HOMER

Software for motif discovery and next-gen sequencing analysis

---

Analyzing Hi-C genome-wide interaction data

HOMER contains several programs and analysis routines to facilitate the analysis of Hi-C data. Hi-C couples chromosome conformation capture (3C) with deep sequencing to reveal regions of genomic DNA that are in close spatial proximity in the nucleus. Hi-C has emerged as a powerful technique to understand how the genome is packaged in cells to control gene expression. Unlike ChIP-PET, 5C, or 4C, Hi-C is unbiased. While HOMER can be jury-rigged to gleam information from other 3C-sequencing based methods, it has been specifically tailored Hi-C analysis.

Specialized Hi-C programs in HOMER

HOMER has several specialized programs for Hi-C analysis. Each is covered in the tutorials below:

makeTagDirectory - special paired-end operations for making HOMER-style tag directories and filtering options for Hi-C

analyzeHiC - primary analysis program - generates interaction matrices, normalization, identification of significant interactions, clustering of domains, generates Circos plots (most of the following programs use this one internally)

runHiCpca.pl - automated PCA analysis on Hi-C data to identify "compartments"
getHiCcorrDiff.pl - calculates the difference in correlation profiles between two Hi-C experiments
findHiCCompartments.pl - find continuous or differential regions from PCA/corrDiff results that describe what compartment regions of DNA belong to (name changed - was called *getDomains.pl*)

findHiCInteractionsByChr.pl - helps automate the finding of high-resolution intra-chromosomal interactions
annotateInteractions.pl - program for re-analysis of significant interactions, such as relating them to ChIP-Seq peaks

SIMA.pl - Novel tool to boost sensitivity by pooling features together when performing interaction calculations

3rd Party Software

The following 3rd Party [Free] Software is used by HOMER for visualizing Hi-C results. They are required for specific functions (i.e. if you don't care about PCA, don't worry about installing R). *Most* are straightforward to install:

Java Tree View - view interaction matrices (or any software to generate and view heatmaps, needed to view standard **analyzeHiC** output)
Circos - software to generating circle interaction diagrams. Tips on installing Circos (Needed when running **analyzeHiC** with "**-circos**" option)
R - statistical computing environment, used for PCA analysis (no special packages needed, used by **runHiCpca.pl** program)
Cytoscape - view processed network files created by **annotateInteractions.pl** and **SIMA.pl**

Make sure R and circos are available in your executable path as Homer will attempt to call these programs directly. Java Tree View is strictly for visualizing output files, as is Cytoscape.

Analyzing Hi-C data with Homer

Below is a description of the general workflow of Hi-C analysis with HOMER, and each section contains detailed information about various analysis steps.

1. Creating Tag Directories, quality control, and read filtering for Hi-C data (**makeTagDirectory**)
2. Creating Background Models for Hi-C Data (**analyzeHiC**)
3. Making Interaction matrices and normalizing interaction counts (**analyzeHiC**)
4. Sub-nuclear compartment analysis/PCA/Clustering (**runHiCpca.pl**, **getHiCcorrDiff.pl**, **findHiCCompartments.pl**)
5. Identifying significant interactions (**analyzeHiC**, **findHiCInteractionsByChr.pl**)
6. Analysis and annotation of interactions with respect to other data types (**annotateInteractions.pl**)
7. Visualizing Interactions and Other Sequencing data with Circos (**analyzeHiC**)
8. Structured Interaction Matrix Analysis (**SIMA.pl**)

Important Tips for analyzing Hi-C data with 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

---

Installing Circos

Installing Circos for use with HOMER can be divided into 3 parts. If you already use Circos on your system, you may only need to pay attention to the last part. I recommend checking out the official Circos installation directions. Below is a quick guide with tips:

1. Download the Circos software and unzip it into a location where you won't delete it.

You want to download the general circos package (i.e. circos-0.62-1.tgz). Save it to location, and unzip it (tar zxvf circos-0.62-1.tgz). (At this point you can also do step 3, which is add the circos bin/ directory to you executable PATH.

