA.pl** offers an option to help quantify repeat RNAs, rRNA, etc. By specifying "**repeats**", HOMER will load the definition of all repeats in the genome and quantify tags on them. Within the repeat definition are tRNAs, rRNAs, etc. so you get more than just transposable elements. Use it like so:

analyzeRNA.pl repeats mm9 -d Macrophage-RNAseq > outputfile.txt

By default, all the repeats are treated like "exons". Repeats from similar classes are combined to give a single expression value. Only one of the positions are reported (randomly). Be careful! It uses a lot of memory! If you're running out of memory, try using fewer tag directories (i.e. analyze one at a time).

To do repeat expression justice, you should carefully consider how the data is mapped to the genome. Normally for ChIP-Seq (or even RNA-Seq), you do not want to consider reads that map to multiple locations in the genome. However, in the case of RNA repeats, this means that you will be discarding many of the reads mapping to repeat regions. One trick that works fairly well is to keep one random position from the mapping, regardless if it maps uniquely to the genome (but only do it for this type of analysis). Typically, if a read maps to multiple locations, those multiple locations are probably all the same type of repeat element, so it will be added to the expression of that repeat class regardless of where it is specifically placed. This is "approximate", so use with care.

The thing to note is that repeat/rRNA/tRNA expression is compounded by the fact that most protocols try to get rid of it, so depending on the efficiency of these clearing steps experiment-to-experiment, you may be measuring the difference in clearing efficiency rather than the actual expression of the RNA.

Command line options for analyzeRNA.pl

Usage: analyzeRNA.pl <rna | repeats | custom RNA/GTF file> <genome version> [additional options...]

Program for quantifying RNA tag counts. The first argument can be "rna" (refseq genes), "repeats" (repeat classes), or a custom RNA definition file. (see website for format)

!!! Right now "repeats" is not memory efficient, need to optimize - i.e. 20Gb !!!

!!! Only run "repeats" with a single tag directory for now !!!

Available Genomes (required argument): (name,org,directory,default promoter set)

Primary Annotation Options:

- d <tag directory 1> [tag directory 2] ... (list of experiment directories to show tag counts for) NOTE: -dfile <file> where file is a list of directories in first column
- rpkm (Report results as reads per kb per million mapped)
- norm <#> (Normalize to total mapped tags: default 1e7)
- normMatrix <#> (Normalize to total tags in gene expression matrix: not used)
- noadj (Don't normalize)
- count <exons|introns|genes|5utr|3utr|cds> (Count tags in introns, exons, etc., default: exons)
- noCondensing (do not condense counts from entries with same ID, default: do condense)
- pc <#> (maximum tags to count per position, default: 0=no limit)
- strand <+|-|both> (count tags on indicated strand, default: +)
- gene <data file> ... (Adds additional data to result based on the closest gene. This is useful for adding gene expression data. The file must have a header, and the first column must be a GeneID, Accession number, etc. If the peak cannot be mapped to data in the file then the entry will be left empty.)
- log (output tag counts as randomized log2 values - for scatter plots)
- sqrt (output tag counts as randomized sqrt values - for scatter plots)
- start <#> (start counting tags relative # offset of beginning of gene)
- end <#> (finish counting tags relative # offset to end of the gene)
- pausing <#> (calculate ratio of pausing first [# bp of transcript] to gene body) Produces 3 columns - promoter rpk, body rpk, and ratio (add -log for log versions) Also sets "-count genes". Use "-strand both" when analyzing Pol II ChIP-Seq

rpk is reads per kb - set `-norm 1e6` or `-normMatrix 1e6` to get rpk



Can't figure something out? Questions, comments, concerns, or other feedback:
cbenner@ucsd.edu



HOMER

Software for motif discovery and ChIP-Seq analysis

Finding Overlapping and Differentially Bound Peaks

HOMER provides a utility for comparing sets of peaks called **mergePeaks**. It's default behavior is to take two or more peak files and return a single peak file containing the unique peak positions from the original files. For example:

```
mergePeaks -d <maximum distance to merge> <peak file1> <peak file2>  
[peak file3] ... > newPeakFile.txt
```

The program will output a new peak file containing the merged peaks to *stdout*. Peaks within the distance in bp specified by "**-d <#>**" will be reported as the average position between the peaks found within the common region (default=100 bp, good for transcription factors). The origin of the peaks is specified in the 7th column of the new peak file. Alternatively you can specify "**-d given**" to require a specific overlap between the start and end coordinates of the peaks. This is more useful if comparing large regions as opposed to peaks. The program will also output the numbers for creating a venn diagram, and these can be directed to a specific file by specifying "**-venn <filename>**".

Separating Peaks into Unique and Overlapping sets

Merging peaks together into a single file is very useful for certain types of analysis, such as making scatter plots that compare the tag-densities between peaks from separate experiments - in this case you want to count tags at specific and common regions. Alternatively, you may be interested in separating the peaks into common and specific sets for focused analysis. To do this use the "**-prefix <filename>**" option - this will create separate files based on overlapping peaks for each set of peaks. For example:

```
mergePeaks -d 100 pu1.peaks cebp.peaks -prefix mmm
```

This will create files named "**mmm_pu1.peaks**", "**mmm_cebp.peaks**", and "**mmm_pu1.peaks_cebp.peaks**".

The output file will contain the following columns:

1. Merged Peak name (will start with "Merged-")
2. chromosome
3. start (average from merged peaks)
4. end (average from merged peaks)

5. strand
6. Average peak score (actually, the average of the original values in column 6 of the peak files - or column 5 of BED files)
7. Original peak files contributing to the merged peak
8. Total number of peaks merged (occasionally more than one peak from a single file will be merged if the peaks are within the specify distance or two or more peaks from one file overlap with the same single peak(s) from another file)

Peak Co-Occurrence Statistics

The **mergePeaks** program will also find calculate the statistics of co-occurrence between peaks in a pairwise fashion. If "**-matrix <filename>**" is specified, HOMER will calculate statistics about the pairwise overlap of peaks. Three separate pairwise matrix files will be produced using the supplied <filename> as a prefix:

filename.logPvalue.matrix.txt (natural log p-values for overlap using the hypergeometric distribution, positive values signify divergence)

filename.logRatio.matrix.txt (natural log of the ratio of observed overlapping peaks to the expected number of overlapping peaks)

filename.count.matrix.txt (raw counts of overlapping peaks)

The statistics are dependent on the effective size of the genome, which can be specified using "**-gsize <#>**" (default: 2,000,000,000)

Co-Bound Peaks

Sometimes you just want to know how many other peaks bind a set of reference peaks. If "**-cobound <#>**", mergePeaks counts how many of the other peak files contain overlapping peaks with the peaks found in the first peak file. It then outputs peak files named "coboundBy0.txt", "coboundBy1.txt", etc. up to the number specified.

Differentially Bound Peaks

To find peaks that are differentially enriched between two experiments, there are two basic options. First, you could run **findPeaks** ([info here](#)) using the 2nd experiment as the control sample. Alternatively, you can use **getDifferentialPeaks**, which will take a given list of peaks and quickly identify which peaks contain significantly more tags in the target experiment relative to the background experiment. To use it, follow this syntax:

**getDifferentialPeaks <peak/BED file> <target Tag directory>
<background Tag directory> [options]**

By default it looks for peaks that have 4-fold more tags (sequencing-depth

independent) and a cumulative Poisson p-value less than 0.0001 (sequencing-depth dependent). These parameters are adjustable with ("**-F <#>**", and "**-P <#>**"). By specifying "**-same**", peaks that are similar between the two tag directories will be returned instead of differential peaks. One caveat is that it is a good idea to set the size of the region used to search for reads to be larger than the actual peaks (i.e. +100 bp relative to the peak size) to avoid problems that arise from experiments with different fragment lengths, etc.

Command Line options for mergePeaks

Usage: mergePeaks [options] <primary peak file> [additional peak/annotation files...]

Merges and/or compares peak/position files (peak files listed twice are only considered once)

General Options:

-strand (Only merge/consider peaks on the same strand, default: either strand)
 -d <#|given> (Maximum distance between peak centers to merge, default: 100)
 Using "-d given" looks for literal overlaps in peak regions
 Use "-d given" when features have vastly different sizes (i.e. peaks vs. introns)
 -file <filename> (file listing peak files to compare - for lots of peak files)
 -gsize <#> (Genome size for significance calculations, default: 2e9)

Merging Peaks Options (default):

-prefix <filename> (Generates separate files for overlapping and unique peaks)
 By default all peaks are sent to stdout
 -matrix <filename> (Generates files with pairwise comparison statistics)
 filename.logPvalue.matrix.txt - ln p-values for overlap, +values for divergence
 filename.logRatio.matrix.txt - ln ratio of observed/expected overlaps
 filename.count.matrix.txt - peak overlap counts
 -venn <filename> (output venn diagram numbers to file, default: to stderr)
 -code (report peak membership as binary instead of by file names)

Classify peaks by how many are co-bound by other peak files vs. reference(1st file)

-cobound <#> (Maximum number of co-bound peaks to consider)
 Will output sets of peaks that are co-bound by various numbers of factors
 to files coBoundBy0.txt, coBoundBy1.txt, coboundBy2.txt, ...
 Or <prefix>.coBoundBy0.txt, <prefix>.coBoundBy1.txt, ...
 -matrix <filename> (generates similar files to above with pairwise overlap statistics)

Single peak file:

(If a single peak file is given, peaks within the maximum distance will be merged)
-filter chrN:XXX-YYY (only analyze peaks within range)
-coverage <output file> (returns the total bp covered by each peak file - use "-d given")

Command Line options for getDifferentialPeaks

Usage: getDifferentialPeaks <peak file> <target tag directory> <background tag directory> [options]

Extracts tags near each peak from the tag directories and counts them, outputting peaks with significantly different tag densities

General Options:

-F <#> (fold enrichment over background tag count, default: 4.0)
-P <#> (poisson enrichment p-value over background tag count, default: 0.0001)
-same (return similar peaks instead of different peaks)
-rev (return peaks with higher tag counts in background instead of target library)
-size <#> (size of region around peak to count tags, default: -fixed)
-fixed (Count tags relative to actual peak start and stop, default)

Output Options:

-strand <both|+|-> (Strand [relative to peak] to count tags from, default: both)
-tagAdjust <#> (bp to shift tag positions to estimate fragment centers, default: auto)
'-tagAdjust auto' uses half of the estimated tag fragment length
-tagAdjustBg <#> (bp to shift background tag positions to estimate fragment centers, default: auto)
'-tagAdjustBg auto' uses half of the estimated tag fragment length
-tbp <#> (Maximum tags per bp to count, 0 = no limit, default: 0)
-tbpBg <#> (Maximum background tags per bp to count, 0 = no limit, default: 0)



Can't figure something out? Questions, comments, concerns, or other feedback:
cbenner@ucsd.edu



HOMER

Software for motif discovery and next-gen sequencing analysis

homerTools - General sequence manipulation

homerTools is a utility program Chuck uses for basic sequence manipulation of FASTQ files, extracting sequences from genome FASTA files, and calculating nucleotide frequencies. To run **homerTools** do the following:

homerTools [command] [command specific options]

i.e. **homerTools trim -3 AAAAAAAA s_1_sequence.txt**

The following commands are available in **homerTools**:

barcodes - for separating and removing 5' barcodes from FASTQ/FASTA files

trim - for trimming by adapter sequence, specific lengths, etc. from FASTQ/FASTA files

freq - for calculating nucleotide frequencies in FASTQ/FASTA/txt sequence files

extract - for extracting specific regions of sequence from genomic FASTA files

Separating 5' Barcodes:

To separate and remove 5' barcodes from sequencing data (where the first "x" base pairs of the read are the barcode):

homerTools barcodes <# length of barcode> [options] <sequence file1> [sequence file2] ...

i.e. **homerTools barcodes 3 s_1_sequence.txt** (removes first 3bp as the barcode and sorts the reads by barcode)

The 3rd argument must be the length of the 5' barcode, which will be the first base pairs in the sequence. By default, this command creates files named "filename.barcode", such as s_1_sequence.txt.AAA, s_1_sequence.txt.AAC, s_1_sequence.txt.AAG etc. The parameter "**-min <#>**" specifies the minimum barcode frequency to keep (default is 0.02 [2%]). The frequency of each barcode is recorded in the output file "filename.freq.txt". If important barcodes were deleted, rerun the command with a smaller value for "**-min <#>**".

Trimming Sequence Files

With all the fancy types of sequencing being done, it is getting common to find adapters as part of the sequences that are analyzed. The trim command allows users to trim sequences from the 3' and 5' ends by either a specific number of nucleotides or remove a specific adapter sequence. The basic command is executed like this:

homerTools trim [options] <sequence file1> [sequence file2]

The output will be placed in files "filename.trimmed" and the distribution of sequence lengths after trimming will be in "filename.lengths" for each of the input files. The following options control how homerTools trims the sequences:

- len <#>** (trim sequences to this length)
- min <#>** (remove sequence that are shorter than this after trimming)
- 3 <#>** (trim this many bp off the 3' end of the sequence)
- 5 <#>** (trim this many bp off the 5' end of the sequence)
- 3 <ACGT>** (trim adapter sequence (i.e. "-3 GGAGGATTT") from the 3' end of the sequence)
- 5 <ACGT>** (trim adapter sequence (i.e. "-5 GGAGGATTT") from the 5' end of the sequence)

For adapter sequence trimming, it will search for the first full match to the sequence and delete the rest of the sequence. For example if you specify "-3 AA", it will search for the first instance of "AA" and delete everything after it. It will also delete partial matches if they are at the end of the sequence (or beginning for 5'). As another example, our lab uses an amplification strategy for RNA that results in the ligation of a polyA tail to the RNA sequence. If the reads are long enough, the read will be just As.

i.e. GAGATTATCTACGTACCGAAAAAAAAAAAAAAAAAAAAA

Trimming with "-3 AAAAAAAAAA" will cleave the complete polyA stretch.

In this example: GAGATTATCTACGTACCGTACTGCATGACGGGAAAA, only the final 4 As would be trimmed.

Extracting Genomic Sequences From FASTA Files

The **extract** command can be used to extract large numbers of specific genomic sequence. The first input file you need is a HOMER style peak file or a BED file with genomic locations. Next, you must have the genomic DNA sequences in one of two formats: (1) a directory of chr1.fa, chr2.fa FASTA files (can be masked file like *.fa.masked), or (2) a single file FASTA file with all of the chromosomes concatenated in one file. The sequences are sent to *stdout* as a tab-delimited file, or as a FASTA formatted file if "-fa" is added to the end of the command. Save the output to a file by adding "> outputfile.txt" to the end of the command. The program is run like this:

homerTools extract <peak/BED file> <FASTA directory or file location> [-fa]

```
i.e. homerTools extract peaks.bed
/home/chucknorris/homer/data/genomes/mm9/ > outputSequences.txt
Or, to get FASTA files back, i.e. homerTools extract peaks.bed
/home/chucknorris/homer/data/genomes/mm9/ -fa >
outputSequence.fa
```

Calculating Nucleotide Frequencies

The **freq** command will calculate nucleotide frequencies from FASTQ, FASTA, or tab-delimited text sequence files. The program tries to auto detect the format, but it may help to specify the format directly ("**-format fastq**", "**-format fasta**", "**-format tsv**"). The program outputs a position-dependent nucleotide/dinucleotide frequency file as a function of the distance from the start of the sequencing reads. The output is sent to *stdout*, unless you specify "**-o <outputfile.txt>**". If you specify "**-gc <outputfile2.txt>**", the program will also create a file that specifies the cumulative frequency of CpG, total G+C, total A+G, and total A+C in each individual sequence.

```
homerTools freq -format fastq s_1_sequence.txt > s_1.frequency.txt
homerTools freq -format fastq s_1_sequence.txt -gc
GCdistribution.txt -o positionFrequency.txt
```

homerTools Command Line options:

Usage: homerTools <command> [--help | options]

Collection of tools for sequence manipulation

Commands: [type "homerTools <command>" to see individual command options]

- barcodes - separate FASTQ file by barcodes
- trim - trim adapter sequences or fixed sizes from FASTQ files(also splits)
- freq - calculate position-dependent nucleotide/dinucleotide frequencies
- extract - extract specific sequences from FASTA file(s)
- decontaminate - remove bad tags from a contaminated tag directory
- cluster - hierarchical clustering of a NxN distance matrix
- special - specialized routines (i.e. only really useful for chuck)

Options for command: barcode

- min <#> (Minimum frequency of barcodes to keep: default=0.020)
- freq <filename> (output file for barcode frequencies, default=file.freq.txt)
- qual <#> (Minimum quality score for barcode nucleotides, default=not used)
- qualBase <character> (Minimum quality character in FASTQ file, default=B)

Options for command: trim

- 3 <#>|[ACGT]> (trim # bp or adapter sequence from 3' end of sequences)
- 5 <#>|[ACGT]> (trim # bp or adapter sequence from 5' end of sequences)
- mis <#> (Maximum allowed mismatches in adapter sequence, default: 0)
- minMatchLength <#> (minimum adapter sequence at edge to match, default: half adapter length)
- len <#> (Keep first # bp of sequence - i.e. make them the same length)
- stats <filename> (Output trimming statistics to filename, default: sent to stdout)
- min <#> (Minimum size of trimmed sequence to keep, default: 1)
- max <#> (Maximum read length, default: 100000)
- suffix <filename suffix> (output is sent to InuptFileName.suffix, default: trimmed)
- lenSuffix <filename suffix> (length distribution is sent to InuptFileName.suffix, default: lengths)
- split <#> (Split reads into two reads at bp #, output to trimmed1 and trimmed2)
- revopp <#> (Return reverse opposite of read [if used with -split, only the 2nd half of the read will be retuned as reverse opposite])

Options for command: freq

- format <tsv|fasta|fastq> (sequence file format, default: auto detect)
- offset <#> (offset of first base in output file, default: 0)
- maxlen <#> (Maximum length of sequences to consider, default: length of 1st seq)
- o <filename> (Output filename, default: output sent to stdout)
- gc <filename> (calculate CpG/GC content per sequence output to "filename")

OutputFormat: name<tab>CpG<tab>GC<tab>AG<tab>AC<tab>Length

Options for command: extract

- fa (output sequences in FASTA format - default is tab-delimited format)

Alternate Usage: homerTools extract stats <Directory of FASTA files>

Displays stats about the genome files (such as length)

Options for command: decontaminate

- frac <#> (Estimate fraction of sample that is contaminated, default: auto)
- estimateOnly (Only estimate the contamination, do not decontaminate)
- o <output tag directory> (default: overrides contaminated tag directory)
- size <#> (Peak size for estimating contamination/Max distance from contaminant reads to remove contaminated reads, default: 250)
- min <#> (Minimum tag count to consider when estimating contamination, default: 20)



Can't figure something out? Questions, comments, concerns, or other feedback:
cbenner@ucsd.edu



HOMER

Software for motif discovery and next-gen sequencing analysis

Miscellaneous Tools for Sharing Data

HOMER contains several utility programs for changing file formats and performing tasks that you may find useful from time to time. Below are a list of programs that may come in handy.

Outputing Tag Directory as a BED file (tagDir2bed.pl):

This is useful when you want to share your sequencing with others as BED format is about the most general format there is out there.

```
tagDir2bed.pl <tag directory> > output.alignment.file.bed
```

```
i.e. tagDir2bed.pl Macrophage-PU.1-ChipSeq/ > mac.pu1.bed
```

This will produce a large BED file that can be used to import the data to other programs.

Covertng between HOMER peak and BED file formats (pos2bed.pl / bed2pos.pl):

Want to load HOMER peaks into the genome browser? Or use them with other software?

Covert a HOMER peak/position file to a BED file:

```
pos2bed.pl <peak file> > output.bed
```

Covert a BED file to a HOMER peak/position file:

```
bed2pos.pl <BED file> > output.peakfile.txt
```



Can't figure something out? Questions, comments, concerns, or other feedback:
cbenner@ucsd.edu



HOMER

Software for motif discovery and next-gen sequencing analysis

Alignment of High-throughput Sequencing Data

Homer does not perform alignment - this is something that must be done before running homer. Several quality tools are available for alignment of short reads to large genomes. Check out [this link](#) for a list of programs that do short read alignment. BLAST, BLAT, and other traditional alignment programs, while great at what they do, are not practical for alignment of these types of data.

If you need help deciding on a program to use, I'll recommend [Bowtie](#) (it's nice and fast).

If you have a core that maps your data for you, don't worry about this step. However, in many cases there is public data available that hasn't been mapped to the genome or mapped to a different version of the genome or mapped with different parameters. In these cases it is nice to be able to map data yourself to keep a nice, consistent set of data for analysis.

Most types of ChIP-Seq/DNase-Seq/MNase-Seq and GRO-Seq simply need to be mapped to the genome, as they represent the sequencing of genomic DNA (or nascent RNA, which should not be spliced yet). If analyzing RNA-Seq, you may be throwing away interesting information about splicing if you simply align the data to the genome. If aligning RNA, I'd recommend sticking to the formal wear and trying [Tophat](#), which does a good job of identifying splice junctions in your data.

Which reference genome (version) should I map my reads to?

Both the organism and the exact *version* (i.e. hg18, hg19) are very important when mapping sequencing reads. Reads mapped to one version are NOT interchangeable with reads mapped to a different version. I would follow this recommendation list when choosing a genome (Obviously try to match species or sub species when selecting a genome):

1. Do you have a favorite genome in the lab that already has a bunch of experiments mapped to it? Use that one.
2. Do any of your collaborators have a favorite genome?
3. Use the latest stable release - I would recommend using genomes curated at UCSC so that you can easily visualize your data later using the [UCSC Genome Browser](#). (i.e. mm9, hg18)

Q: I'm changing genome versions, can I just "liftover" my data using UCSC liftover tool, or do I need to remap it to the new genome version?

If you want to do it right, you need to remap it. This is because some regions of the genome that are considered "unique" in one version may suddenly be found multiple times in the new version and vice versa, so using the liftover tool will yield different results from remapping. However, liftover is fine if you're looking for a quick and dirty solution. If you feel like cheating, as Chuck often does, try **convertCoordinates.pl**. - it's a wrapper that uses the "liftOver" program to migrate peak files and whole Tag Directories.

Should I trim my reads when mapping to the genome?

Depends. In the old days, the read quality dropped off quite a bit past ~30 bp, but these days even the end of sequencing reads are pretty high quality. In the end, I would recommend mapping ~32 bp reads with up to 3 mismatches, using only the uniquely alignable reads for downstream analysis. That will give you access to probably 80-90% of what is interesting in your data set.

I have barcodes and/or adapter sequences in my reads. Should I remove them first or just map them?

You should definitely remove the adapter sequences or other "non-biological" sequences before mapping. Various tools can accomplish this. You can check out [homerTools for trimming sequences and dealing with adapters](#). [Galaxy](#) also has a nice variety of tools for accomplishing this type of stuff.

Example - Alignment with bowtie:

Step 1 - Build Index (takes a while, but only do this once):

After installing [bowtie](#), the reference genome must first be "indexed" so that reads may be quickly aligned. You can download pre-made indices from the bowtie website (check for those [here](#) first). Otherwise, to perform make your own from FASTA files, do the following:

1. Download FASTA files for the unmasked genome of interest if you haven't already (i.e. from [UCSC](#))
2. From the directory containing the FASTA files, run the "bowtie-build" command. For example, for hg18:
 - **/path-to-bowtie-programs/bowtie-build chr1.fa,chr2.fa,chr3.fa,...chrY.fa,chrM.fa hg18**
 - Where ... are the rest of the *.fa files. This command will take a long time to run, but will produce several files named **hg18*.ebwt**
3. Copy the *.ebwt files to the bowtie indexes directory so that bowtie knows where to find them later:

- `cp *.ebwt /path-to-bowtie-programs/indexes/`

Step 2 - Align sequences with bowtie (perform for each experiment):

The most common output format for high-throughput sequencing is FASTQ format, which contains information about the sequence (A,C,G,Ts) and quality information which describes how certain the sequencer is of the base calls that were made. In the case of Illumina sequencing, the output is usually a "**s_1_sequence.txt**" file. In addition, much of the data available in the [SRA](#), the primary archive of high-throughput sequencing data, is in this format. To map this data, run the following command:

```
/path-to-bowtie-programs/bowtie -q --best -m 1 -p <# cpu>  
<genome> <fastq file> <output filename>
```

Where <genome> would be hg18 from the index made above, <fastq file> could be "s_1_sequence.txt", and <output filename> something like "s_1_sequence.hg18.alignment.txt"

The parameters "--best" and "-m 1" are needed to make sure bowtie outputs only unique alignments. There are many options and many different ways to perform alignments, with different trade-offs for different types of projects - well beyond the scope of what I am describing here.

NOTE: HOMER contains automated parsing for uniquely aligned reads from output files generated with bowtie in this fashion. Homer also accepts *eland_result.txt and *_export.txt formats from the Illumina pipeline. If different programs are used, or special parsing of output files are needed, please parse/reformat alignment files to general BED format, which is also accepted by HOMER. HOMER also accepts SAM formatted file. If using BAM files, use "samtools view input.bam > output.sam" to convert to a SAM file.



Can't figure something out? Questions, comments, concerns, or other feedback:
cbenner@ucsd.edu



HOMER

Software for motif discovery and next-gen sequencing analysis

Finding Enriched Peaks, Regions, and Transcripts

HOMER contains a program called **findPeaks** that performs all of the peak calling and transcript identification analysis. (Not to be confused with another peak finding program called [FindPeaks](#), which was also very creatively named). Finding peaks is one of the central goals of any ChIP-Seq experiment, and the same basic principles apply to other types of sequencing such as DNase-Seq. The basic idea is to identify regions in the genome where we find more sequencing reads than we would expect to see by chance. There are number of different approaches one can use to find peaks, and correspondingly there are many different methods for identifying peaks from ChIP-Seq experiments. It is not required that you use HOMER for peak finding to use the rest of the tools included in HOMER ([see below](#)).

findPeaks has 3 basic modes of operation:

factor

Peak finding for single contact or focal ChIP-Seq experiments or DNase-Seq. This type of analysis is useful for transcription factors, and aims to identify the precise location of DNA-protein contact. This type of peak finding uses a FIXED width peak size, which is automatically estimated from the Tag Autocorrelation.

histone

Peak finding for broad regions of enrichment found in ChIP-Seq experiments for various histone marks. This analysis finds variable-width peaks.

groseq

De novo transcript identification from strand specific GRO-Seq. This attempts to identify transcripts from nascent RNA sequencing reads.

HOMER does not perform de novo transcript isoform detection from spliced RNA-Seq. As we have just started analyzing RNA-Seq, and there is already a bunch of great work on this topic, there's no need to reinvent the wheel for this type of analysis. We recommend the [Tophat/Cufflinks](#) family of programs for RNA-Seq isoform detection.

Using findPeaks

To run **findPeaks**, you will normally type:

```
findPeaks <tag directory> -style <factor|histone|groseq> -o auto -i <control tag directory>
```

i.e. **findPeaks ERalpha-ChIP-Seq/ -style factor -o auto -i Control-ChIP-Seq/**

Where the first argument must be the tag directory (required). The other options are not required. The "**-style <...>**" option can be either "**factor**", "**histone**", or "**groseq**". Use the "**-i**" option to specify a control experiment tag directory (good idea when doing ChIP-Seq).

Output files

Use the "**-o <filename>**" to specify where to send the resulting peak file. If "**-o**" is not specified, the peak file will be written to **stdout**.

If "**-o auto**" is specified, the peaks will be written to:

```
"<tag directory>/peaks.txt" (-style factor)
"<tag directory>/regions.txt" (-style histone)
"<tag directory>/transcripts.txt" and "<tag directory>/transcripts.gtf" (-style groseq)
```

The top portion of the peak file will contain parameters and various analysis information. This output differs somewhat for GRO-Seq analysis, and is explained in more detail later. Some of the values are self explanatory. Others are explained below:

```
# HOMER Peaks
# Peak finding parameters:
# tag directory = Sox2-ChIP-Seq
#
# total peaks = 10280
# peak size = 137
```

```
# peaks found using tags on both strands
# minimum distance between peaks = 342
# fragment length = 132
# genome size = 4000000000
# Total tags = 9908245.0
# Total tags in peaks = 156820.0
# Approximate IP efficiency = 1.58%
# tags per bp = 0.001907
# expected tags per peak = 0.523
# maximum tags considered per bp = 1.0
# effective number of tags used for normalization = 10000000.0
# Peaks have been centered at maximum tag pile-up
# FDR rate threshold = 0.001000
# FDR effective poisson threshold = 0.000000
# FDR tag threshold = 8.0
# number of putative peaks = 10800
#
# size of region used for local filtering = 10000
# Fold over local region required = 4.00
# Poisson p-value over local region required = 1.00e-04
# Putative peaks filtered by local signal = 484
#
# Maximum fold under expected unique positions for tags = 2.00
# Putative peaks filtered for being too clonal = 36
#
# cmd = findPeaks Sox2-ChIP-Seq -style factor -o auto
#
# Column Headers:
```

Genome size represents the total effective number of mappable bases in the genome (remember each base could be mapped in each direction)

Approximate IP efficiency describes the fraction of tags found in peaks versus genomic background. This provides an estimate of how well the ChIP worked. Certain antibodies like H3K4me3, ERa, or PU.1 will yield very high IP efficiencies (>20%), while most rand in the 1-20% range. Once this number dips below 1% it's a good sign the ChIP didn't work very well and should probably be optimized.

Below the header information are the peaks, listed in each row. Columns contain information about each peak:

- Column 1: PeakID - a unique name for each peak (very important that peaks have unique names...)
- Column 2: chr - chromosome where peak is located
- Column 3: starting position of peak
- Column 4: ending position of peak
- Column 5: Strand (+/-)
- Column 6: Normalized Tag Counts - number of tags found at the peak, normalized to 10 million total mapped tags (or defined by the user)
- Column 7: (-style factor): Focus Ratio - fraction of tags found appropriately upstream and downstream of the peak center. (see below)
- (-style histone/-style groseq): Region Size - length of enriched region
- Column 8: Peak score (position adjusted reads from initial peak region - reads per position may be limited)
- Columns 9+: Statistics and Data from filtering

Two generic tools are available as part of HOMER to convert peak files to BED files and back. This will allow you to upload your peak files to the UCSC Genome Browser, or convert peak files in BED format from another program into a peak file that can be used by HOMER. These programs are named **pos2bed.pl** and **bed2pos.pl**, which can be used the following way:

```
pos2bed.pl peakfile.txt > peakfile.bed
bed2pos.pl peakfile.bed > peakfile.txt
```

Finding Transcription Factor Peaks with HOMER

To find peaks for a transcription factor use the **findPeaks** command:

```
findPeaks <tag directory> -style factor -o auto -i <input tag directory>
```

i.e. **findPeaks ERalpha-ChIP-Seq/ -style factor -o auto -i MCF7-input-ChIP-Seq**

Identification of Putative Peaks

If **findPeaks** is run in "factor" mode, a fixed peak size is selected based on estimates from the autocorrelation analysis performed during the **makeTagDirectory** command. This type of analysis maximizes sensitivity for identifying locations where the factor makes a single contact with the DNA. Peak size can be set manually with "-size <#>".

findPeaks loads tags from each chromosome, adjusting them to the center of their fragments, or by half of the estimated fragment length in the 3' direction (this value is also automatically estimated from the autocorrelation analysis). The fragment length can be specified manually using the "-fragLength <#>" option. It then scans the entire genome looking for fixed width clusters with the highest density of tags. As clusters are found, the regions immediately adjacent are excluded to ensure there are no "piggyback peaks" feed off the signal of large peaks. By default, peaks must be greater than 2x the peak width apart from on another (set manually with "-minDist <#>"). This continues until all tags have been assigned to clusters.

After all clusters have been found, a tag threshold is established to correct for the fact that we may expect to see clusters simply by random chance. Previously, to estimate the expected number of peaks for each tag threshold, HOMER would randomly assign tag positions and repeat the peak finding procedure. HOMER now assumes the local density of tags follows a Poisson distribution, and uses this to estimate the expected peak numbers given the input parameters much more quickly. Using the expected distribution of peaks, HOMER calculates the expected number of false positives in the data set for each tag threshold, setting the threshold that beats the desired False Discovery Rate specified by the user (default: 0.001, "-fdr <#>").

HOMER assumes the total number of mappable base pairs in the genome is 2,000,000,000 bp (** change from previous version. here 2e9 assumes the actual number of mappable positions is actually 2x [think + and - strand]) , which is "close enough" for human and mouse calculations. You can specify a different genome size using "-gsize <#>". HOMER also uses the reads themselves to estimate the size of the genome (i.e. that highest tag position on each chromosome). If this estimate is lower than the default, it will use that value to avoid using too large of a number on smaller genomes (For example, if you used findPeaks on *drosophila* data without specifying "-gsize").

It is important to note that this false discovery rate controls for the random distribution of tags along the genome, and not any other sources of experimental variation. Alternatively, users can specify the threshold using "-poisson <#>" to calculate the tag threshold that yields a cumulative poisson p-value less than provided or "-tagThreshold <#>" to specify a specific number tags to use as the threshold.

Filtering Peaks

The initial step of peak finding is to find non-random clusters of tags, but in many cases these clusters may not be representative of true transcription factor binding events. To increase the overall quality of peaks identified by HOMER, 3 separate filtering steps can be applied to the initial, putative peaks identified:

Using Input/IgG Sequencing as a Control

To use an Input or IgG sequencing run as a control (**HIGHLY RECOMMENDED**), you must first create a separate tag directory for the input experiment (see here). Additionally, you can use other cleaver experiments as a control, such as a ChIP-Seq experiment for the same factor in another cell or in a knockout. To find peaks using a control, type:

findPeaks <tag directory> -style factor -i <control tag directory> -o auto

i.e. **findPeaks ERalpha-ChIP-Seq/ -style factor -i Input-ChIP-Seq/ -o auto**

HOMER uses two parameters to filter peaks against a control experiment. First, it uses a fold change (which is sequencing depth-independent), requiring each putative peak to have 4-fold more normalized tags in the target experiment than the control (or specify a different fold change with "-F <#>"). In the case where there are no input tags near the putative peak, HOMER automatically sets these regions to be set to the average input tag coverage to avoid dividing by zero. HOMER also uses the poisson distribution to determine the chance that the differences in tag counts are statistically significant (sequencing-depth dependent), requiring a cumulative poisson p-value of 0.0001 (change with "-P <#>"). This effectively removes peaks with low tag counts for which there is a chance the differential enrichment is found simply due to sampling error.

One modification in recent versions of HOMER is the in the size of region used to compare experiment with control tags. Since control experiments are not always performed the same way, e.g. different fragment lengths, it helps to enlarge the peak size for the purposes of comparing experiments to ensure control reads found immediately outside of the peak region are still considered. By default, HOMER enlarges peaks by 2x to search the control experiment (change with "-inputSize <#>"). This may reduce specificity when trying to identify certain types of peaks.

Filtering Based on Local Signal

Our experience with peak finding is that often putative peaks are identified in regions of genomic duplication, or in regions where the reference genome likely differs from that of the genome being sequenced. This produces large regions of high tag counts, and if no Input/IgG sample is available, it can be hard to exclude these regions. Also, it may be advantageous to remove putative peaks that a spread out over larger regions as it may be difficult to pin-

point the important regulatory regions within them.

To deal with this, HOMER will filter peaks based on the local tag counts (similar in principle to MACS). By default, HOMER requires the tag density at peaks to be 4-fold greater than in the surrounding 10 kb region. This can be modified using `"-L <#>"` and `"-localSize <#>"` to change the fold threshold and size of the local region, respectively. As with input filtering, the comparison must also pass a poisson p-value threshold of 0.0001, which can be set using `"-LP <#>"` option.

Filtering Based on Clonal Signal

When we first sifted through peaks identified in ChIP-Seq experiments we noticed there are many peaks near repeat elements that contain odd tag distributions. These appear to arise from expanded repeats that result in peaks with high numbers of tags from only a small number of unique positions, even when many of the other positions within the region may be "mappable". To help remove these peaks, HOMER will compare the number of unique positions containing tags in a peak relative to the expected number of unique positions given the total number of tags in the peak. If the ratio between the later and the former number gets too high, the peak is discarded. The fold threshold can be set with the `"-C <#>"` option (default: `"-C 2"`). HOMER uses the **averageTagsPerPosition** parameter in the **tagInfo.txt** file to adjust this calculation as to not over-penalize ChIP-Seq experiments that are already highly "clonal". If analyzing MNase or other restriction enzyme digestion experiments turn this option off (`"-C 0"`);

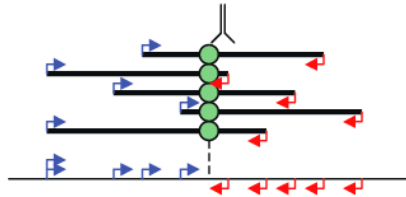
Disabling Filtering

To disable Input, Local, or Clonal filtering set any combination of `"-F 0 -L 0 -C 0"`.

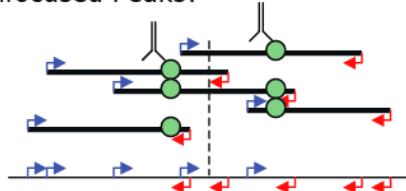
Peak Centering and Focus Ratios

If the option `"-style factor"` or `"-center"` is specified, **findPeaks** will calculate the position within the peak with the maximum ChIP-fragment overlap and calculate a **focusRatio** for the peak. This is not always desired (such as with histone modifications). The focus ratio is defined as the ratio of tags located 5' of the peak center on either strand relative to the total number of tags in the peak. Peaks that contain tags in the ideal positions are more likely to be centered on a single binding site, and these peaks can be used to help determine what sequences are directly bound by a transcription factor. Unfocused peaks, or peaks with low (i.e. <80%) focusRatios may be the result of several closely spaced binding sites or large complexes that cross-link to multiple positions along the DNA. Sometimes this means you have more than one binding site in close proximity (example below), but other times it means you cross-linked the \$#& out of your cells, or you have background in your sample.

Focused Peaks:



Unfocused Peaks:



To isolate focused peaks, you can use the **getFocalPeaks.pl** tool:

```
getFocalPeaks.pl <peak file> <focus % threshold> > focalPeaksOutput.txt
```

i.e. `getFocalPeaks.pl ERpeaks.txt 0.90 > ERfocalPeaks.txt`

Finding Enriched Regions of Variable Length

To find variable length peaks for histone marks, use the **findPeaks** command:

```
findPeaks <tag directory> -style histone -o auto -i <input tag directory>
```

i.e. `findPeaks H3K4me1-ChIP-Seq/ -style histone -o auto -i Input-ChIP-Seq`

If the option **"-style histone"** or **"-region"** is specified, **findPeaks** will stitch together enriched peaks into regions. Note that [local filtering](#) is turned off when finding regions. The most important parameters for region finding are the **"-size"** and **"-minDist"** (and of course the fragment length). First of all, **"-size"** specifies the width of peaks that will form the basic building blocks for extending peaks into regions. Smaller peak sizes offer better resolution, but larger peak sizes are usually more sensitive. By default, **"-style histone"** evokes a peak size of 500.

The second parameter, **"-minDist"**, is usually used to specify the minimum distance between adjacent peaks. If **"-region"** is used, this parameter then specifies the maximum distance between putative peaks that is allowed if they are to be stitched together to form a region. By default this is 2x the peak size. If you think about histone modifications, the signal is never continuous in enriched regions, with reduced signal due to non-unique sequences (that can't be mapped to) and nucleosome depleted regions. **"-minDist"** informs **findPeaks** how big of a gap in the signal will be tolerated for adjacent peaks to be considered part of the same region. (by default **"-style histone"** sets this to 1000).

One thing to note is that you may have to play around with these parameters to get the results you want. If you look at the examples below, you could make arguments for using each of the tracks given what you're interested in and how you would define a "region".

For example: (in the example below, the default size comes from the autocorrelation estimate for the Macrophage-H3K4me1 dataset)

Default Parameters:
findPeaks Macrophage-H3K4me1/ -i Macrophage-IgG -size 150 -minDist 370 > output.txt (i.e. defaults)

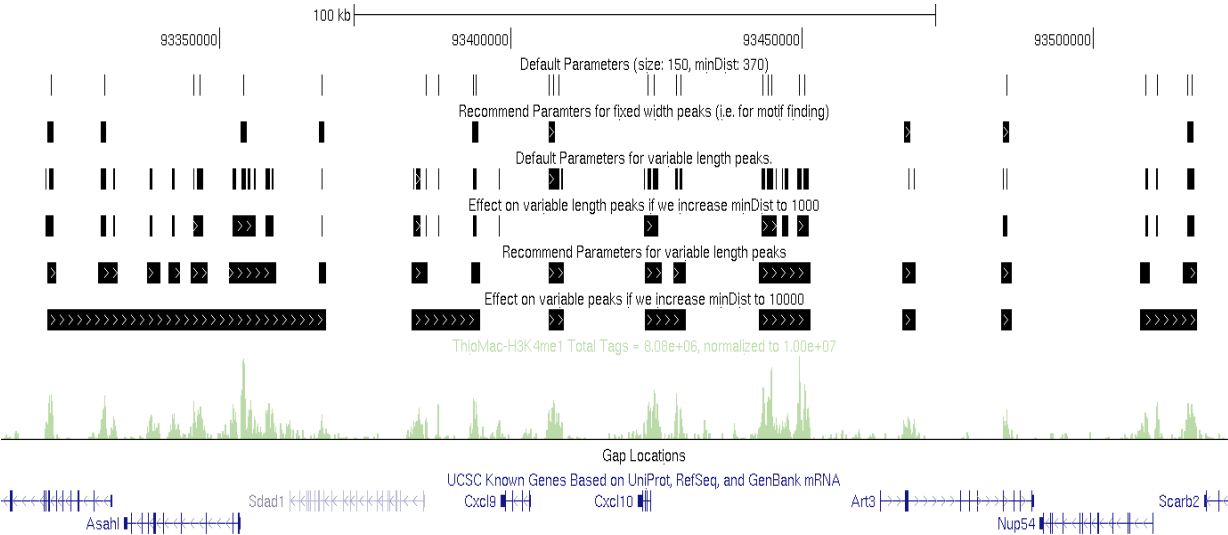
Recommend Parameters for fixed width peaks (i.e. for motif finding):
findPeaks Macrophage-H3K4me1/ -i Macrophage-IgG -size 1000 -minDist 2500 > output.txt

Default Parameters for variable length peaks.
findPeaks Macrophage-H3K4me1/ -i Macrophage-IgG -region -size 150 -minDist 370 > output.txt

Effect on variable length peaks if we increase minDist to 1000.
findPeaks Macrophage-H3K4me1/ -i Macrophage-IgG -region -size 150 -minDist 1000 > output.txt

Recommend Parameters for variable length peaks (H3K4me1 at least).
findPeaks Macrophage-H3K4me1/ -i Macrophage-IgG -region -size 1000 -minDist 2500 > output.txt

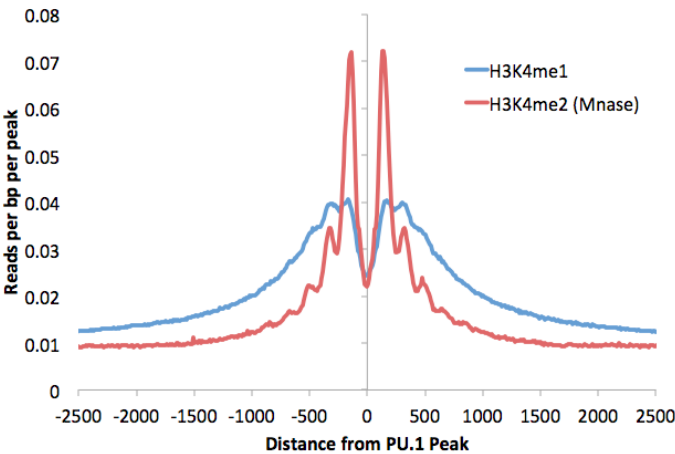
Effect on variable peaks if we increase minDist to 10000 (H3K4me1 at least).
findPeaks Macrophage-H3K4me1/ -i Macrophage-IgG -region -size 1000 -minDist 10000 > output.txt



Finding Histone Modification Peaks and Motif Finding

Finding peaks using histone modification data can be a little tricky - largely because we have very little idea what the histone marks actually do. If you want to find peaks in histone modification data with the purpose of analyzing them for enriched motifs, read this section. The problem with histone modification data (and some other types) is that the signal can spread over large distances. Trying to analyze large, variable length regions for motif enrichment is very difficult and not recommended. As such it is recommended sometimes a better idea to use fixed-size peak finding on histone marks (i.e. H3K4me1 enhancer mark) to improve motif analysis results. Using fixed size peaks helps make sure the assumptions needed for Motif Finding are, let's say, *less violated*, when dealing with regions of constant size.

There are two important differences between finding fixed width peaks for transcription factors and histone modifications. The first is the size of the peaks: Most histone modifications should be analyzed using a peak size in the range of 500-2000 usually (i.e. "-size 1000"). Below is an example of the histone mark distribution around transcription factor peaks (i.e. the things you're hoping Motif Finding will identify), which can be used to help estimate the parameters:



The other is that you should omit the "-center" option (i.e. do not specify "-style factor"). Since you are looking at a region, you do not necessarily want to center the peak on the specific position with the highest tag density, which may be at the edge of the region. Besides, in the case of histone modifications at enhancers, the highest signal will usually be found on nucleosomes surrounding the center of the enhancer, which is where the functional sequences and transcription factor binding sites reside. Consider H3K4me marks surrounding distal PU.1 transcription factor peaks. Typically, adding the -center option moves peaks further away from the functional sequence in these scenarios. An example for finding peaks:

```
findPeaks H3K4me1-ChIP-Seq/ -i Input-ChIP-Seq -size 1000 -o H3K4me1-ChIP-Seq/peaks.1kb.txt
```

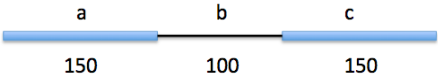
One issue with finding histone modification peaks using the defaults in HOMER is that the Local filtering step removes several of the peaks due to the "spreading" nature of many histone modifications. This can be good and bad. If you are looking for nice concentrated regions of modified histones, the resulting peaks will be a nice set for further analysis such as motif finding. However, if you are looking to identify every region in the cell that has an appreciable amount of modified histone, you may want to disable local filtering, or consider using the "-region" option below e.g.:

```
findPeaks H3K4me1-ChIP-Seq/ -i Input-ChIP-Seq -size 1000 -L 0 -o H3K4me1-ChIP-Seq/peaks.1kb.txt
```

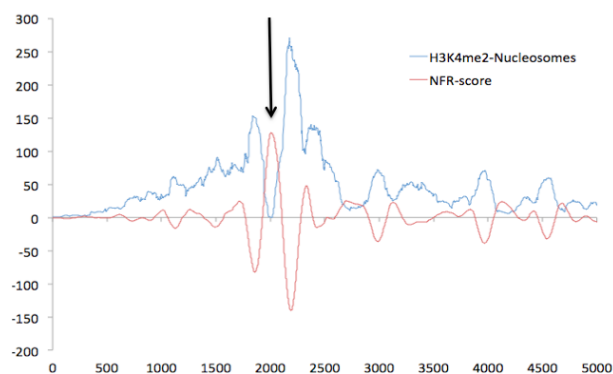
Also, if using MNase-treated chromatin (i.e. nucleosome mapping), you may want to use "-C 0" to avoid filtering peaks composed of highly clonal tag positions. These can arise from well positioned nucleosomes (or sites that are nicely digested by MNase at least).

Nucleosome Free Regions (NFR) -nfr

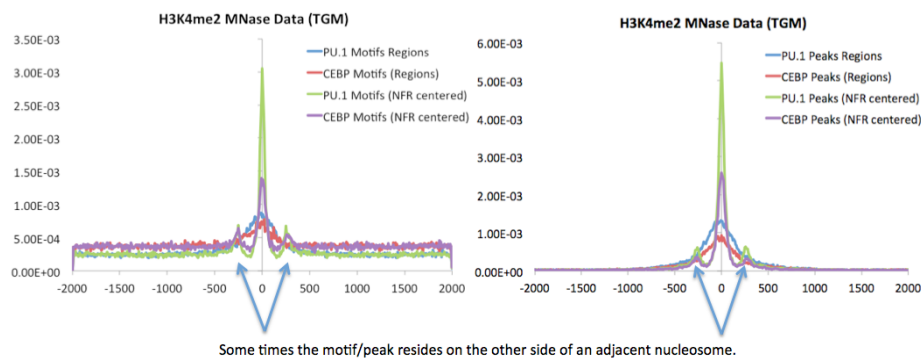
Just like peak centering for transcription factors, **findPeaks** has an option to look for large "dips" in the histone modification signal to infer where the primary nucleosome free region (NFR) is in the region. By adding "-nfr" to the command, HOMER will search for the location within the region that has the greatest differential in ChIP-signal and assign that location as the peak center.



NFR score = (a+c) – b
Where a, b, and c are normalized for their length



Works much better with MNase treated ChIP-Seq samples. The output file will then be centered on the NFR - this is useful for motif finding. By using NFRs instead of whole regions, you can narrow the search for regulatory elements down the +/- 100 bp instead of +/-500 or 1000 bp. Below is an example of ChIP-Seq peak locations with respect to center of H3K4me2-ChIP-Seq regions generated with and without the "-nfr" flag in Macrophages.



Peak finding and Sequencing Saturation

HOMER does not try to estimate sequencing saturation, which is the estimate of whether or not you have sequenced enough tags to identify all the peaks in a given experiment. Generally speaking, if you sequence more, you will get more peaks since your sensitivity will increase. The only real way to assess this is if you can somehow show that all of the "functional" or "real" peaks have high tag counts (i.e. are well above the threshold for identifying a peak), meaning that sequencing more is not likely to identify more "real" peaks. This generally cannot be determined by simply re-sampling the data and repeating the peak finding procedure - you need some sort of outside information to assess peak quality, such as motif enrichment or something else. Simply re-sampling will likely only tell you if you've gotten to the point where you are simply re-sequencing the same fragments again - i.e. becoming clonal.

Analyzing GRO-Seq: *de novo* transcript identification

To find transcripts directly from GRO-Seq, use the `findPeaks` command:

```
findPeaks <tag directory> -style groseq -o auto
```

```
i.e. findPeaks Macrophage-GroSeq -style groseq -o auto
```

GRO-Seq analysis does not make use of an control tag directory

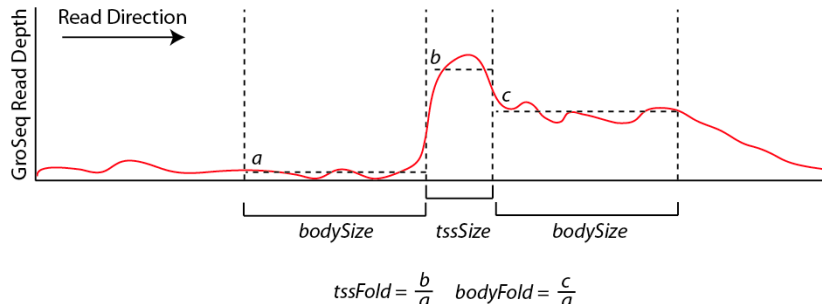
Basic Idea behind GRO-Seq Transcript identification

Finding transcripts using strand-specific GRO-Seq data is not trivial. GRO-Seq measures the production of nascent RNA, and is capable of revealing the loction of protein coding transcripts, promoter anti-sense transcripts, enhancer templated transcripts, long and short functional non-coding and miRNA transcripts, Pol III and Pol I transcripts, and whatever else is being transcribed in the cell nucleus. Identification and quantiftion of these transcripts is important for downstream analysis. Traditional RNA-Seq tools mainly focus on mRNA, which has different features than GRO-Seq, and are generally not useful for identifying GRO-Seq transcripts.

Important NOTE: Just as with ChIP-Seq, not all GRO-Seq data was created equally. Data created by different labs can

have features that make it difficult to have an single analysis technique that works perfectly for each one. As such, there are many parameters to play with the help get the desired results.

A large number of assumptions go into the analysis and are covered in more detail in the [GRO-Seq tutorial \(coming soon\)](#). In a nutshell, findPeaks tracks along each strand of each chromosome, searching for regions of continous GRO-Seq signal. Once it encounters high numbers of GRO-Seq reads, it starts a transcript. If the signal decreases significantly or disappears, the putative transcript is stopped. If the signal increases significantly (and sustainably), then a new transcript is considered from that point on. If the signal spikes, but overall does not increase over a large distance, it is considered an artifact or pause site and not considered in the analysis. Below is a chart that helps explain how the transcript detection works:



By default, new transcripts are created when the *tssFold* exceeds 4 and *bodyFold* exceed 3 ("**-tssFold <#>**", "**-bodyFold <#>**"). A small pseudo-count is added to the tag count from region *a* above to avoid dividing by zero and helps serve to set a minimum threshold for transcript detection ("**-pseudoCount <#>**", default: 1). Most transcripts show robust signal at the start of the transcript, and the *tssFold* helps select for these regions with high accuracy. The *bodyFold* is important for distinguishing between "spikes" in signal and real start sites; if a transcript is real, it's likely that increased levels of transcription follow behind the putative TSS. If the signal is roughly equal before and after the putative TSS, it is more likely to be an artifact.

To increase sensitivity, HOMER tries to adjust the size of the *bodySize* parameter above since it essentially defines the resolution of the detected transcript. If there are a large number of GRO-Seq tags in a region, the *bodySize* can be small since there is adequate data to estimate the location of the transcript. However, if the data is relatively sparse, the *bodySize* needs to be large to get a reliable estimate of the level of the transcript. The minimum and maximum *bodySize*s are 600 and 10000 bp ("**-minBodySize <#>**", "**-maxBodySize <#>**"). HOMER uses the smallest *bodySize* that contains at least *x* number of tags, where *x* is determined as the number of tags where the chance of detecting a *bodyFold* change is less than 0.00001 assuming the read depth varies according to the poisson distribution (adjustable with "**-confPvalue <#>**", or directly with "**-minReadDepth <#>**"). The basic idea is that the threshold for tag counts must be high enough that we don't expect it to vary too much by chance.