2. Upgrade your Perl modules to include the necessary prerequisites for Circos.

Follow the instructions on this page: Perl and Modules. It talks a little while as some of the modules will want to install dependencies. For the most part, this means running the command:

sudo perl -MCPAN -e shell

Followed by commands to install each missing module:
cpan[1]> **install Config::General**

The following modules need to be installed:

Config::General
GD
GD::Polyline
List::MoreUtils
Math::Bezier
Math::Round
Math::VecStat
Params::Validate
Readonly
Regexp::Common
Set::IntSpan

Text::Format
Font::TTF::Font
Clone

By far the trickiest part is installing GD - often, you need to install the system C libraries for GD. This often means running something like:

Linux (Ubuntu): "sudo apt-get install lib-gd2"
Linux (Redhat/CentOS): " yum install gd-devel"
Mac OSX: "sudo fink install gd2".

Try to follow the Circos instructions. Hopefully, buy the time you read this, it will be magically easier.

3. Add the Circos bin directory to you executable path so that you can run it by simply typing "circos" in the command line. (i.e. edit your ~/.bashrc or ~/.bash_profile file to include "PATH=$PATH:/Users/chucknorris/circos-0.54/bin/")

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

HOMER

Software for motif discovery and next-gen sequencing analysis

---

Initial Hi-C data processing with HOMER

Hi-C and variant technologies use paired-end sequencing to assign regions of the genome to the same physical location. After sequencing, you should have two files (or many sets of 2 files) for the first and second reads from each pair. Unlike paired end sequencing for other techniques like genomic resequencing or RNA-Seq, where you might expect the 2nd read to be located within the general vicinity of the first read, read-pairs from Hi-C should be processed independently (e.g. do not use Tophat, bowtie, bwa or any short read alignment software in "paired-end" mode - each read should be mapped independently!).

Once Hi-C reads have been mapped, the **makeTagDirectory** program is used to process the reads. Hi-C has many more sources of noise than expected, so this step is particularly important, and may need to be revisited and tweaked. For this reason, we recommend running the command twice - once to complete the lengthy process of merging all the paired end reads into an "unprocessed" tag directory, then a second time on copies of the first tag directory using different parameters. For example:

makeTagDirectory HiC-Unprocessed reads1-1.sam,reads1-2.sam reads2-1.sam,reads2-2.sam -tbp 1

Then experiment with different processing parameters (remember to include "**-update**" - this tells the program not to look for new data files but reprocess the existing tag directory)

cp -r HiC-Unprocessed/ HiC-HindIIIonly/
makeTagDirectory HiC-HindIIIonly -update -restrictionSite AAGCTT -both -genome hg19 -removePEbg

cp -r HiC-Unprocessed/ HiC-noSelfLigation/
makeTagDirectory HiC-noSelfLigation -update -restrictionSite AAGCTT -removeSelfLigation -removePEbg -genome hg19

This is a more efficient way to approach Hi-C processing if you're not sure what the data looks like yet.

Creating a Paired-End Tag Directory with HOMER

Use **makeTagDirectory** to process paired-end sequencing into a tag directory for use with HOMER programs. For Hi-C, you want to map the reads

independently. This is different than most "paired-end" alignments (such as for RNA-Seq) that constrain the relative position of the fragments. For Hi-C these assumptions do not hold. I'd recommend mapping the same way you would ChIP-Seq data, allowing only unique genomic matches. You may also want to check for restriction sites in your reads - sequence on either side of the restriction site might map to different locations, so it might be wise to trim the sequence after the restriction site to help with mapping (i.e. use "**homerTools** **trim -3 AAGCTT reads.fq**" to trim the reads before mapping)

To create a paired-end tag directory, you must run **makeTagDirectory** by entering the two halves of the mapped reads separated by a comma, without spaces. For example:

> **makeTagDirectory <OutputTagDirectory> <alignment file-read1>,<alignment file-read2> ... [-illuminaPE] [-tbp 1]**
> i.e. **makeTagDirectory ES-HiC lane1-read1.sam,lane1-read2.sam lane2-read1.sam,lane2-read2.sam -illuminaPE -tbp 1**

HOMER matches reads between alignment files by matching their names, but to work with Illumina (which often contains a 0 or 1 at the end to delineate the read source), it will need to remove the last character before trying to match. In this case, add "**-illuminaPE**" to the end of the command. If you download reads from the SRA they often have the same name for each paired end read - in these cases you don't want to remove the last character, so do NOT add "**-illuminaPE**" to the end of the command in that case.

Another typical option to add for Hi-C data is "**-tbp 1**", which will only consider read pairs with the exact same ends once. Given the search space in Hi-C, you would never expect to see reads-pairs start from the exact same places twice in the data (at least in the case of sonication). If you do, they are likely clonal and a result of over-sequencing your library or some other artifact.

On a side note, the HOMER tag directory created for paired-end reads is very similar to the directories created when running **makeTagDirectory** for single-end reads. The paired end reads are stored in *.tags.tsv files (extended format to include the pair), only each read will get 2 entries in the appropriate *.tags.tsv so that the reads can be indexed on each chromosome in one file. A flag is also set in the tagInfo.txt file to inform HOMER to treat the experiment as a paired-end file. Many of the programs in HOMER, such as **makeUCSCfile**, will treat paired-end tag directories as "single-end" directories to create output. This can be useful if you want to visualize read coverage in the UCSC Genome Browser or want to find peaks in the Hi-C read coverage with **findPeaks**.

One important note: the tagInfo.txt file will report the "Total Tags" value as 2x the total number interactions. This is because HOMER is storing each PE tag twice - the software later adjusts this number to reflect this redundancy, but keep this in mind when interpreting this number.

Alternative Input Files

HOMER will also accept "Hi-C Summary" files (use "**-format HiCsummary**"), which is a simple text file format that contains the mapping positions for a read-pair on each line. These files are *tab*-delimited text files with the following column assignments:

Hi-C Summary Format (columns):
1. Read Name (can be blank)
2. chromosome for read 1
3. positions for read 1 (5' end of read, one-indexed)
4. strand of read 1 (+ or -)
5. chromosome for read 2
6. positions for read 2 (5' end of read, one-indexed)
7. strand of read 2 (+ or -)

An example of using a Hi-C summary file:

makeTagDirectory proB-HiC -format HiCsummary proB.HiC.FA.summary.txt

You can also combine Hi-C tag directories or add *tags.tsv files, just like with the regular **makeTagDirectory** command. For example, to combine directories:

makeTagDirectory ES-HiC-All -d ES-HiC-rep1/ ES-HiC-rep2/ ES-HiC-rep3/

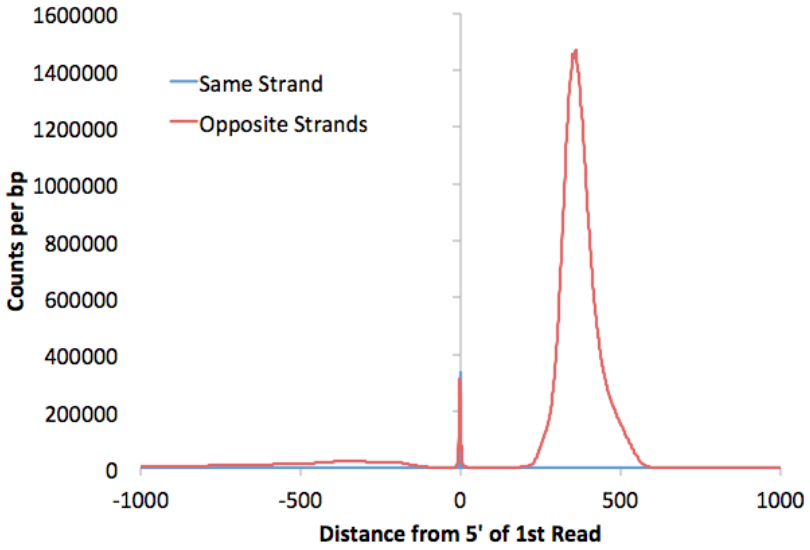
Hi-C Quality Control

When running the **makeTagDirectory** program, several standard quality control analysis files will be created that resemble the analysis carried out for single-end reads. You can also check for GC bias by specifying the genome (i.e. "**-genome hg19**") and "**-checkGC**" at the command line (analysis will be conducted by treating the samples as "single" reads).

The following two Hi-C/paired-end read specific QC files will also be produced by default:

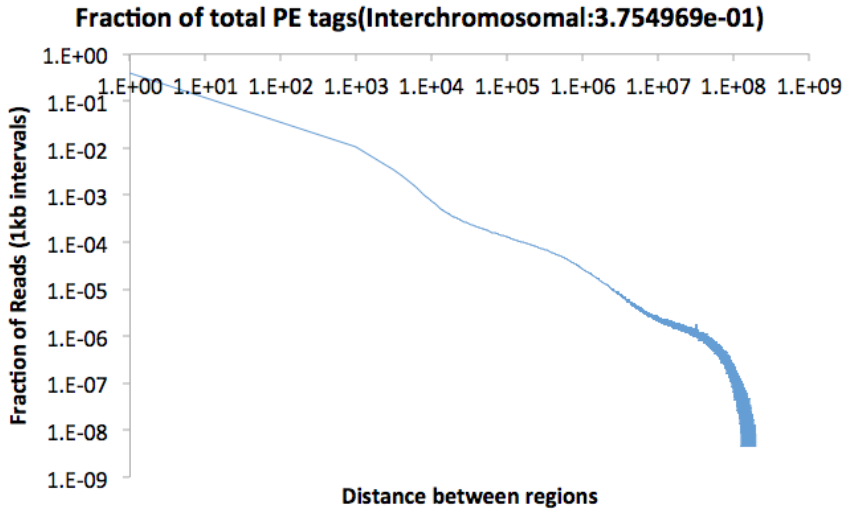
petagLocalDistribution.txt

This file is similar to the autocorrelation file, except that it shows the relationship between the 5' ends of the paired reads. This data is used to determine the fragment size of the Hi-C fragments used for sequencing, assuming that we should see a natural peak from "re-ligation events" or random genomic fragments that got into the sequencing library. Below is an example (Here we'd estimate the fragment length to be about 350 bp):



petagDistDistribution.txt

This file contains a histogram describing the fraction of paired-end reads that are found at different distances from one another. The file stops at 300 million bp, and is divided into 1kb bins. The top of the file also contains the fraction of interchromosomal interactions. Below is an example (plotted as a log vs. log plot). The fraction of PE reads with interactions greater than 300 Mb in distance are shown in the final point.