Using uniquely mappable regions to improve results

Since some transcripts cover very large regions, there are many places where genomic repeats interrupt the GRO-Seq signal of continous transcripts. To help deal with this problem, HOMER can take advantage of mappability information to help estimate transcript levels where uniquely mapping sequencing reads is not possible. In general this information is not really that helpful for ChIP-Seq analysis, but in this case it can make an important difference. For now, HOMER only take specially formatted binary files available below. To use them, download the appropriate version and unzip the archive:

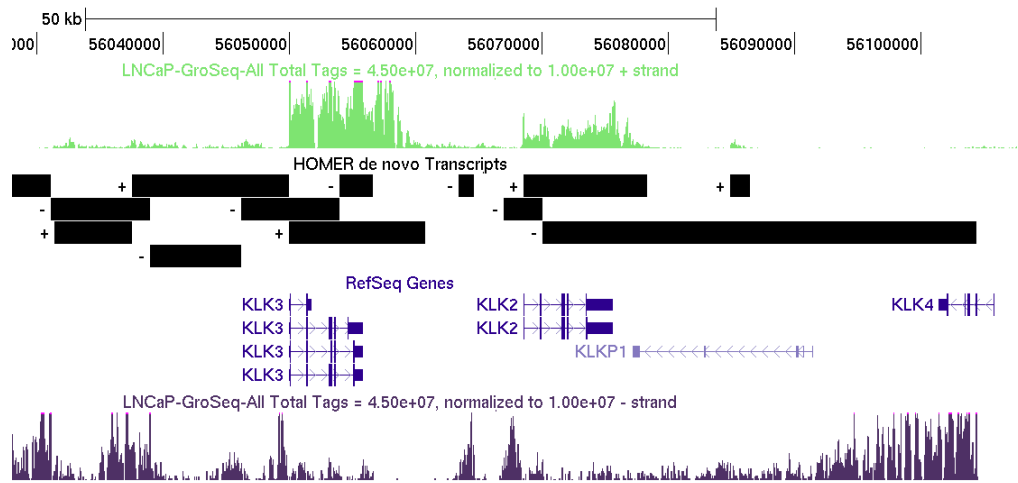
Human: [hg18 hg19](#)
 Mouse: [mm8 mm9](#)
 Fly: [dm3](#)

To use the uniq-map information, specify the location of the unzipped directory on the command line with "**-uniqmap <directory>**":

```
findPeak Macrophage-GroSeq/ -style groseq -o auto -uniqmap mm9-uniqmap/
```

GRO-Seq analysis output

Running findPeaks in groseq mode will produce a file much like the one produced for traditional peak finding, complete with a header section listing the parameters and statistics from the analysis. HOMER can also produce a GTF (gene transfer format) file for use with various programs. If "**-o auto**" is used to specify output, a "transcripts.gtf" file will be created in the tag directory. Otherwise, you can specify the name of the GTF output file by use "**-gtf <filename>**". The GTF file can also be easily uploaded to the UCSC Genome Browser to visualize your transcripts.



The GRO-Seq transcript detection works pretty well, but is likely to get some face-lifts in the near future.

Command line options for findPeaks

Usage: findPeaks <tag directory> [options]

Finds peaks in the provided tag directory. By default, peak list printed to stdout

General analysis options:

- o <filename|auto> (file name for to output peaks, default: stdout)
"-o auto" will send output to "<tag directory>/peaks.txt", ".../regions.txt",
or ".../transcripts.txt" depending on the "-style" option
- style <option> (Specialized options for specific analysis strategies)
factor (transcription factor ChIP-Seq, uses -center, default)
histone (histone modification ChIP-Seq, region based, uses -region -size 500 -L 0)
groseq (de novo transcript identification from GroSeq data)

chipseq/histone options:

- i <input tag directory> (Experiment to use as IgG/Input/Control)
- size <#> (Peak size, default: auto)
- minDist <#> (minimum distance between peaks, default: peak size x2)
- gsize <#> (Set effective mappable genome size, default: 4e9 [Think double stranded!])
- fragLength <#|auto> (Approximate fragment length, default: auto)
- inputFragLength <#|auto> (Approximate fragment length of input tags, default: auto)
- tbp <#> (Maximum tags per bp to count, 0 = no limit, default: auto)
- inputtbp <#> (Maximum tags per bp to count in input, 0 = no limit, default: auto)
- strand <both|separate> (find peaks using tags on both strands or separate, default:both)
- norm # (Tag count to normalize to, default 10000000)
- center (Centers peaks on maximum tag overlap and calculates focus ratios)
- region (extends start/stop coordinates to cover full region considered "enriched")

Peak Filtering options: (set -F/-L/-C to 0 to skip)

- F <#> (fold enrichment over input tag count, default: 4.0)
- P <#> (poisson p-value threshold relative to input tag count, default: 0.0001)
- L <#> (fold enrichment over local tag count, default: 4.0)
- LP <#> (poisson p-value threshold relative to local tag count, default: 0.0001)
- C <#> (fold enrichment limit of expected unique tag positions, default: 2.0)
- localSize <#> (region to check for local tag enrichment, default: 10000)
- inputSize <#> (Size of region to search for control tags, default: 2x peak size)
- fdr <#> (False discovery rate, default = 0.001)
- poisson <#> (Set poisson p-value cutoff, default: uses fdr)
- tagThreshold <#> (Set # of tags to define a peak, default: uses fdr)

GroSeq Options: (Need to specify "-style groseq"):

- tssSize <#> (size of region for initiation detection/artifact size, default: 300)
- minBodySize <#> (size of regoin for transcript body detection, default: 600)
- maxBodySize <#> (size of regoin for transcript body detection, default: 10000)
- tssFold <#> (fold enrichment for new initiation dectection, default: 4.0)
- bodyFold <#> (fold enrichment for new transcript dectection, default: 3.0)
- endFold <#> (end transcript when levels are this much less than the start, default: 25.0)

-fragLength <#> (Approximate fragment length, default: 150)
-uniqmap <directory> (directory of binary files specifying uniquely mappable locations)
Download from <http://biowhat.ucsd.edu/homer/groseq/>
-confPvalue <#> (confidence p-value: 0.00001)
-minReadDepth <#> (Minimum initial read depth for transcripts, default: auto)
-gtf <filename> (Output de novo transcripts in GTF format)
"-o auto" will produce <dir>/transcripts.txt and <dir>/transcripts.gtf

Links to alternative peak finding software

Lots of quality programs exist for finding peaks in ChIP-Seq data. Most use slightly different assumptions and peak definitions that result in slightly different sets of peaks. Many of these programs output peak files in BED format. To covert these to HOMER peak file format, use the bed2pos.pl program to convert the file (as of now, most HOMER programs work with BED files, so this isn't really necessary):

bed2pos.pl peaks.bed > peaks.txt

Peak finding software links:

- [MACS](#) (Liu Lab)
- [ChIPSeq Peak Finder](#) (Wold Lab)
- [CCAT](#) (Good for histone modifications)
- [FindPeaks](#)
- Probably hundreds of others: Google it!



Can't figure something out? Questions, comments, concerns, or other feedback:
cbenner@ucsd.edu



HOMER

Software for motif discovery and ChIP-Seq analysis

File Formats

List of files used by HOMER - might be helpful when encountering problems.

[Another good resource on file formats: UCSC Genome Browser File Formats](#)

Peak/Positions files

These files specify genomic locations similar to BED files. They are tab-delimited text files with a minimum of 5 columns (additional columns are ignored). They are 1-indexed and inclusive, meaning the first nucleotide of a chromosome is referenced as position 1. They are inclusive in the sense that a line with a start of 100 and end of 200 indicates of region of size 101. Columns are as followed:

1. peak name (should be unique)
2. chromosome
3. starting position [integer] (1-indexed)
4. end position [integer]
5. strand [either 0/1 or +/-] (in HOMER strand of 0 is +, 1 is -)
6. Optional/ignored ...

Peak/Position files are very similar to BED files - to convert them use **pos2bed.pl** or **bed2pos.pl**.

BED files

These are essentially the same as Peak/Position files, except that they have a stricter [definition](#) but greater portability. They are also tab-delimited text files - the important difference is that they are 0-indexed, meaning the first nucleotide of the chromosome is referenced as position 0.

1. chromosome
2. starting position [integer] (0-indexed)
3. ending position [integer]
4. peak name
5. value (usually ignored)
6. strand [+/-]

BED files also come in a short form:

1. chromosome

2. starting position [integer] (0-indexed)
3. ending position [integer]
4. strand [+/-]

Peak/Position files are very similar to BED files - to convert them use **pos2bed.pl** or **bed2pos.pl**.

Motif files

These are files for specifying motifs, and are created by HOMER during motif discovery. They are tab-delimited text files. A more elaborate description of the format and how to tinker with it is [here](#). Basically, each motif within the file contains a header row starting with a ">", followed by several rows with 4 columns, specifying the probabilities of each nucleotide at each position.

```
>ASTTCCTCTT 1-ASTTCCTCTT 8.059752 -23791.535714 0
T:17311.0(44 ...
0.726 0.002 0.170 0.103
0.002 0.494 0.354 0.151
0.016 0.017 0.014 0.954
0.005 0.006 0.027 0.963
0.002 0.995 0.002 0.002
0.002 0.989 0.008 0.002
0.004 0.311 0.148 0.538
0.002 0.757 0.233 0.009
0.276 0.153 0.030 0.542
0.189 0.214 0.055 0.543
```

The first row starts with a ">" followed by various information, and the other rows are the positions specific probabilities for each nucleotide (A/C/G/T). These values do not need to be between 0-1. HOMER will automatically normalize whatever values are there, so interger counts are ok. The header row is actually TAB delimited, and contains the following information:

1. ">" + Consensus sequence (not actually used for anything, can be blank) example: >ASTTCCTCTT
2. Motif name (should be unique if several motifs are in the same file) example: 1-ASTTCCTCTT or NFkB
3. Log odds detection threshold, used to determine bound vs. unbound sites (**mandatory**) example: 8.059752
4. (optional) log P-value of enrichment, example: -23791.535714
5. (optional) 0 (A place holder for backward compatibility, used to describe "gapped" motifs in old version, turns out it wasn't very useful :)
6. (optional) Occurrence Information separated by commas, example:
T:17311.0(44.36%),B:2181.5(5.80%),P:1e-10317
 1. T: # (%) - number of target sequences with motif, % of total of total targets
 2. B: # (%) - number of background sequences with motif, % of total background
 3. P: # - final enrichment p-value
7. (optional) Motif statistics separated by commas, example:

Tpos:100.7,Tstd:32.6,Bpos:100.1,Bstd:64.6,StrandBias:0.0,Multiplicity:1.13

1. Tpos: average position of motif in target sequences (0 = start of sequences)
2. Tstd: standard deviation of position in target sequences
3. Bpos: average position of motif in background sequences (0 = start of sequences)
4. Bstd: standard deviation of position in background sequences
5. StrandBias: log ratio of + strand occurrences to - strand occurrences.
6. Multiplicity: The average number of occurrences per sequence in sequences with 1 or more binding site.

Only the first 3 columns are needed. In fact, the rest of the columns are really just statistics from motif finding and aren't important when searching for instances of a motif.

The MOST IMPORTANT value is the 3rd column - this sets the detection threshold, which specifies whether a given sequence is enough of a "match" to be considered recognized by the motif. More on that below.

Internal File Formats:

These are files that you normally won't modify or play with, but in case your interested...

***.tags.tsv files**

These are files used to store sequencing data in HOMER tag directories. They are tab-delimited text files that are sorted to allow for relatively quick access and processing.

1. blank (can be used for a name)
2. chromosome
3. position (1-indexed)
4. strand (0 or 1, +/- not allowed here)
5. Number of reads (can be fractional)
6. length of the read (optional)



Can't figure something out? Questions, comments, concerns, or other feedback:
cbenner@ucsd.edu



HOMER

Software for motif discovery and next-gen sequencing analysis

Creating Custom Motif Matrices

A common task when doing regulatory element analysis is to scan for a specific sequence/motif - but not necessarily with one that is found using motif finding. Often you want to find a very specific sequence, or you want to load your own motif matrix derived from another source. This page will help explain how to get HOMER to play nice with your custom motifs.

Below is a description of the HOMER *.motif format for specifying motifs, as well as some tricks & tips on creating simple motif files.

*.motif format files

HOMER works exclusively with homer motif formatted files - so if you want to find a sequence or motif with HOMER, you must first create a "motif" file. A typical motif file will look something like:

```
>ASTTCCTCTT 1-ASTTCCTCTT 8.059752 -23791.535714 0
T:17311.0(44 ...
0.726 0.002 0.170 0.103
0.002 0.494 0.354 0.151
0.016 0.017 0.014 0.954
0.005 0.006 0.027 0.963
0.002 0.995 0.002 0.002
0.002 0.989 0.008 0.002
0.004 0.311 0.148 0.538
0.002 0.757 0.233 0.009
0.276 0.153 0.030 0.542
0.189 0.214 0.055 0.543
```

The first row starts with a ">" followed by various information, and the other rows are the positions specific probabilities for each nucleotide (A/C/G/T). These values do not need to be between 0-1. HOMER will automatically normalize whatever values are there, so interger counts are ok. The header row is actually TAB delimited, and contains the following information:

1. ">" + **Consensus sequence (not actually used for anything, can be blank) example: >ASTTCCTCTT**
2. **Motif name (should be unique if several motifs are in the same file) example: 1-ASTTCCTCTT or NFkB**
3. **Log odds detection threshold, used to determine bound vs. unbound sites (mandatory) example: 8.059752**

- 4. (optional) log P-value of enrichment, example: -23791.535714
- 5. (optional) 0 (A place holder for backward compatibility, used to describe "gapped" motifs in old version, turns out it wasn't very useful :)
- 6. (optional) Occurrence Information separated by commas, example:
T:17311.0(44.36%),B:2181.5(5.80%),P:1e-10317
 - 1. T:#(%) - number of target sequences with motif, % of total of total targets
 - 2. B:#(%) - number of background sequences with motif, % of total background
 - 3. P:# - final enrichment p-value
- 7. (optional) Motif statistics separated by commas, example:
Tpos:100.7,Tstd:32.6,Bpos:100.1,Bstd:64.6,StrandBias:0.0,Multiplicity:1.13
 - 1. Tpos: average position of motif in target sequences (0 = start of sequences)
 - 2. Tstd: standard deviation of position in target sequences
 - 3. Bpos: average position of motif in background sequences (0 = start of sequences)
 - 4. Bstd: standard deviation of position in background sequences
 - 5. StrandBias: log ratio of + strand occurrences to - strand occurrences.
 - 6. Multiplicity: The average number of occurrences per sequence in sequences with 1 or more binding site.

Only the first 3 columns are needed. In fact, the rest of the columns are really just statistics from motif finding and aren't important when searching for instances of a motif.

The MOST IMPORTANT value is the 3rd column - this sets the detection threshold, which specifies whether a given sequence is enough of a "match" to be considered recognized by the motif. More on that below.

Creating Simple Motifs with seq2profile.pl

HOMER comes with a handy little tool called **seq2profile.pl**. This program automates the creation of motifs from consensus sequences from letters ACGTN.

seq2profile.pl <consensus> [# mismatches] [name] > output.motif
i.e. **seq2profile.pl GGAAGT 0 ets > output.motif**

Output from this example looks like this:

```
>GGAAGT ets 8.28973911259755
0.001 0.001 0.997 0.001
0.001 0.001 0.997 0.001
0.997 0.001 0.001 0.001
0.997 0.001 0.001 0.001
0.001 0.001 0.997 0.001
0.001 0.001 0.001 0.997
```

This script will automatically set the 3rd column to a threshold to only detect

perfect matches. Using this file with tools like `annotatePeaks.pl` will only find sequences matching GGAAGT (or ACTTCC on the reverse strand) If we want to allow up to one mismatch:

i.e. `seq2profile.pl GGAAGT 1 ets > output.motif`

Output from this example looks like this:

```
>GGAAGT ets 1.38498834263571
0.001 0.001 0.997 0.001
0.001 0.001 0.997 0.001
0.997 0.001 0.001 0.001
0.997 0.001 0.001 0.001
0.001 0.001 0.997 0.001
0.001 0.001 0.001 0.997
```

The detection threshold now changed to allow up to one mismatch. This will now identify GGAAGT, GGAAGA, GGAAGC, GGAAGG, ... etc.

Creating motif files manually

You can also create a motif using a text editor or Excel. The first line should contain 3 tab-separated columns with ">whatever", "name", and "threshold", followed by motif matrix. You can also start from an existing motif or use `seq2profile.pl` to start a motif for you.

Once you get the probabilities the way you want it (i.e. maybe you copied them from JASPAR, or from in vitro affinity selection, or a set of binding sites, or whatever), you need to pick the correct detection threshold. During motif discovery, HOMER optimizes this threshold as part of the algorithm. However, since you're making up your own motif, you need to figure out how degenerate you want your motif to be. This is easily the biggest hang-up for people new to the concept of probability matrices and motif finding.

Motif Scanning

In order to select the proper detection threshold, it helps to understand how motifs are "scored" by HOMER. Any given sequence can be scored using the probability matrix. HOMER calculates the score by adding the "log-odds probabilities" for the nucleotide found in each position.

Score for GGATGT

$$\text{score} = \log(pG_1/0.25) + \log(pG_2/0.25) + \log(pA_3/0.25) + \log(pT_4/0.25) + \log(pG_5/0.25) + \log(pT_6/0.25)$$

If the nucleotide in the given position has a high probability in the motif matrix (i.e. 0.9), a positive number is added to the score. If the nucleotide as a neutral probability (i.e. 0.25), then the score is unchanged, and if a

nucleotide is disfavored in the probability matrix (i.e. 0.001), then a negative number is added to the score.

Homer fixes the log-odds expectation at 0.25 for each nucleotide. If an organism has a strong imbalance in nucleotide composition, this causes HOMER sacrifices some sensitivity by keeping the expectation at 0.25 instead of adjusting this to reflect the general sequence composition. However, the huge upside to fixing this at 0.25 is that you can use the motif with HOMER on any sequence in any context and always get recognize the same sequences, which makes life much easier.

If the score is above the threshold (from the 3rd column of the motif file), HOMER will identify the sequence as "recognized" by the motif. If the score is lower, the sequence will be ignored. Most motif thresholds are around 5.0-10.0

To select the correct threshold, you may need to "guess and check" your results to ensure your motif is recognizing the correct sequences. Usually, you can start with a threshold around 5-10. For example, run **findMotifs.pl/findMotifsGenome.pl** with "**-find motifFile.motif**". The score of each motif is found in the 6th column of the output. Looking through the file and the sequences that were identified will give you a better idea of what to set the score at.



Can't figure something out? Questions, comments, concerns, or other feedback:
cbenner@ucsd.edu



HOMER

Software for motif discovery and ChIP-Seq analysis

This is the old version of the documentation: [New Version](#)

Introduction to ChIP-Seq

Coming soon... for now just type "introduction to chip-seq" at your favorite search engine



Can't figure something out? Questions, comments, concerns, or other feedback:
cbenner@ucsd.edu



HOMER

Software for motif discovery and ChIP-Seq analysis

This is the old version of the documentation: [New Version](#)

Alignment of High-throughput Sequencing Data

Homer does not perform alignment - this is something that must be done before running homer. Several quality tools are available for alignment of short reads to large genomes. Check out [this link](#) for a list of programs that do short read alignment. BLAST, BLAT, and other traditional alignment programs, while great at what they do, are not practical for alignment of these types of data.

If you need help deciding on a program to use, I'll recommend [bowtie](#) (it's nice and fast).

If you have a core that maps your data for you, don't worry about this step. However, in many cases there is public data available that hasn't been mapped to the genome or mapped to a different version of the genome or mapped with different parameters. In these cases it is nice to be able to map data yourself to keep a nice, consistent set of data for analysis.

Which reference genome (version) should I map my reads to?

Both the organism and the exact *version* (i.e. hg18, hg19) are very important when mapping sequencing reads. Reads mapped to one version are NOT interchangeable with reads mapped to a different version. I would follow this recommendation list when choosing a genome (Obviously try to match species or sub species when selecting a genome):

1. Do you have a favorite genome in the lab that already has a bunch of experiments mapped to it? Use that one.
2. Do any of your collaborators have a favorite genome?
3. Use the latest stable release - I would recommend using genomes curated at UCSC so that you can easily visualize your data later using the [UCSC Genome Browser](#). (i.e. mm9, hg18)

Should I trim my reads when mapping to the genome?

Depends. In the old days, the read quality dropped off quite a bit past ~30 bp, but these days even the end of sequencing reads are pretty high quality. In the end, I would recommend mapping ~32 bp reads with up to 3 mismatches, using only the uniquely alignable reads for downstream analysis. That will give you access to probably 80-90% of what is interesting in your data set.

Example - Alignment with bowtie:

Step 1 - Build Index (takes a while, but only do this once):

After installing [bowtie](#), the reference genome must first be "indexed" so that reads may be quickly aligned. You can download pre-made indecies from the bowtie website (check for those [here](#) first). Otherwise, to perform make your own from FASTA files, do the following:

1. Download FASTA files for the unmasked genome of interest if you haven't already (i.e. from [UCSC](#))
2. From the directory containing the FASTA files, run the "bowtie-build" command. For example, for hg18:
 - **/path-to-bowtie-programs/bowtie-build chr1.fa,chr2.fa,chr3.fa,...chrY.fa,chrM.fa hg18**
 - Where ... are the rest of the *.fa files. This command will take a long time to run, but will produce several files named **hg18.*.ebwt**
3. Copy the *.ebwt files to the bowtie indexes directory so that bowtie knows where to find them later:
 - **cp *.ebwt /path-to-bowtie-programs/indexes/**

Step 2 - Align sequences with bowtie (perform for each experiment):

The most common output format for high-throughput sequencing is FASTQ format, which contains information about the sequence (A,C,G,Ts) and quality information which describes how certain the sequencer is of the base calls that were made. In the case of Illumina sequencing, the output is usually a "s_1_sequence.txt" file. In addition, much of the data available in the [SRA](#), the primary archive of high-throughput sequencing data, is in this format. To map this data, run the following command:

```
/path-to-bowtie-programs/bowtie -q --best -m 1 -p <# cpu>  
<genome> <fastq file> <output filename>
```

Where <genome> would be hg18 from the index made above, <fastq file> could be "s_1_sequence.txt", and <output filename> something like "s_1_sequence.hg18.alignment.txt"

The parameters "--best" and "-m 1" are needed to make sure bowtie outputs only unique alignments. There are many options and many different ways to perform alignments, with different trade-offs for different types of projects - well beyond the scope of what I am describing here.

NOTE: HOMER contains automated parsing for uniquely aligned reads from output files generated with bowtie in this fashion. Homer also accepts *eland_result.txt and *_export.txt formats from the Illumina pipeline. If different programs are used, or special parsing of output files are needed, please parse/reformat alignment files to general BED format, which is also accepted by HOMER.

[Back to ChIP-Seq Analysis](#)



Can't figure something out? Questions, comments, concerns, or other feedback:
cbenner@ucsd.edu



HOMER

Software for motif discovery and ChIP-Seq analysis

This is the old version of the documentation: [New Version](#)

ChIP-Seq Analysis: Step 1, Creating a "Tag Directory"

To facility the analysis of ChIP-Seq (or any other type of short read re-sequencing data), it is useful to first transform the sequence alignment into platform independent data structure representing the experiment, analogous to loading the data into a database. HOMER does this by placing all relevant information about the experiment into a "Tag Directory", which is essentially a directory on your computer that contains several files describing your experiment.

To create a "Tag Directory", you must have alignment files in one of the following formats:

- BED format
- *.eland_result.txt or *_export.txt format from the Illumina pipeline
- bowtie output format

If your alignment is in a different format, it is recommended that you convert it into a BED file format:

Column1: chromosome
Column2: start position
Column3: end position
Column4: Name (or strand +/-)
Column5: Number of reads at this position
Column6: Strand +/-

Alternatively (or in combination), you can make tag directories from existing tag directories or from tag files (explained below).

To make a tag directory, run the following command:

```
makeTagDirectory <Output Directory Name> [options] <alignment file1>  
[alignment file 2] ...
```

Where the first argument must be the output directory (required). If it does not exist, it will be created. If it does exist, it will be overwritten.

An example:

```
makeTagDirectory Macrophage-PU.1-ChIP-Seq/ pu1.lane1.bed
```

pu1.lane2.bed pu1.lane3.bed

Several additional options exist for **makeTagDirectory**. The program attempts to guess the format of your alignment files, but if it is unsuccessful, you can force the format with "**-format <X>**". To combine tag directories, for example when combining two separate experiments into one, do the following:

makeTagDirectory Combined-PU.1-ChIP-Seq/ -d Exp1-ChIP-Seq/ Exp2-ChIP-Seq/ Exp3-ChIP-Seq/

What does makeTagDirectory do?

makeTagDirectory basically parses through the alignment file and splits the tags into separate files based on their chromosome. As a result, several *.tags.tsv files are created in the output directory. These are made to very efficiently return to the data during downstream analysis. This also helps speed up the analysis of very large data sets without running out of memory.

In the end, your output directory will contain several *.tags.tsv files, as well as a file named "**tagInfo.txt**". This file contains information about your sequencing run, including the total number of tags considered. This file is used by later programs to quickly reference information about the experiment, and can be manually modified to set certain parameters for analysis.

makeTagDirectory also performs several quality control steps which are covered in the next section.

Command line options of makeTagDirectory command:

Usage: **makeTagDirectory** <directory> <alignment file 1> [file 2] ... [options]

Creates a platform-independent 'tag directory' for later analysis.

Currently BED, Eland, and bowtie files are accepted. The program will try to automatically detect the alignment format if not specified

Existing tag directories can be added or combined to make a new one using -d/-t

If more than one format is needed and the program cannot auto-detect it properly, make separate tag directories by running the program separately, then combine them.

Options:

-genome <genome name> (specify genome for later analysis)

To list available genomes, run "??"

-name <experiment name> (optional, names the experiment)

-format <X> where X can be: (with column specifications underneath)

bed - BED format files:

(1:chr,2:start,3:end,4:+/- or read name,5:# tags,6:+/-)

bowtie - output from bowtie (run with --best -k 2 options)

(1:read name,2:+/-,3:chr,4:position,5:seq,6:quality,
7:NA,8:mismatch info)
eland_result - output from basic eland
(1:read name,2:seq,3:code,4:#zeroMM,5:#oneMM,6:#twoMM,7:chr,
8:position,9:F/R,10:-mismatches
eland_export - output from illumina pipeline (22 columns total)
(1-5:read name
info,9:sequence,10:quality,11:chr,13:position,14:strand)
-C (color space mapping with bowtie)
-keep (keep one mapping of each read regardless if multiple equal mappings
exist)
-forceBED (if 5th column of BED file contains stupid values, like mapping
quality
instead of number of tags, then ignore this column)
-d <tag directory> [tag directory 2] ... (add Tag directory to new tag directory)
-t <tag file> [tag file 2] ... (add tag file i.e. *.tags.tsv to new tag directory)

Next: [Basic quality control \(sequence bias, fragment length estimation\)](#)



Can't figure something out? Questions, comments, concerns, or other feedback:
cbenner@ucsd.edu



HOMER

Software for motif discovery and ChIP-Seq analysis

This is the old version of the documentation: [New Version](#)

ChIP-Seq Analysis: Step 2, Basic Quality Control and Parameter Estimation

There are a few things that need to be checked whenever running a new experiment. There are several things that can go wrong when performing high-throughput sequencing, both at the initial steps (i.e. chromatin immunoprecipitation) and the later steps (sample quantification, amplification, and cluster generation etc.). The following analyses are incorporated within the makeTagDirectory program as they should be performed on every data set (even RNA-Seq and other types of sequencing) to get a feel for what was actually sequenced.

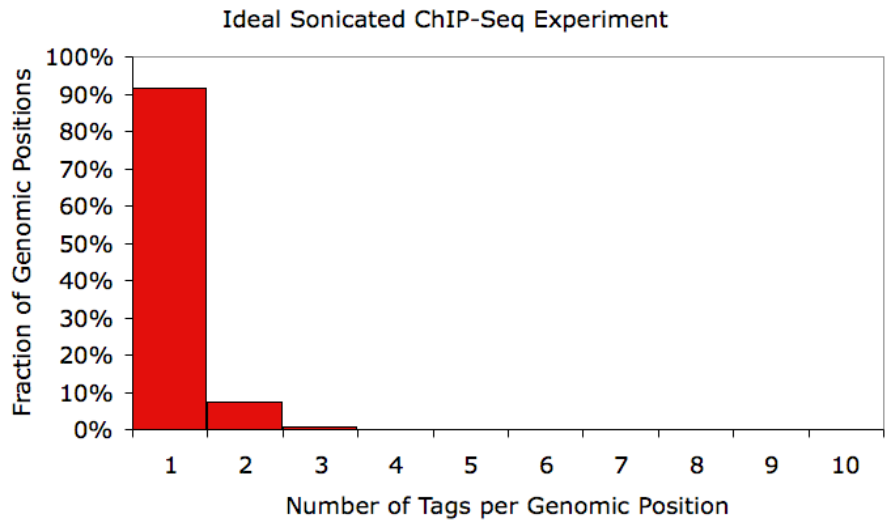
- [Clonal Tag Counts](#)
- [Sequencing Fragment Length Estimation \(tag autocorrelation\)](#)
- [Checking for Sequence Bias](#)

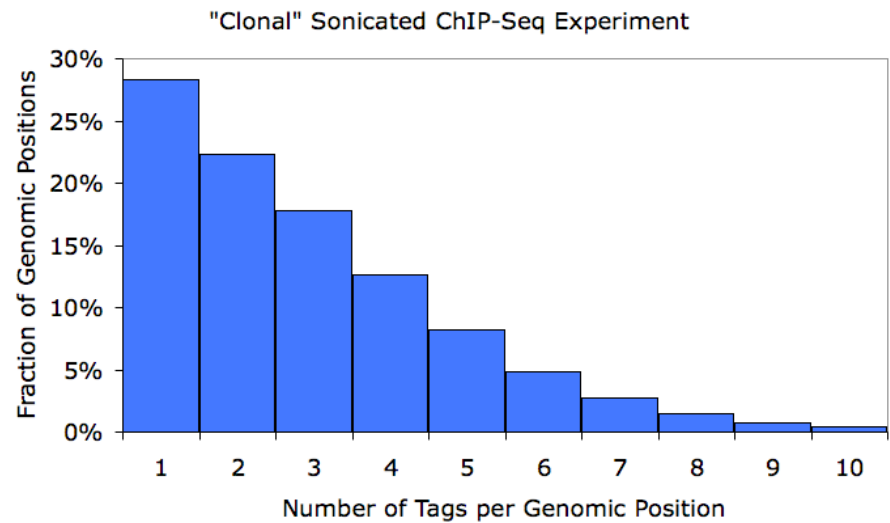
Clonal Tag Counts

Based on the protocol for high-throughput sequencing, it is possible to end up sequencing the "same fragments" over and over again. This happens when the amount of starting material (i.e. IP DNA) is relatively small. After amplification and cluster generation, several clusters can result from the same single piece of DNA that was found in the initial IP.

In the case of sonicated ChIP samples, the DNA should fragment in a somewhat random pattern, making it relatively rare that a fragment starts in exactly the same location in the genome. It is true that in regions with high ChIP enrichment, this is unavoidable, but if most fragments in the sample have several reads in the same positions, this is a serious problem and indicative of clonal reads. Since you are effectively sequencing the same reads over and over again, you are no longer getting new information when sequencing more reads. For RNA-Seq or ChIP-Seq using restriction enzymes this may be expected, but for sonication experiments it is highly unlikely this would happen by chance.

To check for this, **makeTagDirectory** creates a file named **tagCountDistribution.txt** in the tag directory, which contains the distribution of duplicated tag positions. Graphing results from this file in EXCEL, I show an example of an ideal experiment (<1.2 average tags per position) vs. a clonal experiment (~3 tags per position):





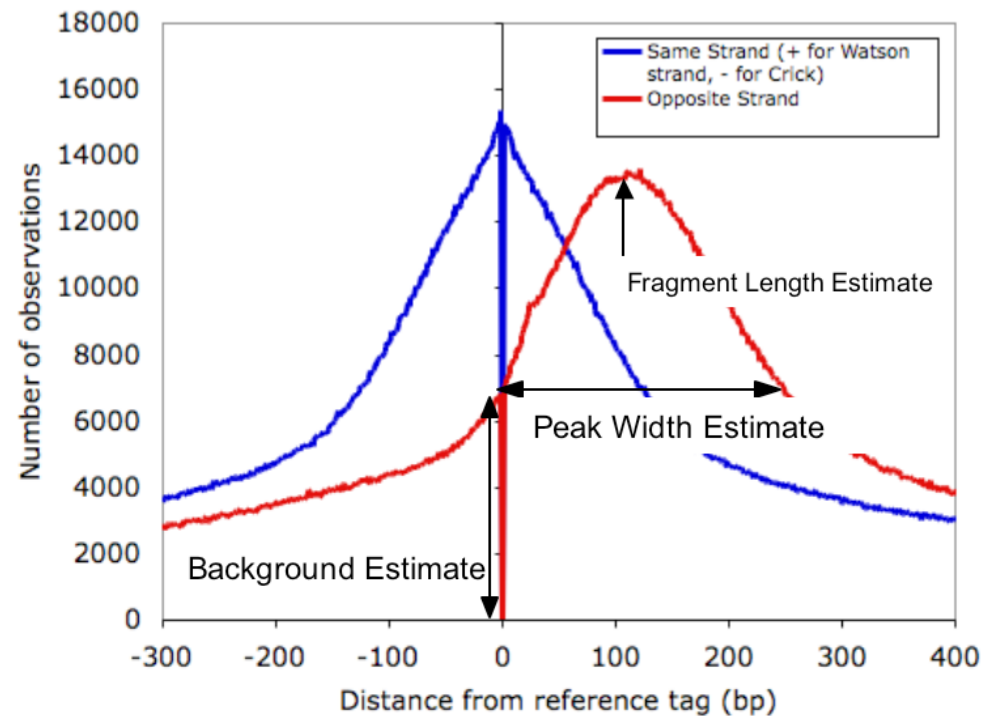
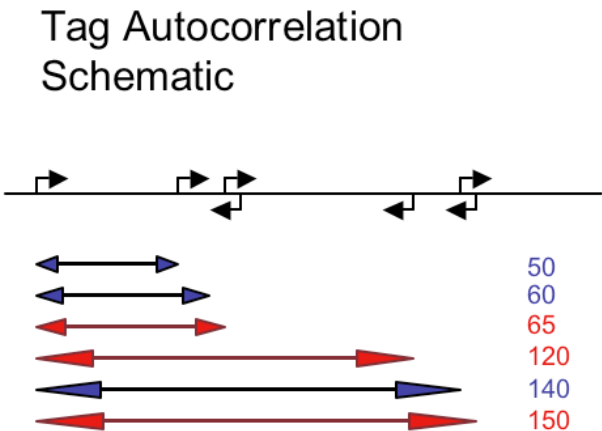
If an experiment looks like the bottom example, it's a good idea to redo the ChIP and re-prep the sample for sequencing. This doesn't mean that useful information cannot still be gleaned from the sample, but it does mean that you are only getting a fraction of the information out that you are paying for. As a rule of thumb, if your **averageTagsPerPosition** parameter in the **tagInfo.txt** file is less than 1.5, you are in good shape. Again, this may not be an appropriate metric for RNA-Seq or enzymatically digested ChIP-Seq (i.e. MNase).

Sequencing Fragment Length Estimation

ChIP works by fragmenting crosslinked chromatin so that only pieces of DNA bound by a particular protein are purified. As a result, and due to the requirements of DNA fragment length for efficient sequencing by next-generation sequencers, only DNA fragments of a particular size are used for ChIP-Seq. This often involves cutting out a band of a specific size or range of sizes from a gel. The specific size of fragments sequenced for a given experiment can be very important in extracting meaningful data and precisely determining the location of binding sites.

To estimate the length of fragments used for sequencing, we perform an autocorrelation analysis of tag positions. This is done by calculating the distance from each tag to every other tag along the same chromosome and making a histogram of these distances, keeping track of which tags are found on the same strand or separate strands. This is shown below:

ChIP-Seq Tag Autocorrelation (Esrrb, ES cells)

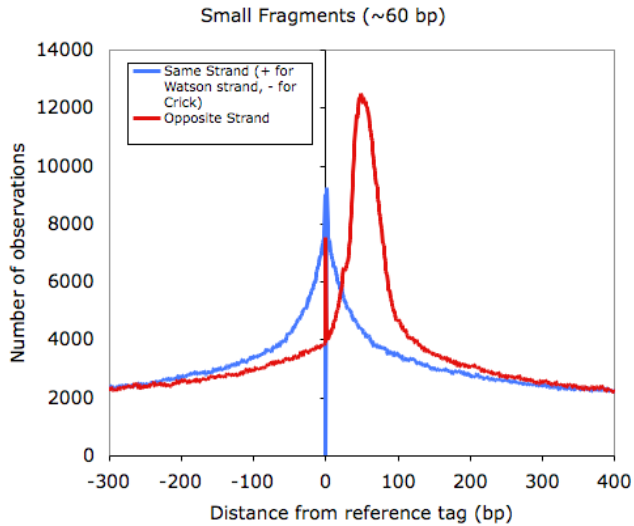


The data to generate this plot is found in the **tagAutocorrelation.txt** file found in the tag directory. Simply open it with EXCEL and make an x-y plot with the data.
NOTE: to change the parameters used to generate the tag autocorrelation, you can reanalyze the experiment using the **autoCorrelateTags.pl** tool.

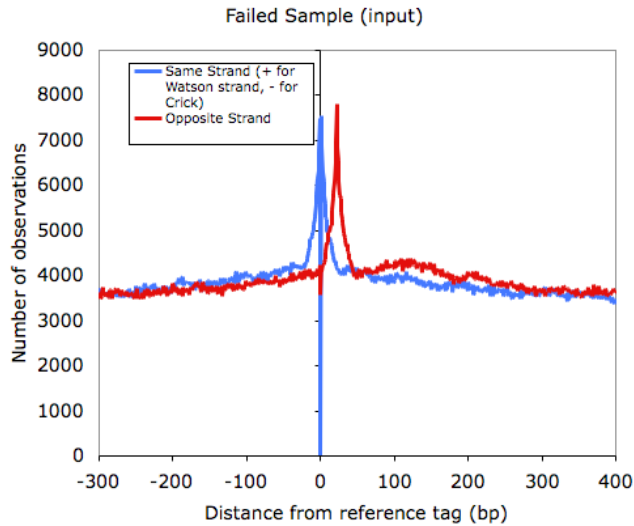
The tag autocorrelation is a nice piece of information for several reasons. First, you can usually tell right off the bat if your experiment worked at all. If there are enriched regions, then you should expect tags to be found in the vicinity of one another. If the autocorrelation plot looks flat, or extremely spiky, that's a good sign something might have gone wrong with the experiment.

The other primary piece of information we extract from the autocorrelation is an estimate of the fragment length used for sequencing. This estimate is derived from the maximum in autocorrelation signal in opposite strand tags found downstream of the reference tag (crest in the red signal above). We can further estimate the relative range in fragment sizes by considering when the opposite strand distribution falls below its corresponding level at the reference tag (i.e. at zero). Since tags found on opposite strands, facing away from each other cannot measure the same protein bound to the DNA, the level of opposite strand autocorrelation found at the origin provides an estimate of the background of the ChIP-Seq signal. This value is then used to estimate the peak size (for transcription factors or other proteins that make contact at specific positions). These values are stored in the **tagInfo.txt** file for automatic look up later.

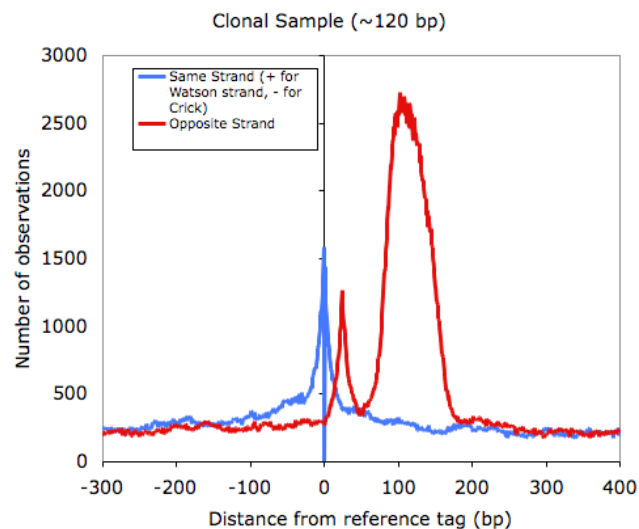
Other examples of tag auto correlations can be found below:



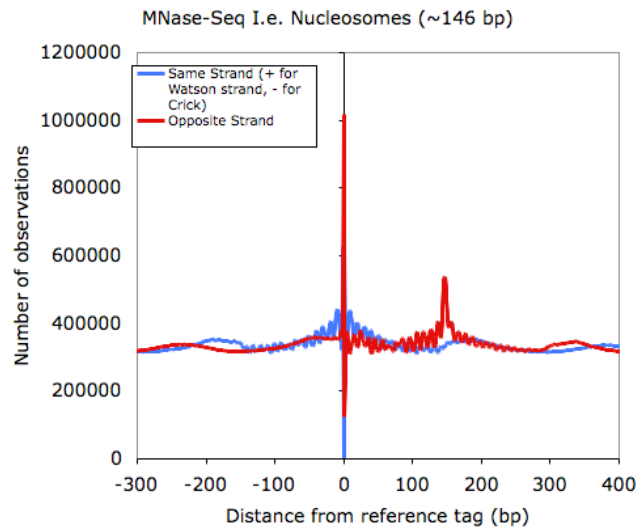
One thing I'll point out is that as the quality of samples decrease, the signal from tags mapping as palindromes increases. For example, the input (mock IP) control sample below shows a strong spike at 23 bp, which corresponds to the length of sequence used to map that sample.



If you sample is clonal (i.e. from the section above), it is common that you will see a high enrichment of opposite strand autocorrelation since you are essentially sequencing both ends of every fragment.



And for fun, because Chuck loves bioinformatics, you can do this with MNase-Seq data (which hints at how nucleosomes are organized)



Checking Sequence Bias

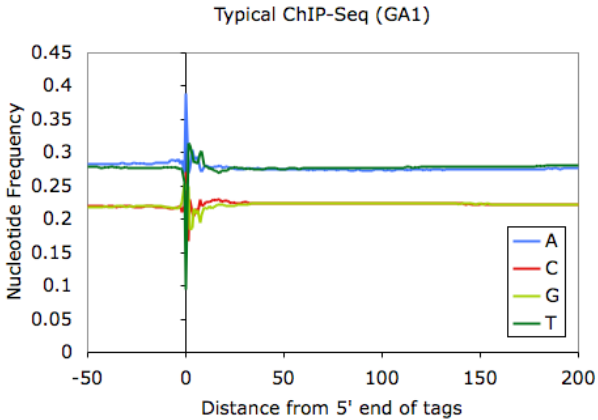
HOMER checks for bias in your sequencing data by aligning every tag in its genomic context and calculating the nucleotide frequency at each position relative to the 5' end of the tag. This can be useful for identifying bias in sequences produced by your experimental protocol, or sequencing issues, or problems with preferential sequencing of GC-rich vs. AT-rich regions.

For now, HOMER does **NOT** automatically analyzed sequence bias as part of the **makeTagDirectory** program (although this will change in the near future). For now, this is handled by the program **checkTagBias.pl**. The important parts of this program are written in C++ so it's fairly fast, even for large data sets. To analyze your sequence bias for a tag directory, type the following:

```
checkTagBias.pl <tag directory> <genome>
```

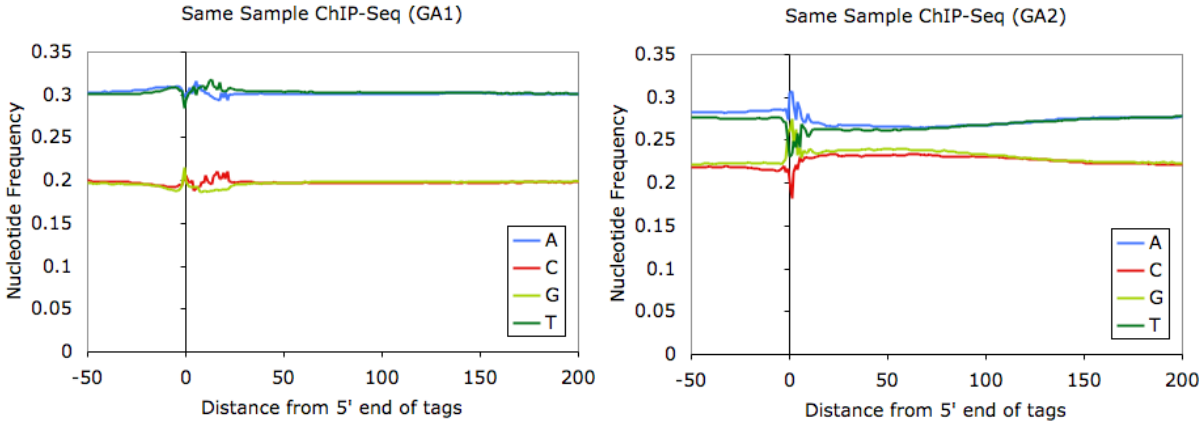
i.e. **checkTagBias.pl Macrophage-PU.1 mm8**

This requires that <genome> (in this case mm8) is loaded by homer (for more info click [HERE](#)). This command will produce a file named **tagFreq.txt** in the tag directory. Opening this file with EXCEL and graphing the first 5 columns will produce something like this:

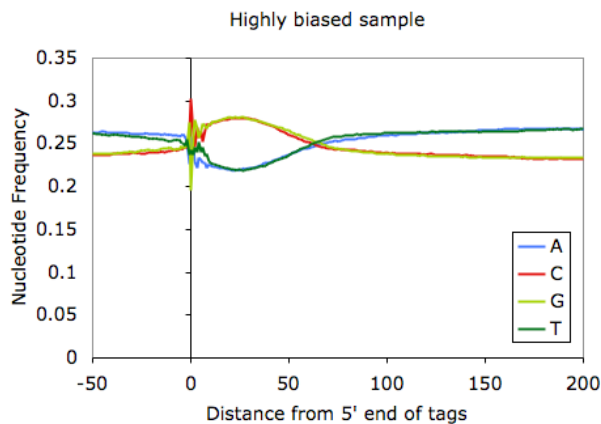


Although it may depend heavily on what is sequenced, most ChIP-Seq samples should look something like the above example. It is very typical to have strong sequence preferences right at the 5' edge of the tag, where sequence preferences introduced by the ligation of sequencing adapters will enrich for fragments with specific sequences. Variation within the first ~30 bp is normally indicative of bias from the base calling of the sequencer.

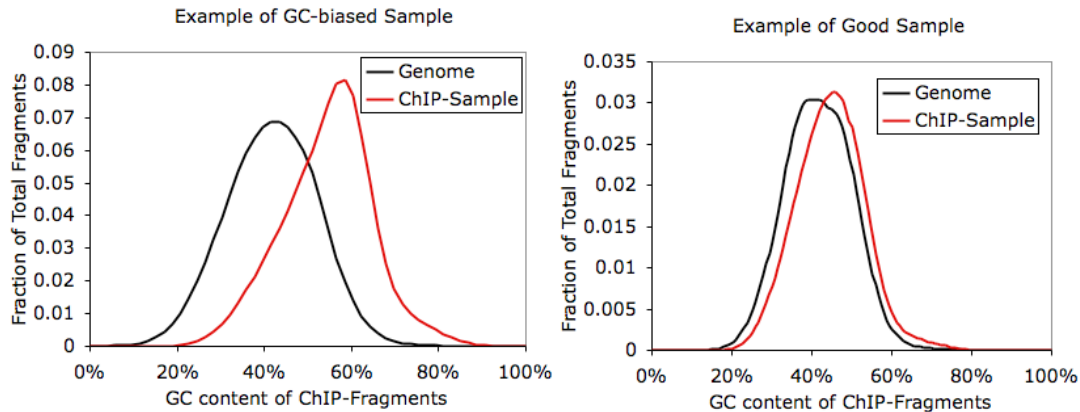
One observation worth noting is that sequencing performed on the Illumina GA2 (vs. GA1) tends to select for *fragments* that have higher overall GC-content. For example (using the same sample):



This unfortunately can bias where you might find peaks. In extreme cases where experimental mishaps may have contributed, the GC-bias can be very extreme (see below):

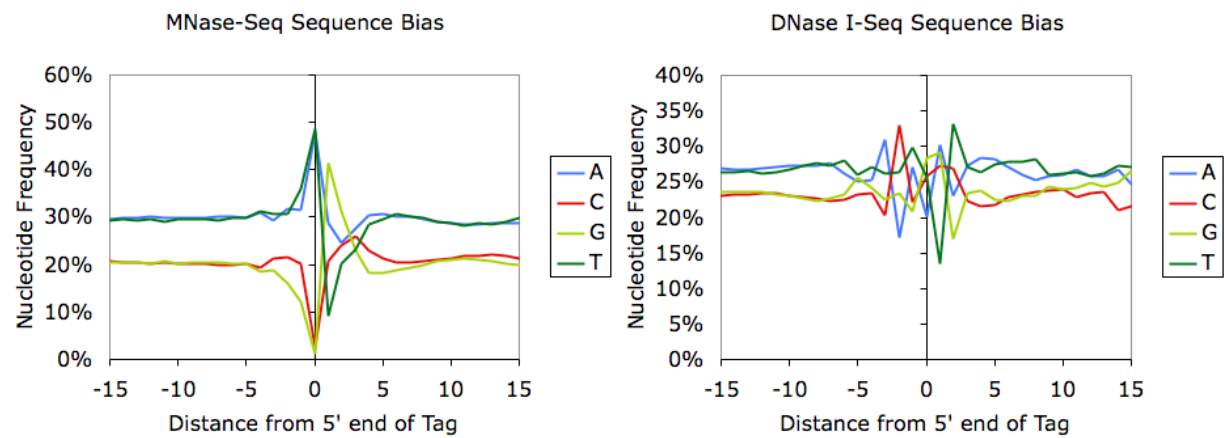


In this case you're basically screwed. Since in most cases the fragments you pull down in the ChIP experiment are background anyway, you shouldn't see a major shift in GC-content in your ChIP-Seq fragments. In the case above, you can also essentially approximate the length of the ChIP-Seq fragments (~55 bp according to the autocorrelation analysis). Another way to check for GC-bias related problems is to check the files named **tagGCcontent.txt** and **tagCpGcontent.txt** in the tag directory after running **checkTagBias.pl**. These files contain data describing the distribution of GC and CpG content for each fragment from sequencing. Below is an example corresponding to the experiment shown above:

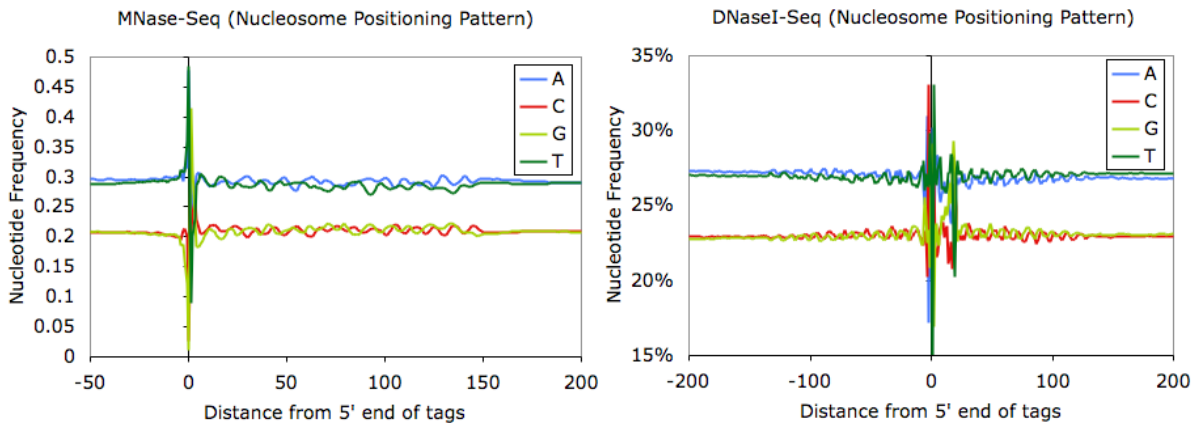


Obviously, if your sequencing looks like this, your peak finding will basically be blind to peaks in AT-rich regions of the genome. In this case, the experiment needs to be repeated.

Sequencing bias can also be used for discovery; for example, the sequencing bias can reveal sequence preferences for restriction enzymes. Examination of MNase- and DNaseI-digested DNA reveals the sequences that are preferentially cleaved by each enzyme:



And if you ask Chuck, he'd tell you that you should zoom out on the MNase and DNase tag bias plots to reveal sequences that help position nucleosomes:



Next: [Creating files to view your data in the UCSC Genome Browser](#)



Can't figure something out? Questions, comments, concerns, or other feedback:
cbenner@ucsd.edu



HOMER

Software for motif discovery and ChIP-Seq analysis

This is the old version of the documentation: [New Version](#)

ChIP-Seq Analysis: Visualizing Experiments with the UCSC Genome Browser

NOTE: Recent upgrades by the UCSC Genome Browser may cause files made with HOMER v2.3 or earlier to produce errors. HOMER v2.4 fixes this.

The [UCSC Genome Browser](#) is quite possibly one of the best computational tools ever developed. Not only does it contain an incredible amount of data in a single application, it allows users to upload custom information such as data from their ChIP-Seq experiments so that they can be easily visualized.

The basic strategy HOMER uses is to create a bedGraph formatted file that can then be uploaded as a custom track to the genome browser. This is accomplished using the **makeUCSCfile** program. To make a ucsc visualization file, type the following:

makeUCSCfile <tag directory> -o auto

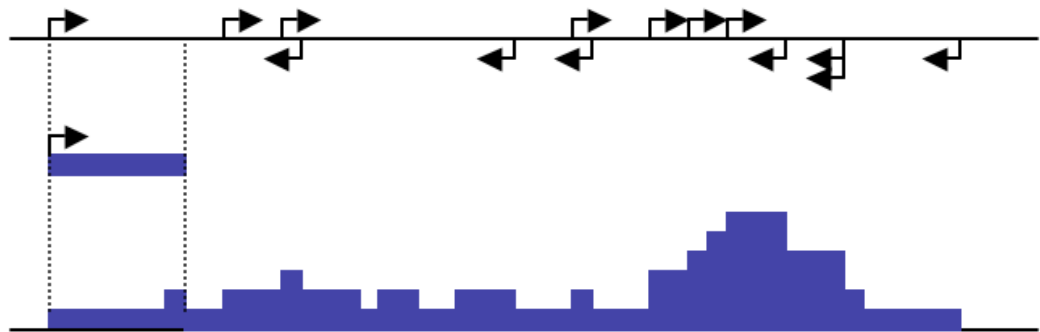
i.e. **makeUCSCfile PU.1-ChIP-Seq/ -o auto**
(output file will be in the PU.1-ChIP-Seq/ folder named PU.1-ChIP-Seq.ucsc.bedGraph.gz)

The "-o auto" with make the program automatically generate an output file name (i.e. TagDirectory.ucsc.bedGraph.gz) and place it in the tag directory which helps with the organization of all these files. The output file can be named differently by specifying "-o outputfilename" or by simply omitting "-o", which will send the output of the program to *stdout* (i.e. add " > outputfile" to capture it in the file outputfile). It is recommended that you zip the file using **gzip** and directly upload the *zipped* file when loading custom tracks at UCSC.

To visualize the ChIP-Seq experiment in the UCSC Genome Browser, go to Genome Browser [page](#) and select the appropriate genome (i.e. the genome that the sequencing tags were mapped to). Then click on the "add custom tracks" button (this will read "manage custom tracks" once at least one custom track is loaded). Enter the file created earlier in the "Paste URLs or data" section and click "Submit".

What does makeUCSCfile do?

The program works by approximating the ChIP-fragment density at each position in the genome. This is done by starting with each tag and extending it by the estimated fragment length (determined by [tag auto correlation](#), or it can be manually specified using "-fragLength <#>"). The ChIP-fragment density is then defined as the total number of overlapping fragments at each position in the genome. Below is a diagram that depicts how this works:



As great as the UCSC Genome Browser is, the large size of recent ChIP-Seq experiments results in custom track files that are very large. In addition to taking a long time to upload, the genome browser has trouble loading excessively large files. To help cope with this, the **makeUCSCfile** program works by specifying a target file size when zipped (default 50MB). In order to meet the specified target file size, **makeUCSCfile** merges adjacent regions of tag density levels by their weighted average to reduce the total number lines in the final bedGraph file. If you have trouble loading getting your file to load, try reducing the size of the file using the "-fsize <#>" option (i.e. "-fsize 2e7"). To force the creation of larger files, use a very large file size (i.e. "-fsize 1e50") - this will create a file that does not merge any regions and displays a "native" view of the data.

Tags can be visualized separately for each strand using the "-strand separate" option.

Changing the Resolution

In an effort to reduce the size of large UCSC files, one attractive option is to reduce the overall resolution of the file. By default, **makeUCSCfile** will make full resolution (i.e. 1 bp) files, but this can be changed by specifying the "-res <#>" option. For example, "-res 10" will cause changes in ChIP-fragment density to be reported only every 10 bp.

Normalization of UCSC files

In order to easily compare ChIP-fragment densities between different experiments, **makeUCSCfile** will normalize density profiles based on the total number of mapped tags for each experiment. As with other programs apart of HOMER, the total number of tags is normalized to 10 million. This means that tags from an experiment with only 5 million mapped tags will count for 2 tags apiece. The total of tags to normalize to can be changed using the "-norm <#>" option.

Separating data from different strand (i.e. good for RNA-Seq data)

You can specify that HOMER separates the data based on the strand by using the "-strand <...>" option. The following options are available:

- "-strand both" : default behavior for ChIP-Seq
- "-strand separate" : separate data by strand
- "-strand +" : only show the positive strand (i.e. watson strand) data
- "-strand -" : only show the negative strand (i.e. crick strand) data

Creating bigWig files with HOMER

Some data sets are very large, but you still want to see all of the details from your sequencing in the UCSC Genome Browser. HOMER can also produce [bigWig files](#) by running the conversion program for you (**bedGraphToBigWig**). The only catch is that **you must have access**

[to a webserver](#) where you can post the resulting bigWig file - this is because instead of uploading the whole file to UCSC, the browser actually looks for the data file on YOUR webserver and grabs only the parts it needs. Slick, eh. Chuck uses this all the time for big experiments.

To use this feature, you must download the [bedGraphToBigWig program from UCSC](#) and place it somewhere in your executable path (i.e. the /path-to-homer/bin/ folder). To make a bigWig, add the "**-bigWig -fsize 1e20**" parameters to your makeUCSCfile command. When making a bigWig, you usually want to see all of the tag information, so make sure the "-fsize" options is large. You also need to specify an output file using "**-o <bigwigfilename>**" and also capture the stdout stream using "**> trackfileoutput.txt**". You can also use "**-o auto**". The "trackfileoutput.txt" will contain the header information that is uploaded as a costum track to UCSC.

After running the makeUCSCfile program with the bigWig options, you need to do the following:

1. Copy the *.bigWig file to your webserver location and make sure it is viewable over the internet.
2. Need to edit the "trackfileoutput.txt" file and enter the URL of your bigWig file (... bigDataUrl=http://server/path/bigWigFilename ...)
3. Upload the "trackfileoutput.txt" file to UCSC as a costum track to view your data.

For example:

```
makeUCSCfile <tag directory> -o auto -bigWig -fsize 1e20 > trackInfo.txt
```

i.e.

```
makeUCSCfile PU.1-ChIP-Seq/ -o auto -bigWig -fsize 1e20 > PU.1-bigWig.trackInfo.txt  
cp PU.1-ChIP-Seq/PU.1-ChIP-Seq.ucsc.bigWig /Web/Server/Root/Path/  
** edit PU.1-bigWig.trackInfo.txt to have the right URL **
```

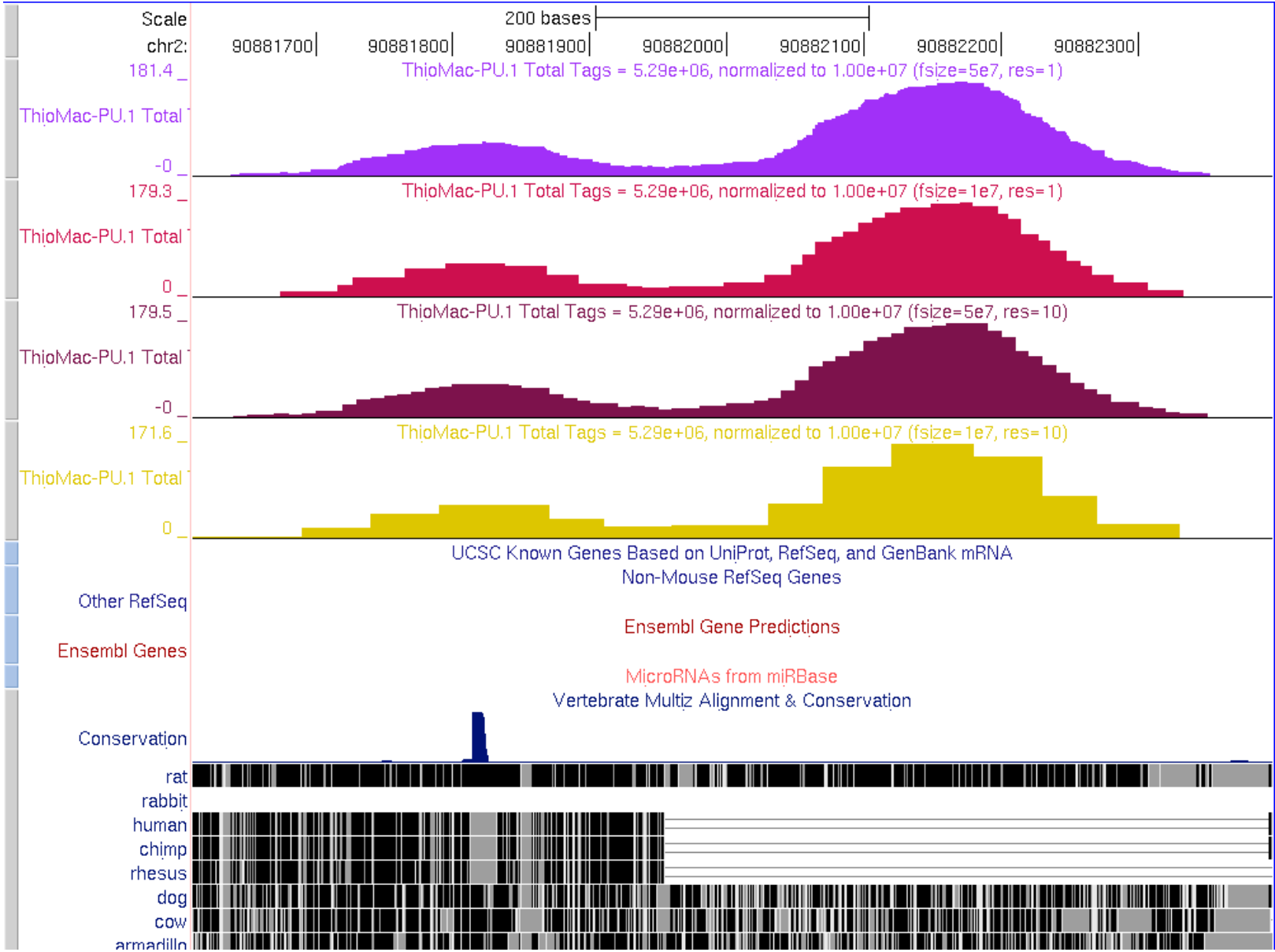
NOTE: As of now, a bigWig file can only be composed of a single track - if you want to separate the data by strands, do the following:

```
makeUCSCfile PU.1-ChIP-Seq/ -o PU.1.positiveStrand.bigWig -bigWig -fsize 1e20 -strand + > PU.1-  
bigWig.trackInfo.positiveStrand.txt  
makeUCSCfile PU.1-ChIP-Seq/ -o PU.1.negativeStrand.bigWig -bigWig -fsize 1e20 -strand - > PU.1-  
bigWig.trackInfo.negativeStrand.txt  
cp PU.1.positiveStrand.bigWig PU.1.negativeStrand.bigWig /Web/Server/Root/Path/  
cat PU.1-bigWig.trackInfo.positiveStrand.txt PU.1-bigWig.trackInfo.negativeStrand.txt > PU.1-bigWig.trackInfo.both.txt  
** edit PU.1-bigWig.trackInfo.both.txt to have the right URLs for both the negative and positive strands **
```

Examples of UCSC bedGraph files

The following shows what the same data set looks like changing options for file size (-fsize) and resolution (-res). Usually it's best to use one or the other.

1. -fsize 5e7 -res 1
2. -fsize 1e7 -res 1
3. -fsize 5e7 -res 10
4. -fsize 1e7 -res 10



Command line options for makeUCSCfile

Usage: makeUCSCfile <tag directory> [options]

Creates a bedgraph file for visualization using the UCSC Genome Browser

General Options:

- fsize <#> (Size of file, when gzipped, default: 5e7)
- strand <both|separate> (control if reads are separated by strand, default: both)
- fragLength <#|auto> (Approximate fragment length, default: auto)
- res <#> (Resolution, in bp, of file, default: 1)
- norm <#> (Total number of tags to normalize experiment to, default: 1e7)
- color <(0-255),(0-255),(0-255)> (no spaces, rgb color for UCSC track, default: random)
- bigWig (creates a full resolution bigWig file and track line file)
This requires bedGraphToBigWig to be available
- o <filename|auto> (send output to this file - will be gzipped, default: prints to stdout)
auto: this will place an appropriately named file in the tag directory

Next: [Finding Peaks \(ChIP-enriched regions\) in the genome](#)



Can't figure something out? Questions, comments, concerns, or other feedback:
cbenner@ucsd.edu



HOMER

Software for motif discovery and ChIP-Seq analysis

This is the old version of the documentation: [New Version](#)

ChIP-Seq Analysis: Finding Peaks (ChIP-enriched Regions)

Finding peaks is one of the central goals of any ChIP-Seq experiment. The basic idea is to identify regions in the genome where we find more ChIP-Seq tags than we would expect to see by chance. There are number of different approaches one can use to find ChIP-Seq peaks, and correspondingly there are many different methods for identifying peaks from ChIP-Seq experiments. It is not required that you use HOMER for peak finding to use the rest of the tools included in HOMER ([see below](#)).

The most important distinction between HOMER and several other peak finding algorithms is that by default HOMER searches for peaks of **FIXED SIZE**. This is ideal for achieving maximum sensitivity when finding peaks for Transcription Factors or other proteins that make contact at a single point along the DNA. Peaks at these locations should contain nearly identical tag distributions (i.e. since ChIP-fragments are only ~150 bp, binding to a specific site should not (in principle) produce peaks wider than the fragment length in either direction).

The default settings in HOMER is not particularly good at identifying "enriched blocks" which are typical of repressive histone modifications such as H3K9me3 and H3K27me3. HOMER will find regions that are enriched regardless of shape and size, but if your research requires that you accurately identify the boundaries of large enrichment blocks of varying size, try using the option "**-region**". (See [Finding Regions of Variable Length](#)) Histone marks that are more "focal", such as H3K4me1, H3K4me2, H3K4me3 or H3/H4 acetylation are easily found with HOMER when using a larger peak size (i.e. -size 1000)

The other reason HOMER uses fixed size peaks is that the assumptions needed for Motif Finding are, let's say, *less violated*, when dealing with regions of constant size.

Finding Transcription Factor Peaks with HOMER

To find peaks for a transcription factor use the **findPeaks** command:

findPeaks <tag directory> -center -o <output peak file>

i.e. findPeaks ERalpha-ChIP-Seq/ -center -o ERalpha.peaks.txt

Where the first argument must be the tag directory (required). The "**-center**" option ensures the the peak is place on the region of maximum ChIP-fragment density, and is recommended when analyzing ChIP-Seq for transcription factors. It is highly recommended that you develop a strategy for keeping track of these files. For example, it's usually nice to place peak files in the tag directory they are associated with so that they are easy to find later. For example:

findPeaks ERalpha-ChIP-Seq/ -center -o ERalpha-ChIP-Seq/peaks.txt

Homer attempts to automatically determine parameters needed for peak finding - most importantly the size of the peak and the length of ChIP-fragments (determined from the [tag auto correlation](#), stored as parameters in the <tag directory>/tagInfo.txt file). These can be overwritten using the "**-size <#>**" and "**-fragLength <#>**" options. You can also set the minimum distance between peaks using "**-minDist <#>**" (default is 2.5x the peak size). Strand specific peak finding is also available ("**-strand separate**").

Be default, HOMER assumes an effective genome size of 2 billion (fine for human or mouse). This can be explicitly set using the "**-gsize <#>**" option. HOMER determines the threshold of tags needed to call a peak significant by assuming that non-enriched ChIP-fragment concentrations are approximated by the [Poisson distribution](#), which is similar to just about every other method out there. The cutoff for statistically significant peaks can be specified by "**-fdr <#>**" or

with "-poisson <#>".

Output: the Peak file

HOMER outputs peak information in the form of an text file which is easily opened with EXCEL. A typical peak file will look something like this:

<	A	B	C	D	E	F	G	H	I	J	K	L	M	
1	#	Peak finding parameters:												
2	#	tag directory = pu1/												
3	#													
4	#	total peaks = 32922												
5	#	peak size = 243												
6	#	peaks found using tags on both strands												
7	#	minimum distance between peaks = 607												
8	#	fragment length = 142												
9	#	genome size = 2000000000												
10	#	Total tags = 5293530.0												
11	#	Total tags in peaks = 1008630.0												
12	#	Approximate IP efficiency = 19.05%												
13	#	tags per bp = 0.002372												
14	#	expected tags per peak = 0.577												
15	#	maximum tags considered per bp = 1.0												
16	#	FDR rate threshold = 0.001000												
17	#	FDR tag threshold = 7.0												
18	#	number of putative peaks = 49373												
19	#													
20	#	input tag directory = input/												
21	#	Fold over input required = 0.00												
22	#	Putative peaks filtered by input = 15198												
23	#													
24	#	size of region used for local filtering = 10000												
25	#	Fold over local region required = 0.00												
26	#	Putative peaks filtered by local signal = 725												
27	#													
28	#	Maximum fold under expected unique positions for tags = 2.00												
29	#	Putative peaks filtered for being too clonal = 528												
30	#													
31	#	cmd = findPeaks pu1/ -i input/ -F 0.0000001 -L 0.001												
32	#													
33	#	Column Headers:												
34	#PeakID	chr	start	end	strand	score	focusRatio(i	Total Tags	Control Tag	Fold Chang	Local Fold	C Clonal Fold	Change/p-value	
35	chr4-1	chr4	72475073	72475316	+	346	0	1498.8	0.6	8.33E-10	1.17E-07	0.72		
36	chr8-1	chr8	72525008	72525251	+	272	0	2024.4	0.6	8.33E-10	1.89E-08	0.91		
37	chr9-1	chr9	48310730	48310973	+	185	0	380.7	3	2.69E-08	4.37E-08	1.05		
38	chr6-1	chr6	89867048	89867291	+	181	0	294.8	0.6	8.33E-10	1.37E-07	0.96		
39	chr3-1	chr3	89997184	89997427	+	180	0	249.8	0.6	8.33E-10	1.01E-07	0.88		
40	chr2-1	chr2	1.68E+08	1.68E+08	+	178	0	263.8	0.6	8.33E-10	1.47E-07	0.92		
41	chr17-2	chr17	82390408	82390651	+	176	0	1497.8	0.6	8.33E-10	2.72E-08	1.38		
42	chr1-1	chr1	1.82E+08	1.82E+08	+	172	0	327.7	0.6	8.33E-10	1.25E-07	1.07		
43	chr2-2	chr2	1.32E+08	1.32E+08	+	170	0	270.8	2	1.09E-08	1.05E-07	0.97		
44	chr12-1	chr12	56307901	56308144	+	162	0	286.8	2	1.09E-08	4.20E-08	1.05		
45	chr2-3	chr2	50664027	50664270	+	161	0	322.7	3	2.69E-08	4.66E-08	1.12		
46	chr9-3	chr9	1.2E+08	1.2E+08	+	161	0	239.8	2	1.09E-08	5.06E-08	0.97		
47	chr2-4	chr2	1.57E+08	1.57E+08	+	160	0	267.8	0.6	8.33E-10	7.43E-08	1.02		
48	chr3-2	chr3	1.07E+08	1.07E+08	+	159	0	284.8	0.6	8.33E-10	1.05E-07	1.06		
49	chr4-2	chr4	98716855	98717098	+	159	0	300.8	1	2.31E-09	3.12E-08	1.1		
50	chr6-2	chr6	52772190	52772433	+	151	0	282.8	3	2.69E-08	3.02E-08	1.12		
51	chr3-3	chr3	1.05E+08	1.05E+08	+	150	0	198.8	0.6	8.33E-10	6.75E-08	0.93		

The first several rows start with a "#", which contains meta information about the peak finding procedure. Below this information are the peaks, listed in each row. Columns contain information about each peak:

- Column 1: PeakID - a unique name for each peak (very important that peaks have unique names...)
- Column 2: chr - chromosome where peak is located
- Column 3: starting position of peak

- Column 4: ending position of peak
- Column 5: Strand (+/-)
- Column 6: Score - number of tags found clustered together (may differ from total tag count depending on how many tags were considered per bp)
- Column 7: Focus Ratio - fraction of tags found appropriately upstream and downstream of the peak center. Describes the chances that factor binds a single spot on the DNA. (will be set to zero if -center was NOT used)
- Columns 8+: Statistics and Data from filtering

Two generic tools are available as part of HOMER to convert peak files to BED files and back. This will allow you to upload your peak files to the UCSC Genome Browser, or convert peak files in BED format from another program into a peak file that can be used by HOMER. These programs are named **pos2bed.pl** and **bed2pos.pl**, which can be used the following way:

```
pos2bed.pl peakfile.txt > peakfile.bed
bed2pos.pl peakfile.bed > peakfile.txt
```

How findPeaks works

findPeaks initially works by analyzing tag positions as opposed to ChIP-fragment densities. This is partially done to keep the program flexible for various tasks. In order to analyze ChIP-Seq data for peaks, the first step is to "adjust" the positions of tags to the center of their ChIP-fragments so that tag densities can be more accurately calculated. This tag position adjustment is half of the estimated ChIP-fragment length in the 3' direction relative to the original position of the tag.

The program then sorts tags along each chromosome and efficiently looks at each tag position to calculate the number tags found within the designated peak size. It then sorts tag positions from those with the highest density of tags through those with the lowest, assigning putative peaks and masking nearby positions within the minimum distance acceptable for neighboring peaks (-minDist). Positions that are masked essentially represent positions near a better candidate peak, and as the program encounters masked positions, it further masked positions near positions that were already masked to ensure peaks identified come from local maxima. After cycling through all tag positions, a putative list of peaks is generated representing local maximum in tag distributions. These are subjected to additional filters based on input sequencing, etc.

Filtering Peaks

The initial step of peak finding is to find non-random clusters of tags, but in many cases these clusters may not be representative of try transcription factor binding events. To increase the overall quality of peaks identified by HOMER, 3 separate filtering steps can be applied to the initial peaks identified:

Using Input/IgG Sequencing as a Control

To use an Input or IgG sequencing run as a control (**HIGHLY RECOMMENDED**), you must first create a separate tag directory for the input experiment ([see here](#)). Additionally, you can use other cleaver experiments as a control, such as a ChIP-Seq experiment for the same factor in another cell or in a knockout. To find peaks using a control, type:

```
findPeaks <tag directory> -center -i <control tag directory> -o <output peak file>
```

```
i.e. findPeaks ERalpha-ChIP-Seq/ -i Input-ChIP-Seq/ -center -o ERalpha.peaks.txt
```

HOMER offers two ways to filter peaks against a control experiment. First, if you prefer a simple fold change, HOMER will require that each putative peak be greater than 4-fold (or specify with "-F <#>"). In the case where there are no input tags near the putative peak, HOMER automatically sets these regions to be set to the average input tag coverage to avoid dividing by zero. Alternatively, HOMER can adhere to more statistically rigorous measures of differential tag enrichment by using the poisson distribution to model peak similarity. To filter peaks based on their poisson p-value, use "-F <#>" but enter a number between 0 and 1 (i.e. "-F 0.0001").

Filtering Based on Local Signal

Our experience with peak finding is that often putative peaks are identified in regions of genomic duplication, or in regions where the reference genome likely differs from that of the genome being sequenced. This produces large regions of high tag counts, and if no Input/IgG sample is available, it can be

hard to exclude these regions. Also, it may be advantageous to remove putative peaks that are spread out over larger regions as it may be difficult to pinpoint the important regulatory regions within them.

To deal with this, HOMER will filter peaks based on the local tag counts (similar in principle to MACS). By default, HOMER requires the tag density at peaks to be 4-fold greater than in the surrounding 10 kb region. This can be modified using "-L <#>" and "-localSize <#>" to change the fold threshold and size of the local region, respectively. Alternatively, a poisson p-value threshold can be set using "-L <#>" between 0 and 1.

Filtering Based on Clonal Signal

When we first sifted through peaks identified in ChIP-Seq experiments we noticed there are many peaks near repeat elements that contain odd tag distributions. These appear to arise from expanded repeats that result in peaks with high numbers of tags from only a small number of unique positions, even when many of the other positions within the region may be "mappable". To help remove these peaks, HOMER will compare the number of unique positions containing tags in a peak relative to the expected number of unique positions given the total number of tags in the peak. If the ratio between the later and the former number gets too high, the peak is discarded. The fold threshold can be set with the "-C <#>" option (default: "-C 2"). HOMER uses the **averageTagsPerPosition** parameter in the **tagInfo.txt** file to adjust this calculation as to not over-penalize ChIP-Seq experiments that are already highly "[clonal](#)". If analyzing MNase or other restriction enzyme digestion experiments turn this option off ("-C 0");

Disabling Filtering

To disable Input, Local, or Clonal filtering set any combination of "-F 0 -L 0 -C 0".

Finding Peaks of fixed length using Histone Modification Data.

Finding peaks using histone modification data can be a little tricky - largely because we have very little idea what the histone marks actually do. If you want to find peaks in histone modification data with the purpose of analyzing them for enriched motifs, read this section. If you are trying to annotate regions of the genome that are covered by a given mark, read the next section. The problem with histone modification data (and some other types) is that the signal can spread over large distances. Trying to analyze large, variable length regions for motif enrichment is very difficult and not recommended.

There are two important differences between finding fixed width peaks for transcription factors and histone modifications. The first is the size of the peaks: Most histone modifications should be analyzed using a peak size in the range of 500-2000 usually (i.e. "-size 1000"). The other is that you should omit the "-center" option. Since you are looking at a region, you do not necessarily want to center the peak on the specific position with the highest tag density, which may be at the edge of the region. Besides, in the case of histone modifications at enhancers, the highest signal will usually be found on nucleosomes surrounding the center of the enhancer, which is where the functional sequences and transcription factor binding sites reside. Typically, adding the -center option moves peaks further away from the functional sequence in these scenarios. An example for finding peaks:

```
findPeaks H3K4me1-ChIP-Seq/ -i Input-ChIP-Seq -size 1000 -o H3K4me1-ChIP-Seq/peaks.1kb.txt
```

One issue with finding histone modification peaks using the defaults in HOMER is that the Local filtering step removes several of the peaks due to the "spreading" nature of many histone modifications. This can be good and bad. If you are looking for nice concentrated regions of modified histones, the resulting peaks will be a nice set for further analysis such as motif finding. However, if you are looking to identify every region in the cell that has an appreciable amount of modified histone, you may want to disable local filtering, or consider using the "-region" option below e.g.:

```
findPeaks H3K4me1-ChIP-Seq/ -i Input-ChIP-Seq -size 1000 -L 0 -o H3K4me1-ChIP-Seq/peaks.1kb.txt
```

Also, if using MNase-treated chromatin (i.e. nucleosome mapping), you may want to use "-C 0" to avoid filtering peaks composed of highly clonal tag positions. These can arise from well positioned nucleosomes (or sites that are nicely digested by MNase)

Finding Enriched Regions of Variable Length

If the option "-region" is specified, **findPeaks** will stitch together enriched peaks into regions. Note that [local filtering](#) is turned off when finding regions. The most important parameters for region finding are the "-size" and "-minDist" (and of course the fragment length). First of all, "-size" specifies the width of peaks that will form the basic building blocks for extending peaks into regions. Smaller peak sizes offer better resolution, but larger peak sizes are usually more sensitive. By default **findPeaks** uses the estimated peak size from the autocorrelation analysis, although you may want to use a larger peak size for histone

modifications (i.e. 500 or 1000) where the minimum size of peaks observed are usually fairly large anyway.

The second parameter, "**-minDist**", is usually used to specify the minimum distance between adjacent peaks. If "**-region**" is used, this parameter then specifies the maximum distance between putative peaks that is allowed if they are to be stitched together to form a region. By default this is 2.5x the peak size. If you think about histone modifications, the signal is never continuous in enriched regions, with reduced signal due to non-unique sequences (that can't be mapped to) and nucleosome depleted regions. "-minDist" informs findPeaks how big of a gap in the signal will be tolerated for adjacent peaks to be considered part of the same region.

One thing to note is that you may have to play around with these parameters to get the results you want. If you look at the examples below, you could make arguments for using each of the tracks given what you're interested in and how you would define a "region".

For example: (in the example below, the default size comes from the autocorrelation estimate for the Macrophage-H3K4me1 dataset)

Default Parameters:

```
findPeaks Macrophage-H3K4me1/ -i Macrophage-IgG -size 150 -minDist 370 > output.txt (i.e. defaults)
```

Recommend Parameters for fixed width peaks (i.e. for motif finding):

```
findPeaks Macrophage-H3K4me1/ -i Macrophage-IgG -size 1000 -minDist 2500 > output.txt
```

Default Parameters for variable length peaks.

```
findPeaks Macrophage-H3K4me1/ -i Macrophage-IgG -region -size 150 -minDist 370 > output.txt
```

Effect on variable length peaks if we increase minDist to 1000.

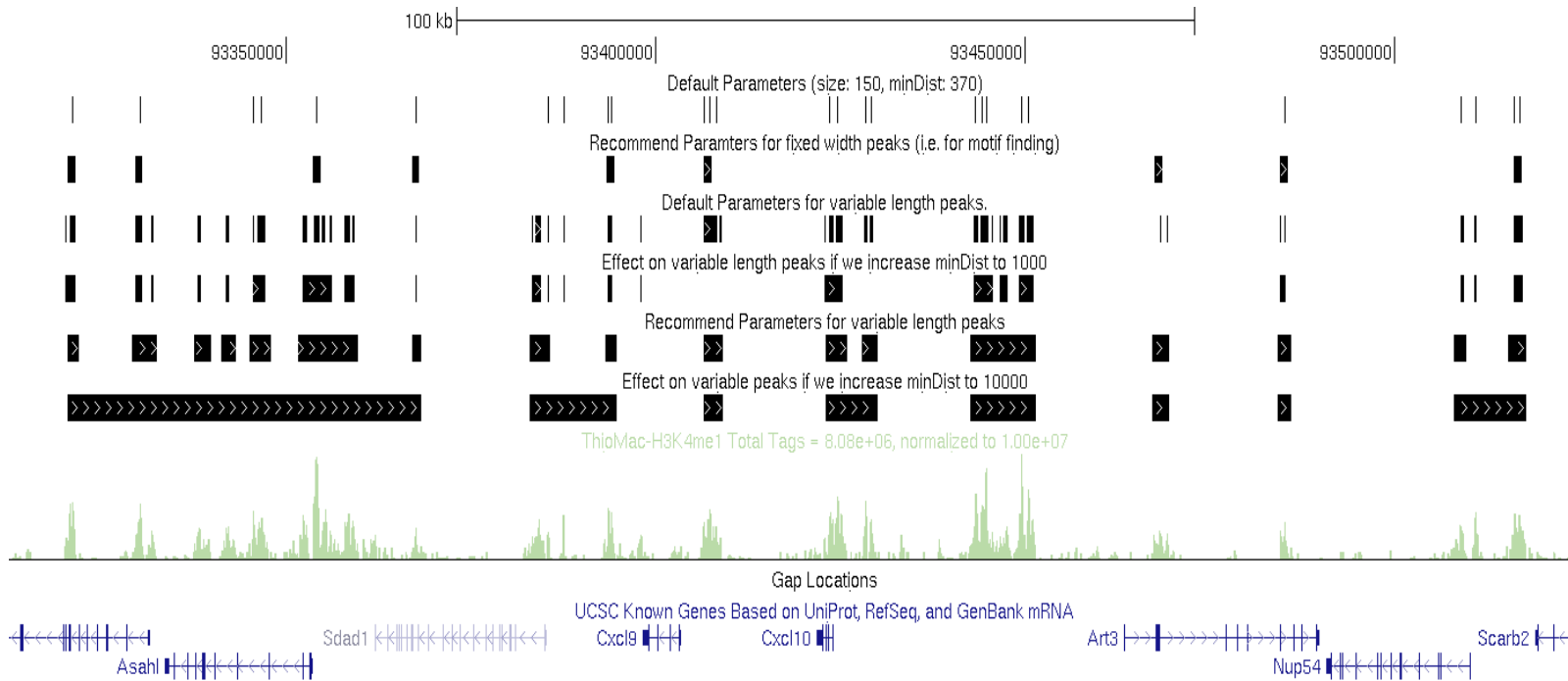
```
findPeaks Macrophage-H3K4me1/ -i Macrophage-IgG -region -size 150 -minDist 1000 > output.txt
```

Recommend Parameters for variable length peaks (H3K4me1 at least).

```
findPeaks Macrophage-H3K4me1/ -i Macrophage-IgG -region -size 1000 -minDist 2500 > output.txt
```

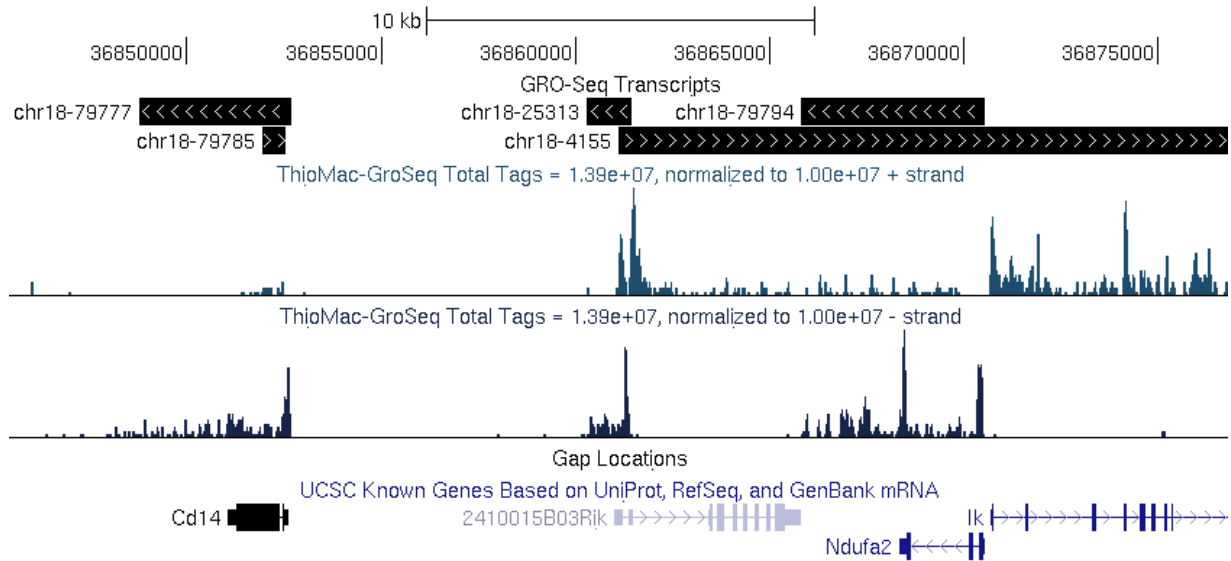
Effect on variable peaks if we increase minDist to 10000 (H3K4me1 at least).

```
findPeaks Macrophage-H3K4me1/ -i Macrophage-IgG -region -size 1000 -minDist 10000 > output.txt
```

findPeaks can also be used to define transcripts/exons from RNA-Seq and it's variants. The only catch is that HOMER won't figure out the splicing. However, for methods such as GRO-Seq (global nuclear run-on sequencing) this can be useful for getting a sense for what the transcripts are since we don't have to worry about introns in this type of data. If you sequencing is strand specific, make sure to use **"-strand specific"**. Also, it's recommended to disable clonal filtering when analyzing RNA (**"-C 0"**).

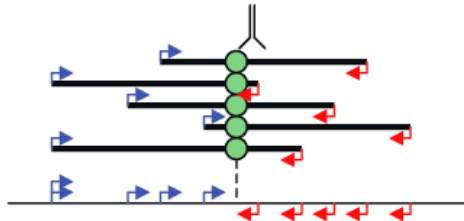
```
findPeaks GRO-Seq -strand separate -size 200 -fragLength 100 -minDist 2000 -region -C 0 > transcripts.txt
```



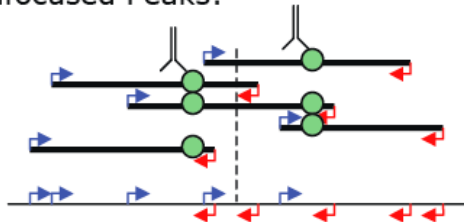
Peak Centering and Focus Ratios

If the option **"-center"** is specified, **findPeaks** will calculate the position within the peak with the maximum ChIP-fragment overlap and calculate a **focusRatio** for the peak. This is not always desired (such as with histone modifications). The focus ratio is defined as the ratio of tags located 5' of the peak center on either strand relative to the total number of tags in the peak. Peaks that contain tags in the ideal positions are more likely to be centered on a single binding site, and these peaks can be used to help determine what sequences are directly bound by a transcription factor. Unfocused peaks, or peaks with low (i.e. <80%) focusRatios may be the result of several closely spaced binding sites or large complexes that crosslink to multiple positions along the DNA.

Focused Peaks:



Unfocused Peaks:



To isolate focused peaks, you can use the `getFocalPeaks.pl` tool:

```
getFocalPeaks.pl <peak file> <focus % threshold> > focalPeaksOutput.txt
```

i.e. `getFocalPeaks.pl ERpeaks.txt 0.90 > ERfocalPeaks.txt`

Peak finding and Sequencing Saturation

HOMER does not try to estimate sequencing saturation, which is the estimate of whether or not you have sequenced enough tags to identify all the peaks in a given experiment. Generally speaking, if you sequence more, you will get more peaks since your sensitivity will increase. The only real way to assess this is if you can somehow show that all of the "functional" or "real" peaks have high tag counts (i.e. are well above the threshold for identifying a peak), meaning that sequencing more is not likely to identify more "real" peaks. This generally cannot be determined by simply resampling the data and repeating the peak finding procedure - you need some sort of outside information to assess peak quality, such as motif enrichment or something else. Simply resampling will likely only tell you if you've gotten to the point where you are simply resequencing the same fragments again - i.e. becoming clonal.

Command line options for findPeaks

Usage: `findPeaks <tag directory> [options]`

Finds peaks in the provided tag directory. By default, peak list printed to stdout

General Options:

- o <output file> (default: stdout)
- i <input tag directory> (Experiment to use as IgG/Input/Control)
- size <#> (Peak size, default: auto)
- minDist <#> (minimum distance between peaks, default: peak size x2.5)
- gsize <#> (Set effective genome size, default: 2e9)
- fragLength <#|auto> (Approximate fragment length, default: auto)
- inputFragLength <#|auto> (Approximate fragment length of input tags, default: auto)
- tbp <#> (Maximum tags per bp to count, 0 = no limit, default: auto)
- inputtbp <#> (Maximum tags per bp to count in input, 0 = no limit, default: auto)
- strand <both|separate> (find peaks using tags on both strands or separate, default:both)
- norm # (Tag count to normalize to, default 10000000)

Peak localization/shape options:

- center (Centers peaks on maximum tag overlap and calculates focus ratios)
- region (extends start/stop coordinates to cover full region considered "enriched")

Peak Filtering options: (set -F/-L/-C to 0 to skip)

- F <#> (fold enrichment over input tag count, default: 4.0)
if set between 0 and 1, poisson p-value cutoff will be used, i.e. 0.001
- L <#> (fold enrichment over local tag count, default: 4.0)
if set between 0 and 1, poisson p-value cutoff will be used, i.e. 0.001
- C <#> (fold enrichment limit of expected unique tag positions, default: 2.0)
- localSize <#> (region to check for local tag enrichment, default: 10000)
- fdr <#> (False discovery rate, default = 0.001)
- poisson <#> (Set poisson p-value cutoff, default: uses fdr)
- tagThreshold <#> (Set # of tags to define a peak, default: uses fdr)

Links to alternative peak finding software

Lots of quality programs exist for finding peaks in ChIP-Seq data. Most use slightly different assumptions and peak definitions that result in slightly different sets of peaks. Many of these programs output peak files in BED format. To convert these to HOMER peak file format, use the `bed2pos.pl` program to convert the file:

`bed2pos.pl peaks.bed > peaks.txt`

Peak finding software links:

- [MACS](#) (Liu Lab)
- [ChIPSeq Peak Finder](#) (Wold Lab)
- [CCAT](#) (Good for histone modifications)
- [FindPeaks](#)
- Probably hundreds of others: Google it!

Next: [Finding Motifs in ChIP-Seq Peaks](#)



Can't figure something out? Questions, comments, concerns, or other feedback:
cbenner@ucsd.edu



HOMER

Software for motif discovery and ChIP-Seq analysis

This is the old version of the documentation: [New Version](#)

ChIP-Seq Analysis: Finding Enriched Motifs in ChIP-Seq Peaks

HOMER was initially developed to automate the process of finding enriched motifs in ChIP-Seq peaks. More generally, HOMER analyzes genomic positions, not limited to only ChIP-Seq peaks, for enriched motifs. The main idea is that all the user really needs is a file containing genomic coordinates (i.e. [peak file](#)), and HOMER will generally take care of the rest. To analyze a peak file for motifs, run the following command:

findMotifsGenome.pl <peak file> <genome> <output directory> [options]

i.e. **findMotifsGenome.pl ERpeaks.txt hg18r ER_MotifOutput/ -len 8,10,12**

A variety of output files will be placed in the <output directory>, including html pages showing the results.

The findMotifsGenome.pl program is a wrapper that helps set up the data for analysis using the HOMER motif discovery algorithm. By default this will perform *de novo* motif discovery as well as check the enrichment of known motifs. If you have not done so already, please look over [this page](#) describing how HOMER analyzes sequences for enriched motifs.

An important prerequisite for analyzing genomic motifs is that the appropriate genome [must be configured for use with HOMER](#).

Acceptable Input files

findMotifsGenome.pl accepts files in HOMER's "peak file" format. The minimum file requirements are as follows (separated by TABs):

- Column1: Unique Peak ID
- Column2: chromosome
- Column3: starting position
- Column4: ending position
- Column5: Strand (+/- or 0/1, where 0="+", 1="-")

Additional columns will be ignored. If starting with a BED file, convert it to a peak

file using the **bed2pos.pl** program. If using a EXCEL, make sure to save files as a "Text (Windows)" if running MacOS. If errors occur, it is likely that the file is not in the correct format, or the first column is not actually populated with unique identifiers.

!!! COMMON PROBLEM: If this program isn't working, make sure you save your peak files as "text (windows)" from EXCEL when on a Mac. Run the **checkPeakFile.pl** program to see if your file is the correct format, and use **changeNewLine.pl** if you didn't save your file in "text (windows)" format.

!!! OK - an even MORE common PROBLEM, particularly if you use a different peak finding program, make SURE you use UNIQUE peak IDs. If you think about it, the point of having a peak ID is so that you can tell them apart, so having duplicates is a horrible idea!!! Repeated peak IDs will cause older versions of HOMER to crash!! The program **renamePeaks.pl** is now included to rename the peaks if you're you need help with this.

Important motif finding parameters

Region Size ("**-size <#>**") - this specifies the size (centered on the peak centers) to look for motifs. I'd recommend 50 bp for establishing the primary motif bound by a given transcription factory, 200 bp for finding "co-enriched" motifs for a transcription factor, and 1000 bp for searching H3K4me or H3/H4 acetylated regions.

Motif length ("**-len <#>**" or "**-len <#>,<#>,...**") - specify the length of motifs to be found. HOMER will find motifs of each size separately and then combine the results at the end. The length of time it takes to find motifs increases greatly with increasing size. In general, it's best to try out enrichment with shorter lengths (i.e. 8 and/or 10) before trying longer lengths (i.e. 12 or 14). Much longer motifs can be found with HOMER, but it's best to use smaller sets of sequence when trying to find long motifs (i.e. use "**-len 20 -size 50**"), otherwise it may take way too long (or take too much memory).

Number of motifs to find ("**-S <#>**") - specifies the number of motifs of each length to find. (recommend "**-S 50**" or "**-S 100**").

Normalize GC% content instead of CpG% content ("**-gc**"), or disable GC/CpG normalization ("**-noweight**").

Use custom background regions ("**-bg <peak file of background regions>**") - these will still be normalized for CpG% or GC% content just like randomly chosen sequences

By default, **findMotifsGenome.pl** uses the binomial distribution to score motifs. This works well when the number of background sequences greatly out number the target sequences - however, if you are using "**-bg**" option above, and the

number of background sequences is smaller than target sequences, it is a good idea to use the hypergeometric distribution instead ("**-h**").

Find enrichment of individual oligos ("**-oligo**"). This creates output files in the output directory named *oligo.length.txt*.

Force findMotifsGenome.pl to re-prepare genome for the given region size ("**-prepare**").

How findMotifsGenome.pl works

There are a series of steps that the program goes through to find quality motifs:

1. Extract sequences from the genome corresponding to the peaks in the input file
2. Removes sequences with >70% Ns
3. Calculate the CpG/GC content of input sequences
4. (If not done during a previous run) Prepare genome for control fragments of the specified size
5. Randomly select background sequences matching CpG characteristics of input sequences
6. Perform *de novo* motif finding
7. Generate output files for *de novo* motif finding
8. Check enrichment of known motifs
9. Generate output files for known motif enrichment

Interpreting motif finding results

The format of the output files generated by **findMotifsGenome.pl** are identical to those generated by the promoter-based version **findMotifs.pl** ([description](#)).

In general, when analyzing ChIP-Seq / ChIP-Chip peaks you should expect to see strong enrichment for a motif resembling the site recognized by the DNA binding domain of the factor you are studying. Enrichment p-values reported by HOMER should be very very significant (i.e. $< 1e-50$). If this is not the case, there is a strong possibility that the experiment may have failed in one way or another. For example, the peaks could be of low quality because the factor is not expressed very high.

[Practical Tips for Motif finding](#)

Command-line options for findMotifsGenome.pl

Program will find de novo and known motifs in regions in the genome

Usage: findMotifsGenome.pl <pos file> <genome> <output directory> [additional options]

Example: findMotifsGenome.pl peaks.txt mm8r peakAnalysis -size 200 -len 8

Basic options:

- bg <background position file> (genomic positions to be used as background, default=automatic)
- len <#>[,<#>,<#>...] (motif length, default=10) [NOTE: values greater 12 may cause the program to run out of memory - in these cases decrease the number of sequences analyzed (-N)]
- size <#> (fragment size to use for motif finding, default=200)
- S <#> (Number of motifs to optimize, default: 100)
- mis <#> (global optimization: searches for strings with # mismatches, default: 2)
- depth [low|med|high|allnight] (time spent on local optimization default: med)

Scanning sequence for motifs

- find <motif file> (This will cause the program to only scan for motifs)

Known Motif Options

- mcheck <motif file> (known motifs to check against de novo motifs, default: /bioinformatics/homer/data/knownTFs/all.motifs)
- mknown <motif file> (known motifs to check for enrichment, default: /bioinformatics/homer/data/knownTFs/known.motifs)

Sequence normalization options:

- tss (normalize based on distance from TSS)
- cgtss (normalize based on CpG content and distance from TSS)
- cg DEFAULT (normalize based on CpG content)
- noweight (no CG correction)

Advanced options:

- h (use hypergeometric for p-values, binomial is default)
- N <#> (Number of sequences to use for motif finding, default=max(50k, 2x input)
- noforce (will attempt to reuse sequence files etc. that are already in output directory)
- local <#> (use local background, # of equal size regions around peaks to use i.e. 2)
- gc (use GC% instead of CpG% for sequence content normalization [NOT WORKING...])
- noknown (don't search for known motif enrichment)
- nocheck (don't search for de novo vs. known motif similarity)
- nomotif (don't search for de novo motif enrichment)
- norevopp (don't search reverse strand for motifs)
- redundant <#> (Remove redundant sequences matching greater than # percent, i.e. -redundant 0.5)
- float (allow Homer to adjust the degeneracy threshold for known motifs to get best p-value[dangerous])
- mask <motif file1> [motif file 2]... (motifs to mask before motif finding)
- refine <motif file1> (motif to optimize)
- rand (randomize target and background sequences labels)
- ref <peak file> (use file for target and background - first argument is list of peak ids for targets)
- oligo (perform analysis of individual oligo enrichment)

- dumpFasta (Dump fasta files for target and background sequences for use with other programs)
- prepare (force new background files to be created)

Next: [Annotating Peaks](#)



Can't figure something out? Questions, comments, concerns, or other feedback:
cbenner@ucsd.edu



HOMER

Software for motif discovery and ChIP-Seq analysis

This is the old version of the documentation: [New Version](#)

ChIP-Seq Analysis: Annotating Peaks

Homer contains a useful, all-in-one program for performing peak annotation called **annotatePeaks.pl**. In addition to associating peaks with nearby genes, **annotatePeaks.pl** can perform Gene Ontology Analysis, associate peaks with gene expression data, calculate ChIP-Seq Tag densities from different experiments, and find motif occurrences in peaks. **annotatePeaks.pl** can also be used to create histograms and heatmaps.

!!! COMMON PROBLEM: If this program isn't working, make sure you save your peak files as "text (windows)" from EXCEL when on a Mac. Run the **checkPeakFile.pl** program to see if your file is the correct format, and use **changeNewLine.pl** if you didn't save your file in "text (windows)" format.

Basic usage:

annotatePeaks.pl <peak file> <genome> > <output file>

i.e. **annotatePeaks.pl** ERpeaks.txt hg18 > outputfile.txt

The first two argument, the <peak file> and <genome>, are required, and must be the first two arguments. Other optional command line arguments can be placed in any order after the first two. By default, **annotatePeaks.pl** prints the program output to *stdout*, which can be captured in a file by appending " > filename" to the command. With most uses of **annotatePeaks.pl**, an the output is a data table that is meant to be opened with EXCEL or similar program. An example of the output can be seen below:

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T
1	ClusterID	Chr	Start	End	Direction	Cluster Tag	FDR	Annotation	Conservatio	Distance to	Nearest Prc	PromoterID	Nearest Un	Nearest Re	Nearest En	Gene Name	Gene Alias	Gene Description		
2	chr4-1	chr4	72475073	72475316	+	346	0	intergenic	NA	416239	NM_001013	433719	Mm.34829	NM_001013	ENSMUSG0	Gm11487	OTTMUSG0	predicted gene 11487		
3	chr8-1	chr8	72525008	72525251	+	272	0	intergenic	NA	29035	NM_001045	234362	Mm.45848	NM_001045	ENSMUSG0	Zfp868	AI449175	zinc finger protein 868		
4	chr9-1	chr9	48310730	48310973	+	185	0	intergenic	NA	46193	NM_010924	18113	Mm.45854	NM_010924	ENSMUSG0	Nnmt	-	nicotinamide N-methyltransferase		
5	chr6-1	chr6	89867048	89867291	+	181	0	intergenic	NA	8391	NM_053227	113854	Mm.37715	NM_053227	ENSMUSG0	V1rb4	V1RA10	V1 vomeronasal 1 receptor, B4		
6	chr17-1	chr17	39451383	39451626	+	180	0	intergenic	NA	220522	NM_031190	18663	Mm.47833	NM_031190	ENSMUSG0	Pgk2	Pgk-2	Tcp- phosphoglycerate kinase 2		
7	chr3-1	chr3	89997184	89997427	+	180	0	intron (NM	NA	1784	NM_010555	16194	Mm.2856	NM_010555	ENSMUSG0	Il6ra	CD126	IL-6 interleukin 6 receptor, alpha		
8	chr2-1	chr2	1.68E+08	1.68E+08	+	178	0	intergenic	NA	-7113	NM_011201	19246	Mm.27791	NM_011201	ENSMUSG0	Ptpn1	PTP-1B	PTP protein tyrosine phosphatase, non-		
9	chr17-2	chr17	82390408	82390651	+	176	0	intergenic	NA	-733080	NM_134117	106522	Mm.31197	NM_001480	ENSMUSG0	Pkdcc	AI115348	protein kinase domain containing,		
10	chr1-1	chr1	1.82E+08	1.82E+08	+	172	0	intergenic	NA	-4872	NM_001126	19165	Mm.33085	NM_001126	ENSMUSG0	Psen2	AI266870	presenilin 2		
11	chr2-2	chr2	1.32E+08	1.32E+08	+	170	0	intron (NM	NA	50516	NM_018824	54338	Mm.10358	NM_018824	ENSMUSG0	Slc23a2	AI844736	solute carrier family 23 (nucleobase		
12	chr2-1	chr2	5.22079E1	5.220914E1	+	162	0	intergenic	NA	-9005	NM_011955	26443	Mm.20210	NM_011955	ENSMUSG0	Rargas	TATA	proteasome (prosome, proteasome)		

Description of Columns:

1. Peak ID
2. Chromosome
3. Peak start position
4. Peak end position
5. Strand

6. Peak Score
7. FDR/Peak Focus Ratio
8. Annotation (i.e. Exon, Intron, ...)
9. Conservation
10. Distance to nearest TSS
11. Nearest TSS: Native ID of annotation file
12. Nearest TSS: Entrez Gene ID
13. Nearest TSS: Unigene ID
14. Nearest TSS: RefSeq ID
15. Nearest TSS: Ensembl ID
16. Nearest TSS: Gene Symbol
17. Nearest TSS: Gene Aliases
18. Nearest TSS: Gene description
19. Additional columns depend on options selected when running the program.

As of now, basic annotation is based on alignments of RefSeq transcripts to the UCSC hosted genomes.

Adding Gene Expression Data

annotatePeaks.pl can add gene-specific information to peaks based on each peak's nearest annotated TSS. To add gene expression or other data types, first create a gene data file (tab delimited text file) where the first column contains gene identifiers, and the first row is a header describing the contents of each column. In principle, the contents of these columns doesn't matter. To add this information to the annotation result, use the "-gene <gene data file>".

```
annotation.pl <peak file> <genome> -gene <gene data file> > output.txt
```

For peaks that are near genes with associated data in the "gene data file", this data will be appended to the end of the row for each peak.

Gene Ontology Enrichment

annotatePeaks.pl can automate the discovery of functionally enriched biological annotations using the genes near ChIP-Seq peaks. Since this analysis will produce several files, you must specify an output directory with "-go <gene ontology output directory>". Alternatively, you can run gene ontology [directly on a list of gene IDs](#).

Calculating ChIP-Seq Tag Densities across different experiments

annotatePeaks.pl is a useful program for cross-referencing data from multiple experiments. In order to count the number of tags from different ChIP-Seq experiments, you must first [create tag directories](#) for each of these experiments. Once created, tag counts from these directories in the vicinity of your peaks can be added by specifying "-d <tag directory 1> <tag directory 2> ...". You can specify as many tag directories as you like. Tag totals for each directory will be placed in new columns starting on column 18. For example:

```
annotatePeaks.pl pu1peaks.txt mm8 -size 400 -d Macrophage-PU.1/ Bcell-PU.1/ > output.txt
```

output.txt, when opened in EXCEL, will look like this:

N	O	P	Q	R	S	T	U	V	W	X
est Refs	Nearest Ense	Gene Name	Gene Alias	Gene Descrip	Macrophage-PU.1/ Tag C	Bcell-PU.1/ Tag Count in 400 bp (8861792 Total, normalization factor = 1.13, effe				
0010133	ENSMUSG000	Gm11487	OTTMUSG000	(predicted ger	3268.14	8209.4				
0010455	ENSMUSG000	Zfp868	AI449175 AV	zinc finger pr	4919.21	11197.51				
025423	ENSMUSG000	1110059E24f	-	RIKEN cDNA 1	3.78	1005.44				
010924	ENSMUSG000	Nmrnt	-	nicotinamide	935.1	1751.34				
011412	ENSMUSG000	Slit3	Slit2	slit homolog	3.78	804.58				
077662	ENSMUSG000	Ctso	A330105D01	cathepsin O	5.67	920.81				
028810	ENSMUSG000	Rnd3	2610017M01	Rho family G	846.32	1663.32				
011518	ENSMUSG000	Syk	-	spleen tyrosi	627.18	1649.78				

HOMER automatically normalizes each directory by the total number of mapped tags such that each directory contains **10 million tags**. This total can be changed by specifying **"-norm <#>"** or by specifying **"-noadj"**, which will skip this normalization step.

The other important parameter when counting tags is to specify the size of the region you would like to count tags in with **"-size <#>"**. For example, **"-size 1000"** will count tags in the 1kb region centered on each peak, while **"-size 50"** will count tags in the 50 bp region centered on the peak (default is 200). The number of tags is not normalized by the size of the region.

One last thing to keep in mind is that in order to fairly count tags, HOMER will automatically center tags based on their estimated ChIP-fragment lengths. This is can be overridden by specifying a fixed ChIP-fragment length using **"-len <#>"**.

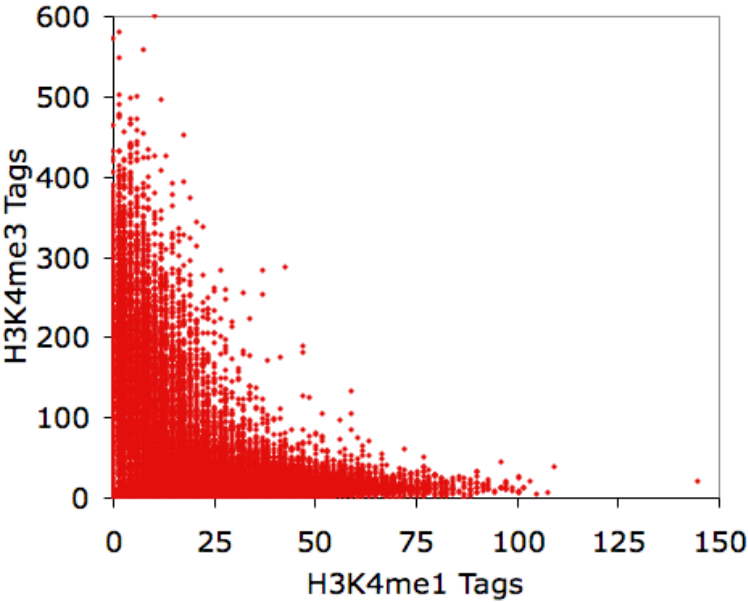
Making Scatter Plots

X-Y scatter plots are a great way to present information, and by counting tag densities from different tag directories, you can visualize the relative levels of different ChIP-Seq experiments. Because of the large range of values, it can be difficult to appreciate the relationship between data sets without log transforming the data (or sqrt to stay Poisson friendly). Also, due to the digital nature of tag counting, it can be hard to properly assess the data from a X-Y scatter plot since may of the data points will have the same values and overlap. To assist with these issues, you can specify **"-log"** or **"-sqrt"** to transform the data. These functions will actually report "log(value+1+rand)" and "sqrt(value+rand)", respectively, where rand is a random "fraction of a tag" that adds *jitter* to your data so that data points with low tag counts will not have exactly the same value. For example, lets look at the distribution of H3K4me1 and H3K4me3 near Oct4 peaks in mouse embryonic stem cells:

```
annotatePeaks.pl Oct4.peaks.txt mm8 -size 1000 -d H3K4me1-ChIP-Seq/ H3K4me3-ChIP-Seq/ > output.txt
```

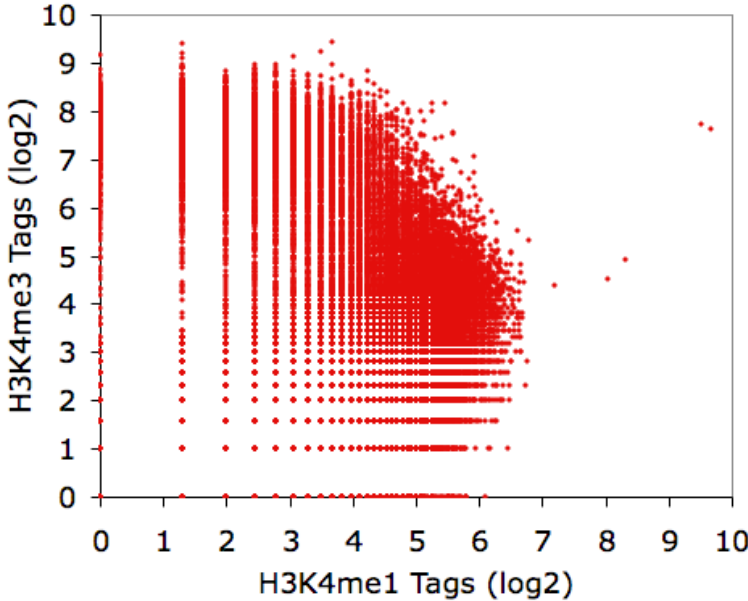
Opening output.txt with EXCEL and plotting the last two columns:

Tag Counts from 1kb around Oct4 Peaks in Stem Cells



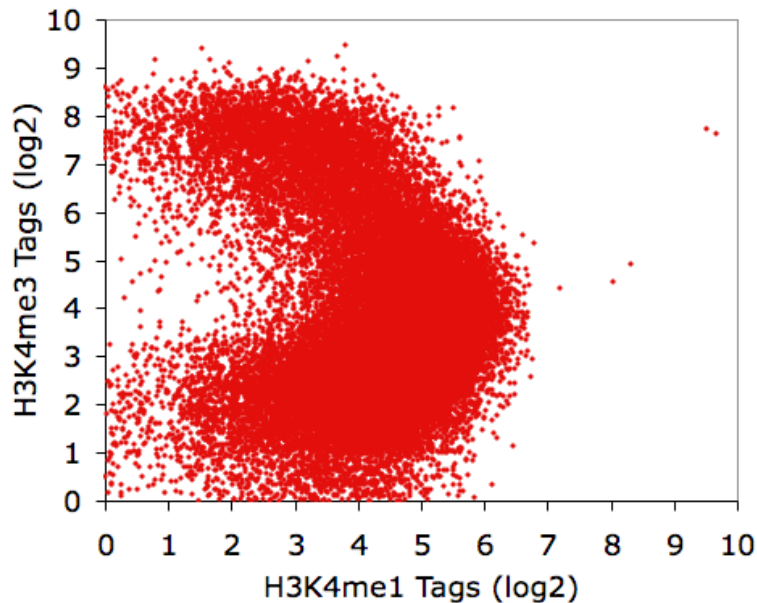
Using EXCEL to take the log(base 2) of the data:

Tag Counts from 1kb around Oct4 Peaks in Stem Cells



Now using the "-log" option:
`annotatePeaks.pl Oct4.peaks.txt mm8 -size 1000 -log -d H3K4me1-ChIP-Seq/ H3K4me3-ChIP-Seq/ > output.txt`

Tag Counts from 1kb around Oct4 Peaks in Stem Cells



Believe it or not, all of these X-Y plots show the same data. Interesting, eh?

Finding instances of motifs near peaks

Figuring out which peaks have instances of [motifs found with findMotifsGenome.pl](#) is very easy. Simply use `"-m <motif file1> <motif file 2>..."` with `annotatePeaks.pl`. (Motif files can be concatenated into a single file for ease of use) This will search for each of these motifs near each peak in your peak file. Use `"-size <#>"` to specify the size of the region around the peak center you wish to search. Found instance of each motif will be reported in additional columns of the output file. For example:

`annotatePeaks.pl pu1peaks.txt mm8 -size 200 -m pu1.motif cebp.motif > output.txt`

Opening output.txt with EXCEL:

O	P	Q	R	S	T	U	V
Ensembl Gene Name	Gene Alias	Gene Descrip	CpG%	GC%	PU.1/ThioMac-PU.1-ChIP-Seq/Homer Distance From Peak(sequence,strand,conservation)	CEBP/CEBPb-ChIP-Seq/Homer Dist	
3MUSG000 Dsn1	1700022L09f DSN1, MIND I		0.015	0.621891	-55(GGAGGAAGTG,+,0.00),-26(TGGGGAAGTG,+,0.00),35(GCAGGAAGTG,+,0.00)		
3MUSG000 Cd53	A1323659 O> CD53 antigen		0.01	0.432836	34(TGAGGAAGTG,+,0.00)		
3MUSG000 Atg4c	Apg4-C Apg4 autophagy-re		0.01	0.378109	8(AGAGGAAGTG,+,0.00),54(CACTTCCTCT,-,0.00)	-28(ATTTCATAAC,+,0.00)	
3MUSG000 Tax1bp1	1200003J11f Tax1 (human)		0.019608	0.307692			
3MUSG000 Plakho2	A1840380 MK pleckstrin ho		0.005	0.532338	-11(TGGGGAAGTG,+,0.00),33(TGAGGAAGTG,+,0.00)		
3MUSG000 Ctnnb2nl	AA552995 Ax CTTNBP2 N-t		0.01	0.462687	-97(GGAGGAAGTG,+,0.00),48(ACAGGAAGTG,+,0.00)		
3MUSG000 Pltp	OD107 phospholipid		0.005	0.467662	29(CAGTTCCTTT,-,0.00)		
3MUSG000 Ifngr2	Ifgr2 Ifgt interferon ga		0.01	0.462687	-24(CACTTCCTCA,-,0.00),49(CACTTCCTCA,-,0.00)		
3MUSG000 Aff2	Fmr2 Ox19 CAF4/FMR2 fai		0	0.512438			
3MUSG000 Syk	- spleen tyrosi		0.01	0.497512	40(AGGGGAAGTG,+,0.00),60(GGAGGAAGTG,+,0.00)		
3MUSG000 Exoc6	4833405E05f exocyst comp		0.015	0.432836	-41(CAGTTCCTCT,-,0.00)		
3MUSG000 Ccdc109b	9030408N13l coiled-coil do		0.005	0.482587	71(GGAGGAAGTG,+,0.00)	53(GTTGTGTAAG,-,0.00)	
3MUSG000 Ctnnd2	Catnd2 Nprajcatenin (cad		0.021858	0.472826	-54(AGCGGAAGTG,+,0.00),18(CAGTTCCTGT,-,0.00),34(CACTTCCTCT,-,0.00)	64(GTTTCATAAT,-,0.00)	
3MUSG000 Slc22a1	Slc22a1 Slc22a1 solute carrier		0.02	0.547264	-28(AGAGGAAGTG,+,0.00),-16(CACTTCCTCT,-,0.00)		

Each instance of the motif is specified in the following format (separated by commas):

Distance from Peak Center(Sequence Matching Motif,Strand,Average Conservation)

The average conservation will not be reported unless you specify `"-cons"`, and that will only work if conservation information has been configured (not available for all genomes). I wouldn't worry about this. Also, when finding motifs, the average CpG/GC content will

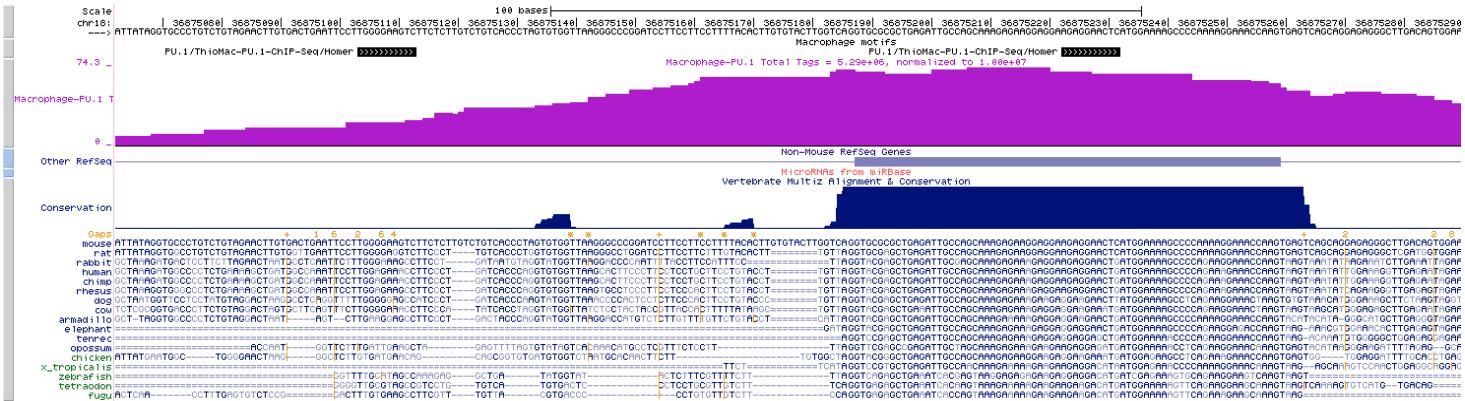
automatically be reported since it has to extract peak sequences from the genome anyway.

There are a bunch of motif specific options such as **"-norevopp"** (only search + strand relative to peak strand for motifs), **"-nmotifs"** (just report the total number of motifs per peak), **"-rmrevopp <#>"** (tries to avoid double counting reverse opposites within # bps).

Visualizing Motif positions in the UCSC Genome Browser

This feature may seem slightly out of place, but since annotatePeaks.pl is the workhorse of HOMER, you can add **"-mbed <filename>"** in conjunction with **"-m <motif file 1> [motif file 2] ..."** to produce a BED file describing motif positions near peaks that can be loaded as a custom track in the UCSC genome browser. In the example below, you would load motif.bed as a custom track:

```
annotatePeaks.pl pu1peaks.txt mm8 -size 200 -m pu1.motif cebp.motif -mbed motif.bed > output.txt
```



Finding the distance to other sets of Peaks

In order to find the nearest peak from another set of peaks, use **"-p <peak file 1> [peak file 2] ..."**. This will add columns to the output spreadsheet that will specify the nearest peak ID and the distance to that peak. If all you want is the distance (so you can sort this column), add the option **"-pdist"** to the command. Otherwise, if you prefer to count the number of peaks in the peak file found within the indicated regions (i.e. with **"-size <#>"**), add **"-pcount"**.

Command line options for annotatePeaks.pl

Usage: `annotatePeaks.pl <peak file | tss> <genome version> [additional options...]`

Available Genomes (required argument): (name,org,directory,default promoter set)

Peak vs. TSS mode:

If the first argument is "tss" (i.e. `annotatePeaks.pl tss mouse ...`) then a TSS centric analysis will be carried out. Tag counts and motifs will be found relative to the TSS. (no position file needed)

NOTE: The default TSS peak size is 4000 bp, i.e. +/- 2kb (change with -size option)
-list <gene id list> (subset of genes to perform analysis [unigene, gene id, accession, probe, etc.], default = all promoters)
-TSS <promoter set> (promoter definitions, default=genome default)
-cTSS <promoter position file i.e. peak file> (should be -2000 to +2000 centered on TSS,refseqIDs)

Available Promoter Sets (use with -TSS): (name,org,directory,genome,masked genome)

Primary Annotation Options:

- m <motif file 1> [motif file 2] ... (list of motifs to find in peaks)
 - mscore (reports the highest log-odds score within the peak)
 - nmotifs (reports the number of motifs per peak)
 - fm <motif file 1> [motif file 2] (list of motifs to filter from above)
 - rmrevopp <#> (do not double count reverse opposite motif occurrences, # bp window to look for them)
- d <tag directory 1> [tag directory 2] ... (list of experiment directories to show tag counts for) NOTE: -dfile <file> where file is a list of directories in first column
- p <peak file> [peak file 2] ... (to find nearest peaks)
 - pdist to report only distance (-pdist2 gives directional distance)
 - pcount to report number of peaks within region
- gene <data file> ... (Adds additional data to result based on the closest gene. This is useful for adding gene expression data. The file must have a header, and the first column must be a GeneID, Accession number, etc. If the peak cannot be mapped to data in the file then the entry will be left empty.)
- go <output directory> (perform GO analysis using genes near peaks)
- matrix <filename> (outputs a motif co-occurrence matrix)
- mbed <filename> (Output motif positions to a BED file to load at UCSC | -mpeak <filename>)

Annotation vs. Histogram mode:

- hist <bin size in bp> (i.e 1, 2, 5, 10, 20, 50, 100 etc.)

The -hist option can be used to generate histograms of position dependent features relative to the center of peaks. This is primarily meant to be used with -d and -m options to map distribution of motifs and ChIP-Seq tags. For ChIP-Seq peaks for a Transcription factor you might want to use the -center option (below) to center peaks on the known motif

NOTE: tss Mode is OK

Histogram Mode specific Options:

- nuc (calculated mononucleotide frequencies at each position,
 - Will report by default if extracting sequence for other purposes like motifs)
- di (calculated dinucleotide frequencies at each position)
- histNorm <#> (normalize the total tag count for each region to 1, where <#> is the minimum tag total per region - use to avoid tag spikes from low coverage)
- ghist - outputs profiles for each gene, for peak shape clustering - only uses first directory
- rm <#> (remove occurrences of same motif that occur within # bp)

Peak Centering: (other options are ignored)

- center <motif file> (This will re-center peaks on the specified motif, or remove peak if there is no motif in the peak. ONLY recentering will be performed, and all other options will be ignored. This will output a new peak file that can then be reanalyzed to reveal fine-grain structure in peaks (It is advised to use -size < 200) with this to keep peaks from moving too far (-mirror flips the position))
- multi (returns genomic positions of all sites instead of just the closest)

Advanced Options:

- log (output tag counts as randomized log2 values - for scatter plots)
- sqrt (output tag counts as randomized sqrt values - for scatter plots)
- size # (Peak size[from center of peak], default=inferred from peak file)
 - size #,# (i.e. -size -10,50 count tags from -10 bp to +50 bp from center)
 - size "given" (count tags etc. using the actual regions - for variable length regions)
- local # (size in bp to count tags as a local background (10-100x peak size recommended)
- len # (Fragment length, default=150)
- pc # (maximum number of tags to count per bp, default=0 [no maximum])
- cons (Retrieve conservation information for peaks/sites)
- CpG (Calculate CpG/GC content)
- ratio (process tag values as ratios - i.e. chip-seq, or mCpG/CpG)
- norevopp (do not search for motifs on the opposite strand [works with -center too])
- noadj (do not adjust the tag counts based on total tags sequenced)
- norm # (normalize tags to this tag count, default=1e7, 0=average tag count in all directories)
- pdist (only report distance to nearest peak using -p, not peak name)
- noann (skip annotation step)

[Back to ChIP-Seq Analysis](#)



Can't figure something out? Questions, comments, concerns, or other feedback:
cbenner@ucsd.edu



HOMER

Software for motif discovery and ChIP-Seq analysis

This is the old version of the documentation: [New Version](#)

ChIP-Seq Analysis: Analyzing a ChIP-Seq experiment with one command

Even I don't like typing the same commands over and over again. The following command performs the standard set of analysis commands so that you can do better things while your data is processed.

analyzeChIP-Seq.pl <Tag Directory> <genome> [general options] [-A | B | C | D sub-program options]

i.e. a common use: **analyzeChIP-Seq.pl Factor-ChIP-Seq/ hg18r -i Input-ChIP-Seq -focus -A factor_alignment_file.bed factor_alignment_file2.bed**

This command performs 4 separate tasks labeled as A,B,C & D:

- A. Runs **makeTagDirectory** to parse alignment files, set up the tag directory, and performs basic QC such as tag auto correlation and checks for sequence bias.
- B. Runs **makeUCSCfile** and **findPeaks** to generate UCSC Genome Browser files and peak files for the experiment.
- C. Runs **findMotifsGenome.pl** to determine enriched motifs in your ChIP-Seq peaks.
- D. Runs **annotatePeaks.pl** to generate an annotated peak file and performs GO analysis on genes found near the peaks.

As output, this program will create standard files in the "Tag Directory" including an "index.html" file that links you to each of the output files.

There are a couple of general options that can be used with **analyzeChIP-Seq.pl**:

- i <input tag directory> : "Tag directory" to use as a control for peak finding.
- size <#> : Force this peak size for analysis (default is "auto" for peak finding, 200 bp for motif analysis and 50 bp for focused peak analysis)
- focus : This will find enriched motifs in 90% focused peaks using only +/- 25 bp of sequence, useful for identifying the primary motif bound by the factor.
- enhancer : This will set the peak size to "-size 1000" and only perform motif analysis on peaks > 3kb from the TSS.

To specifically tailor the options used by the sub-programs, first enter the "general options" you want above, then enter **"-A"** followed by the options you want passed to the sub-programs. For example, the most used sub-program option I use is the following:

"-A s_1_eland_result.txt" - this passes the alignment file to the **makeTagDirectory** program so that it will be used to make the Tag Directory.

If you've already made a tag directory, no need to use the "-A blah blah" option. In similar fashion, to tell the motif finding to check motifs of length 10, 11, and 12, add **"-C -len 10,11,12"** to the end of the command.

[**Back to ChIP-Seq Analysis**](#)



Can't figure something out? Questions, comments, concerns, or other feedback:
cbenner@ucsd.edu



HOMER

Software for motif discovery and ChIP-Seq analysis

This is the old version of the documentation: [New Version](#)

ChIP-Seq Analysis: Finding Overlapping and Differentially Bound Peaks

HOMER provides a utility for comparing sets of peaks called **mergePeaks** (replacing mergePeaks.pl in v2.4). It's default behavior is to take two or more peak files and return a single peak file containing the unique peak positions from the original files. For example:

```
mergePeaks -d <maximum distance to merge> <peak file1> <peak file2>  
[peak file3] ... > newPeakFile.txt
```

The program will output a new peak file containing the merged peaks to stdout. Peaks within the distance in bp specified by "**-d <#>**" will be reported as the weighted average position (based on 6th column peak scores) between the peaks found within the common region (default=100 bp). The origin of the peaks is specified in the 7th column of the new peak file. Alternatively you can specify "**-d given**" to require a specific overlap between the start and end coordinates of the peaks. This is more useful if comparing large regions as opposed to peaks, which tend to be highly localized. The program will also output the numbers for creating a venn diagram, and these can be directed to a specific file by specifying "**-venn <filename>**".

Merging peaks together into a single file is very useful for certain types of analysis, such as making scatter plots that compare the tag-densities between peaks from separate experiments - in this case you want to count tags at specific and common regions. Alternatively, you may be interested in separating the peaks into common and specific sets for focused analysis. To do this use the "**-prefix <filename>**" option - this will create separate files based on overlapping peaks for each set of peaks. For example:

```
mergePeaks -d 100 pu1.peaks cebp.peaks -prefix mmm
```

This will create files named "**mmm_pu1.peaks**", "**mmm_cebp.peaks**", and "**mmm_pu1.peaks_cebp.peaks**".

Peak Co-Occurance Statistics

The **mergePeaks** program will also find calculate the statistics of co-occurrence between peaks in a pairwise fashion.

Differentially Bound Peaks

For now, simply use [peak finding](#) to find peaks that are differentially enriched between two samples. Simply substitute the "Tag Directory" of the Input experiment with that of another experiment.

[Back to ChIP-Seq Analysis](#)



Can't figure something out? Questions, comments, concerns, or other feedback:
cbenner@ucsd.edu



HOMER

Software for motif discovery and ChIP-Seq analysis

This is the old version of the documentation: [New Version](#)

ChIP-Seq Analysis: Creating Histograms from High-throughput Sequencing data and Motifs

HOMER can be used to make histograms that document ChIP-Seq library and motif densities relative to specific positions in the genome. This can be done near peaks, subsets of peaks, or near promoters, exon junctions or anywhere else you find interesting. To make histograms, use the `annotatePeaks.pl` program but add the parameters `"-hist <#>"` to produce a tab delimited text file that can then be visualized using EXCEL or other data visualization software.

Basic usage:

`annotatePeaks.pl <peak file> <genome> -size <#> -hist <#> -d <tag directory 1> [tag directory2] ... -m <motif 1> <motif 2> ... > <output matrix file>`

i.e. `annotatePeaks.pl ERpeaks.txt hg18 -size 6000 -hist 25 -d MCF7-H3K4me1/ MCF7-H3K4me2/ MCF7-H3K4me3/ > outfile.txt`

Running this command is very similar to [creating annotated peak files](#) - in fact, most of the data can be used to make both types of files - hence the reason for combining this functionality in the same command. By default, HOMER normalizes the output histogram such that the resulting units are **per bp per peak**, on top of the standard total mapped tag normalization of **10 million tags**.

Histograms of Tag Directories:

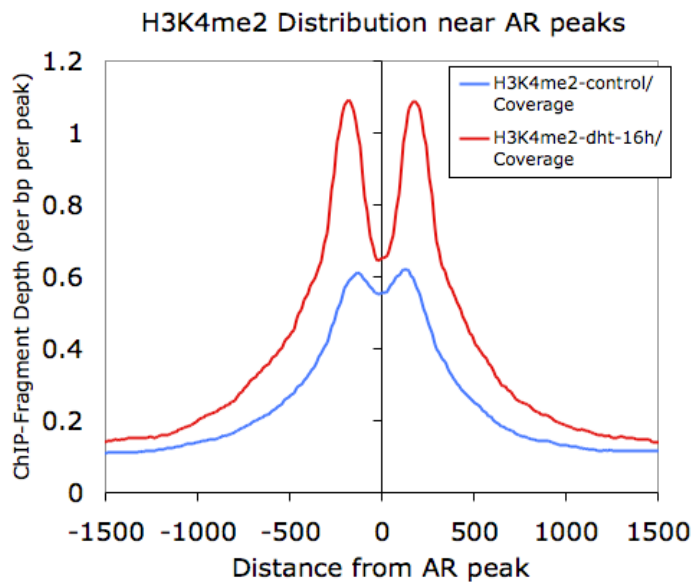
For each tag directory or motif, HOMER will output 3 columns in the histogram. In the case of tag directories, the first column will indicate **ChIP-Fragment Coverage**, which is calculated by extending tags by their estimated ChIP-fragment length, and is analogous to the profiles made for the UCSC Genome Browser. The 2nd and 3rd columns report the density of 5' and 3' aligned tags, and are independent of fragment length. For example, let's look at H3K4me2 distribution near Androgen Receptor (AR) peaks before and after 16 hours of treatment with testosterone (dht):

`annotatePeaks.pl ARpeaks.txt hg18 -size 4000 -hist 10 -d H3K4me2-control/ H3K4me2-dht-16h/ > outfile.txt`

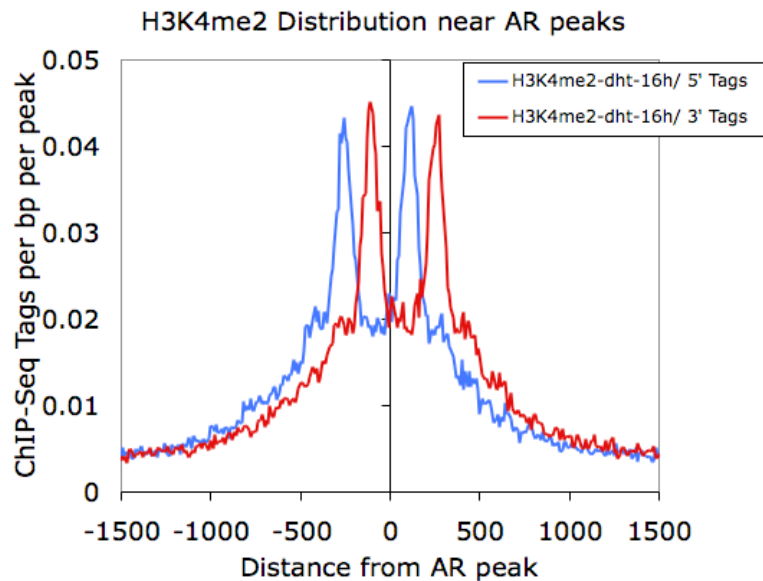
Opening outfile.txt with EXCEL, we see:

	A	B	C	D	E	F	G
1	Distance from Center	H3K4me2-control/ Coverage	H3K4me2-control/ 5' Tags	H3K4me2-control/ 3' Tags	H3K4me2-dht-16h/ Coverage	H3K4me2-dht-16h/ 5' Tags	H3K4me2-dht-16h/ 3' Tags
2	-2000	0.046971	0.001564	0.001734	0.059736	0.001824	0.002136
3	-1990	0.052394	0.003621	0.003179	0.06552	0.004344	0.002928
4	-1980	0.05661	0.003689	0.003298	0.070872	0.004008	0.003528
5	-1970	0.059721	0.003026	0.003128	0.075048	0.003816	0.003912
6	-1960	0.063614	0.003638	0.002975	0.079128	0.00396	0.00372
7	-1950	0.066742	0.002958	0.003298	0.084048	0.004392	0.00372
8	-1940	0.069751	0.003315	0.003145	0.087792	0.004128	0.003432
9	-1930	0.072403	0.003281	0.00323	0.091416	0.003312	0.004632
10	-1920	0.075259	0.002924	0.002958	0.0936	0.003504	0.004224

Graphing columns B and E while using column A for the x-coordinates, we get the following:



However, if we graph only the 5' and 3' tags that come from the H3K4me2-dht-16h directory (columns F and G):



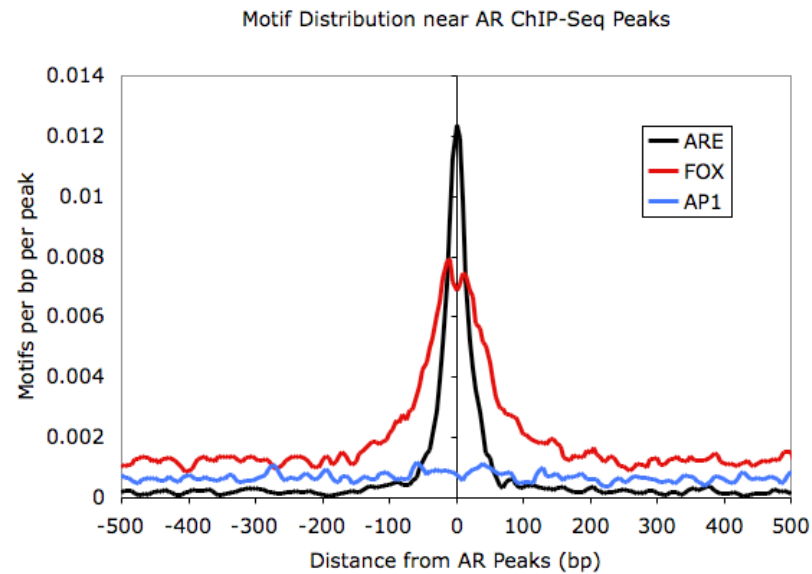
Here we can see how the 5' and 3' reads from the H3K4me2 marked nucleosomes are distributed near the AR sites.

Histograms of Motif Densities:

Making histograms out of motif occurrences is very similar to ChIP-Seq tag distributions. Run the `annotatePeaks.pl` program with "`-hist <#>`" and "`-m <motif file>`" (you can also find motif densities and tag densities at the same time):

```
annotatePeaks.pl ARpeaks.txt hg18 -size 1000 -hist 5 -m are.motif fox.motif ap1.motif > outputfile.txt
```

Graphing outputfile.txt with EXCEL:



[Back to ChIP-Seq Analysis](#)



Can't figure something out? Questions, comments, concerns, or other feedback:
cbenner@ucsd.edu



HOMER

Software for motif discovery and ChIP-Seq analysis

This is the old version of the documentation: [New Version](#)

ChIP-Seq Analysis: Creating Heatmaps from High-throughput Sequencing data

HOMER is not capable of generating actual Heatmaps *per se*, but it will generate the data matrix (similar to a gene expression matrix) that can then be visualized using standard gene expression heatmap tools. For example, I will generate a heatmap data matrix file using HOMER, and then open it with [Cluster 3.0 \(Micheal Eisen/de Hoon\)](#) to cluster it and/or visualize it with [Java Tree View \(by Alok J. Saldanha\)](#). In reality, you can use any clustering and/or heatmap visualization software (i.e. R).

Basic usage:

```
annotatePeaks.pl <peak file> <genome> -size <#> -hist <#> -ghist -d <tag
directory 1> [tag directory2] ... > <output matrix file>
```

```
i.e. annotatePeaks.pl ARpeaks.txt hg18 -size 6000 -hist 25 -ghist -d
H3K4me2-control/ H3K4me2-dht-16h/ > outputfile.txt
```

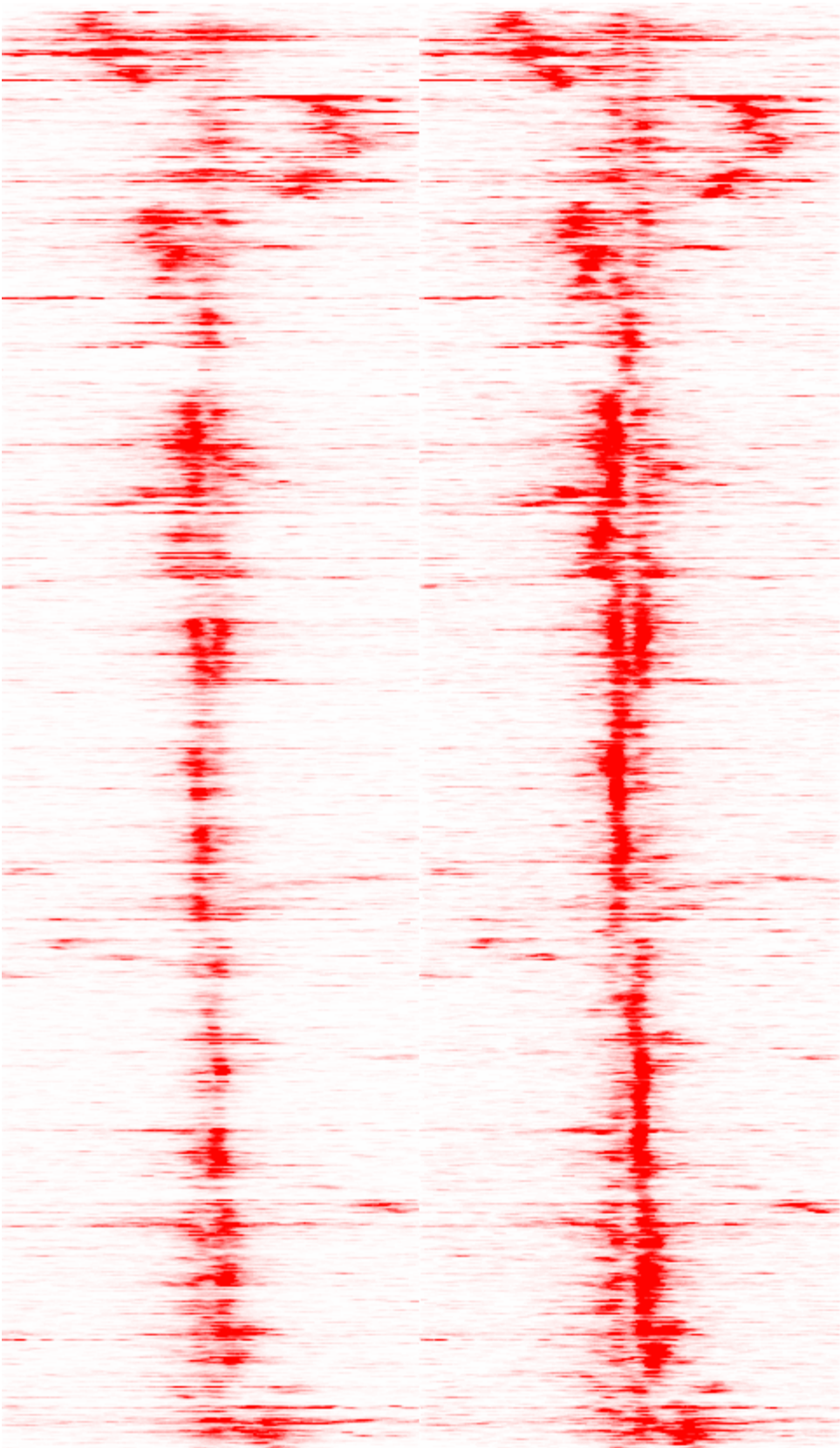
Running this command is very similar to making histograms with `annotatePeaks.pl`. In fact, a heatmap isn't really all that different from a histogram - basically, instead of averaging all of the data from each peak, we keep data from each peak separate and visualize it all together in a heatmap. The key difference when making a heat map or a histogram is that you must add "**-ghist**" when making a heatmap.

Format of Data Matrix Output File

The resulting file is a tab-delimited text file where the first row is a header file and the remaining rows represent each peak from the input peak file. The first set of columns will describe the ChIP-fragment densities from the **first tag directory** as a function of distance from the center of the peak, with bin sizes corresponding to the parameter used with "**-hist <#>**". After the first block of columns, a second block will start over with ChIP-Fragment densities from the **2nd tag directory**, and so on. If you would like to cluster this file to help organize the ChIP-Fragment distribution patterns, make sure you only cluster the "genes" (i.e. rows).

Creating the Heatmap:

If we run the command above, we have a file named outputfile.txt that contains the "heatmap matrix". Opening that file with [Cluster 3.0](#), and clustering the genes with hierarchical clustering using average linkage, we end up with files outputfile.gtr and outputfile.cdt. Due to the scaling of hierarchical clustering (N^2), it's best to keep the number of peaks relatively small (<5k) to avoid clustering all night. After the clustering is complete, use Java Tree View to open the *.cdt file. You'll like need to scale the pixels and adjust the colors to clearly see the whole picture. For example, this is what I get (H3K4me2-control on the left, H3K4me2-dht-16h on the right):



[Back to ChIP-Seq Analysis](#)



Can't figure something out? Questions, comments, concerns, or other feedback:
cbenner@ucsd.edu



HOMER

Software for motif discovery and ChIP-Seq analysis

This is the old version of the documentation: [New Version](#)

ChIP-Seq Analysis: Centering Peaks on Motifs

One cool analysis strategy is to center peaks on a specific motif. For example, by centering peak for the ETS Transcription Factor PU.1 on the PU.1 motif (GAGGAAGT), you can map the spacial relationship between the bound PU.1 motif and other sequence features and ChIP-Seq tags.

To center peaks on a motif, run **annotatePeaks.pl** with the following options:

```
annotatePeaks.pl <peak file> <genome> -size <#> -center <motif file> >  
newpeakfile.txt
```

```
i.e. annotatePeaks.pl PU1peaks.txt mm8r -size 200 -center pu1.motif >  
pu1centeredPeaks.txt
```

[Back to ChIP-Seq Analysis](#)



Can't figure something out? Questions, comments, concerns, or other feedback:
cbenner@ucsd.edu



HOMER

Software for motif discovery and next-gen sequencing analysis

RNA Motif Analysis

HOMER was not originally designed with RNA in mind, but it can be used to successfully analyze data for RNA motifs. By RNA motifs, we mean short sequence elements in RNA sequences akin to DNA motifs, not structural elements such as hairpins and stuff like that. For example, HOMER can be used to successfully determine miRNA seeds in sets of co-regulated mRNAs, or RNA binding elements in CLIP-Seq data.

The `"-rna"` option can be used with `findMotifs.pl` and `findMotifsGenome.pl`, resulting in + strand only motif searching and motif display/matching with "U" instead of "T". HOMER does not contain a list of well known "RNA motifs" yet, so no "known motif" analysis is performed. If using FASTA files, please use "T" (normal DNA encoding) in the input files for now.

Analyzing Co-regulated Gene Lists for RNA motifs

HOMER contains preconfigured PROMOTER sets comprised of RefSeq mRNA sequences, or only the 5' and 3' UTRs. These are useful for analyzing gene lists for motifs in their mRNAs. To run the analysis, us `findMotifs.pl` with a mRNA PROMOTER set, and options for RNA motifs will be automatically set.

```
findMotifs.pl mir1-downregulated.genes.txt human-mRNA MotifOutput/ -rna -len 8
```

You don't actually need to specify `-rna` for this case since with the use of `"human-mRNA"` it's understood. Anyway, the output will look something like this:


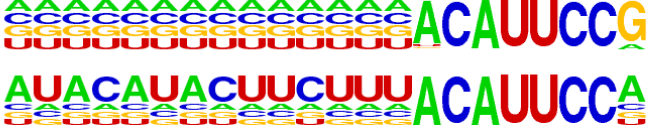

Homer *de novo* Motif Results

[Known Motif Enrichment Results](#)
[Gene Ontology Enrichment Results](#)
If Homer is having trouble matching a motif to a known motif, try copy/pasting the matrix file into [STAMP](#)
More information on motif finding results: [HOMER](#) | [Description of Results](#) | [Tips](#)
Total target sequences = 1351
Total background sequences = 19001
** - possible false positive*

Rank	Motif	P-value	log P-value	% of Targets	% of Background	STD(Bg STD)	Best Match/Details
1	ACAUUCCG	1e-116	-2.684e+02	55.29%	24.70%	1843.3bp (1700.6bp)	hsa-miR-206 MIMAT0000462 Homo sapiens miR-206 Targets (miRBase) More Information Similar Motifs Found
2	AGUAUGAC	1e-41	-9.637e+01	68.39%	49.40%	1877.0bp (1667.9bp)	hsa-miR-485-3p MIMAT0002176 Homo sapiens miR-485-3p Targets (miRBase) More Information Similar Motifs Found
3	CAUCGACG	1e-28	-6.530e+01	60.92%	45.25%	1651.5bp (1428.1bp)	hsa-miR-181a* MIMAT0000270 Homo sapiens miR-181a* Targets (miRBase) More Information Similar Motifs Found

For now, HOMER will try to match the results to the human list of miRNA seeds (from miRBase):

Matches to Known Motifs

hsa-miR-206 MIMAT0000462 Homo sapiens miR-206 Targets (miRBase)	
Match Rank: 1	
Score: 0.85	
Offset: -14	
Orientation: forward strand	
Alignment: -----ACATTCCG	
CCACACACTTCTTACATCCA	
Score: 0.85	
Offset: -14	
Orientation: forward strand	
Alignment: -----ACATTCCG	
ATACATACTTCTTACATCCA	
Score: 0.83	
Offset: -12	
Orientation: forward strand	
Alignment: -----ACATTCCG	
GGCAAAGAAGGACATTCCT	

In this case, the motif matches the miR-1 consensus seed (which is shared by miR-206 and miR-613).

- There are two RNA specific options for findMotifs.pl in rna mode:
- min <#> : minimum length of mRNA to consider (basically removes extremely short mRNA sequences from the analysis)
 - max <#> : maximum length of mRNA to consider (removes really long RNAs from the analysis)

Analyzing CLIP-Seq for RNA motifs

HOMER can analyze strand-specific genomic regions for motifs, such as the regions that would be defined from CLIP-Seq. To do this, just run findMotifsGenome.pl using the "-rna" flag (make sure your regions are strand specific!!). For now, HOMER just uses the same random genomic background used for ChIP-Seq motif finding. You could imagine that a better RNA motif finding background would be RNA, i.e. strand specific exon/intron sequences. You'd be right, but managing this with respect to the different experiments (i.e. intronic binding vs. mRNA binding vs. non-coding RNA binding) is tricky and for now left up to the user (you can specify your own strand specific background with "-bg <peak/BED file>"). Trying this with FOX CLIP-Seq data:

```
findMotifsGenome.pl fox2.clip.bed hg17 MotifOutput -rna
```

This will give the following results (which resembles a UGCAUG FOX motif):

Homer *de novo* Motif Results

[Known Motif Enrichment Results](#)

[Gene Ontology Enrichment Results](#)




If Homer is having trouble matching a motif to a known motif, try copy/pasting the matrix file into [STAMP](#)

More information on motif finding results: [HOMER](#) | [Description of Results](#) | [Tips](#)

Total target sequences = 8476

Total background sequences = 37836

* - possible false positive

Rank	Motif	P-value	log P-value	% of Targets	% of Background	STD(Bg STD)	Best Match/Details
1		1e-273	-6.306e+02	30.57%	15.27%	53.9bp (73.6bp)	dme-miR-954-3p MIMAT0020846 Drosophila (miRBase) More Information Similar Motifs Found
2		1e-34	-7.881e+01	5.65%	3.08%	56.0bp (76.5bp)	mmu-miR-3060 MIMAT0014827 Mus musculus (miRBase) More Information Similar Motifs Found
3		1e-21	-4.901e+01	0.28%	0.02%	54.1bp (43.8bp)	hsa-miR-4457 MIMAT0018979 Homo sapiens (miRBase) More Information Similar Motifs Found



Can't figure something out? Questions, comments, concerns, or other feedback:
cbenner@ucsd.edu



HOMER

Software for motif discovery and ChIP-Seq analysis

Practical Tips to Motif Finding with HOMER

Below are some general tips for getting the most out of you motif analysis when using HOMER. Be sure to look over [this section about judging motif quality!](#)

What to do if motif finding takes too long...

Ctrl+C... If you are using reasonable parameters (see next section), it shouldn't take more than an hour or so, and in most cases much less.

Choosing the length of motifs to find

It's almost always a good idea to start with the default parameters. Resist the urge to find motifs larger than 12 bp the first time around. Longer motifs will show up as different short motifs when finding shorter motifs. If there aren't any truly significant motifs when looking at short motifs, it is unlikely that you will find good long motifs either. And it doesn't take much time to check for short motifs.

i.e. **-S 50 -len 8,10**

Once you do find motifs that look promising, try looking for longer motifs ("**-len 12,14**").

If searching for motifs >12 or 14 bp (i.e. 20 bp motifs), try to do the following:

- try to reduce the number of target sequences to include only high quality sequences (such as ["focused" peaks](#) or peak with the highest peak scores).
- try limiting the length of sequences used (i.e. "**-size 50**" when using **findMotifsGenome.pl**)
- try limiting the total number of background sequences (i.e. "**-N 20000**" when using **findMotifsGenome.pl**)

In a practical sense, you should be able to search for motifs of length 20 or 30 when analyzing ~10k peaks with parameters "**-len 20,30 -size 50 -N 25000**". HOMER wasn't really designed to find really long motifs; since it is an empirical motif finder, the sequence "space" gets a bit sparse at lengths >16, but in practice it still works.

How many sequences can HOMER handle?

In theory, a lot (i.e. millions). It has been designed to work well with ~10k target sequences and 50k background sequences. If you are using a large number of sequences with **findMotifs.pl**, you may want to use the "**-b**" option, which switches to the [cumulative binomial distribution](#) for motif scoring, which is faster to calculate and gives essentially the same results when using large numbers of sequences. The binomial is used by default in **findMotifsGenome.pl**. (I guess it should be called BOMER !?).

Choosing background sequences

Most of the methods in HOMER attempt to select the proper background for you, but in some cases this doesn't work. Normally, HOMER attempts to normalize the CpG content in target and background sequences. If you believe normalizing the overall G+C content (GC) is better, use the option "**-gc**" when performing motif finding with either **findMotifs.pl** or **findMotifsGenome.pl**.

In some cases the user may have a better idea of what the background should be, so HOMER offers the following options:

Promoters: When using analyzing promoters with **findMotifs.pl**, if you wish to use a specific set of promoters as background, place them in a text file (1st column is the ID) and use the "**-bg <background IDs file>**" option. Genes found in the target and background will be removed from the background set so

that they don't cancel out each other. Examples:

- Use expressed genes from a microarray as background
- Use only genes represented on the microarray as background

Genomic Regions: When analyzing peaks/regions with **findMotifsGenome.pl**, you can specify the genomic regions of appropriate background regions by placing them in their own peak file and using the **"-bg <background peak file>"**. Examples:


- Specify peaks common to two cell types as background when trying to find motifs specific to a set of cell-type specific peaks - this will help cancel out the primary motif and reveal the co-enriched motifs
- If peaks are near Exons, specify regions on Exons as background to remove triplet bias.

FASTA Files: Here you have (the necessary) freedom to specify whatever you want!

You man also want to disable CpG/GC normalization depending on how you selected your background, which can be done with **"-noweight"**.

How to Judge the Quality of the Motifs Found

WARNING: Because this is the hardest thing for people to understand, I'll say it again here. HOMER will print the best guess for the motif next to the motif results, but before you tell your adviser that your factor is enriched for that motif, it is highly recommended that you look at the alignment!!! Here is an example of what might be going on:

8		3.941e-24	-5.389e+01	YY1_01 More Information Similar Motifs Found
---	---	-----------	------------	---

In this case, HOMER has identified YY1 as the "best guess" match for this *de novo* motif. Well, lets click on "More Information" and see what's up:

YY1(M00069)


Match Rank: 2

Score: 6.11

Offset: 0

Orientation: reverse strand

Alignment: AGCAGGCA-----
ANCAGNCAAGATGGCCGNGN



As you can see in this case, the motif aligns to the edge of the known YY1 motif, and not to the core of the YY1 motif (CAAGATGGC). This doesn't mean that the YY1 motif is not enriched in your data, but unless there are other motif results that show enrichment of the other parts of the YY1 motif, it is not likely that the YY1 motif is enriched in your data set.

And as always, remember that HOMER is a *de novo* motif tool!!!! Even though HOMER will guess the best match, if it is a novel motif, your don't want to trust that match anyway. Hence, the you can see the importance of viewing the alignment and getting a feel for what evidence exists either for or against this assignment.

There are many cases where HOMER will find motifs with very low p-values, but the motifs might look "suspicious". Poor quality motifs can be loosely classified into the following groups:

Low Complexity Motifs:

These types of motifs tend to show preference for same collection of 1, 2, 3, or 4 nucleotides in each position and are typically very degenerate. For example:



These motifs typically arise when a systematic bias exists between target and background sequence sets. Commonly they will be very high in GC-content, in which case you may want to try adding "-gc" to your motif finding command to normalize by total GC-content instead of CpG-content.

Other times this will come up when analyzing sequences for various genomic features that have not been controlled for in the background - for example, comparing sequences from promoters to random genomic background sequences in some organisms will show preferences for purines or pyrimidines. HOMER is very sensitive, so if there is a bias in the composition of the sequences, HOMER will likely pick it up.

Simple Repeat Motifs:

Some times motifs will show repeats of certain patterns:



Usually motifs like this will be accompanied by several other motifs looking highly similar. Unless there is a good reason to believe these may be real, it's best to assume there is likely a problem with the background. These can arise if your target sequences are highly enriched on exons (think triplets) and other types of sequences, and if "-gc" doesn't help, you may have to think hard about the types of sequences that you are trying to analyze and try to match them. (i.e. Promoters vs. Promoters, Exons vs. Exons etc.)

Small Quantity Motifs / Repeats:

These are a little harder to explain. These look like real motifs but are found in an incredibly low percentage of targets - i.e. like an oligo or part of a repeat that is in a couple of the target sequences that appears as a significant motif. Statistically speaking they are enriched, but likely not real. These are the biggest problem when looking for motifs in promoters from a small list of regulated genes. In principle, in a motif is present in **less than 5% of the targets sequences**, there may be a problem.

Leftover Junk:


These are motifs that appear in your lower in your results list after you've discovered high quality motifs. If an element is highly enriched in your sequences, HOMER will (hopefully) find it, mask it, and then continue to look for motifs. In this case, many of the other motifs that HOMER finds will be offsets or degenerate versions of highly enriched motif(s) found at the beginning. For example (another PU.1 example):

The top motif identified:

Rank	Motif	P-value	log P-pvalue	Best Match/Details
1		0.000e+00	-2.938e+04	PU.1/ThioMac-PU.1-ChIP-Seq/Homer More Information

Examples further down the list:

8		0.000e+00	-3.711e+03	PU.1/ThioMac-PU.1-ChIP-Seq/Homer More Information
22		0.000e+00	-1.692e+03	STAT1/Stat More Information

74		6.727e-190	-4.356e+02	MAF(M00648) More Information
----	---	------------	------------	---

This are not necessarily negative results, but they should be place in context. This commonly happens in ChIP-Seq data sets where the immunoprecipitated protein is highly expressed and binds strongly a ton of binding sites. These "other" motifs are likely also capable of binding PU.1 and probably represent low affinity binding sites, but giving them too much individual attention is not recommended in this context given they are motifs that have been constructed using leftover oligos in the motif finding process that didn't make it into the most highly enrichment motifs. A safer way to approach these elements is to repeat the motif finding procedure with regions lacking the top motif, or by adding "**-mask <motif file>**" to the motif finding command to cleanly mask the top motif from the motif finding procedure.



Can't figure something out? Questions, comments, concerns, or other feedback:
cbenner@ucsd.edu