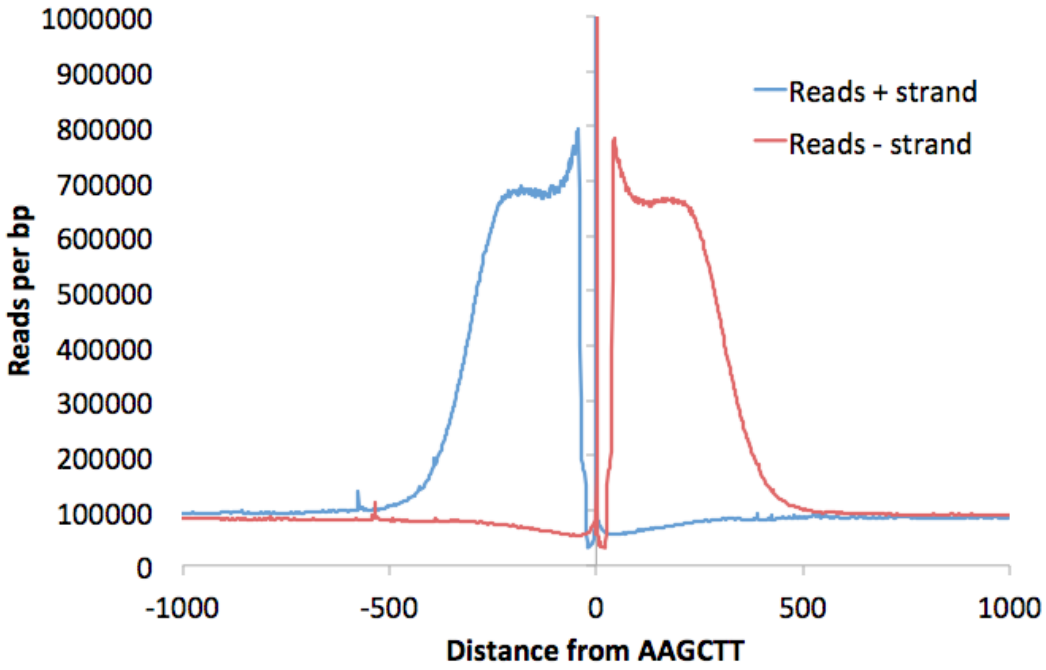


Restriction Sites

One major problem that can arise during Hi-C is that "background" ligation events may occur. These are ligation events that occur after sonication [I think] as they seem to occur between random fragments of DNA with no regard to restriction site used for the Hi-C assay. In theory, all reads should be in the vicinity of the restriction site used for the assay, and if they aren't, there is a good chance they are noise (more later on this). To see the distribution of reads around your restriction site, use the option "**-restrictionSite <seq>**" (i.e. "-**restrictionSite AAGCTT**"). You MUST specify the genome so that HOMER can figure out

where the restriction sites are located (i.e. "**-genome mm9**"). If the restriction site has a lot of star activity, you can also add "**-rsmis <#>**" (i.e. "**-rsmis 1**") to indicate how many mismatches are tolerated in the sites. By default it only looks for perfect restriction sites.

Adding these options will produce a file named "petagRestrictionDistribution.<seq>.mis<#>.txt" (such as petagRestrictionDistribution.AAGCTT.mis0.txt). This file will display the distribution of reads relative to the restriction site. Graphing it with Excel will look something like this:



Filtering Uninformative Reads

The Hi-C protocol can produce sequencing libraries that contain many read pairs that are not representative of true interactions. This includes continuous genomic fragments, self-ligation or re-ligation products, and background non-specific ligation that occurs independent of a restriction site.

-removePEbg : This will remove paired-end reads that are likely continuous genomic fragments or re-ligation events. If this option is specified, read pairs are removed if they are separated less than **1.5x** the sequencing insert fragment length. This distance is estimated from the distribution found in the file "**petagLocalDistribution.txt**" described above. If you wish to specify your own size, use "**-fragLength <#>**" to set this length manually.

-removeSpikes <#size> <# fold> : This option will remove PE reads that originate from regions of unusually high tag density. These regions arise for various reasons, for example, a region was duplicated in a given cell line. These types of anomalies tend to create artifacts in the data. Removing these from the analysis can help clean things up. The size parameter is the window of tag

density that will be examined to find anomalies, and #-fold is how many tags over the average is needed to determine if a locus should be removed from the dataset. I'd recommend something like "**-removeSpikes 10000 5**" to get ride of the regions that are really bad. For example, this will scan all 10kb regions in the genome, calculate the average read density, and remove reads from regions that contain more than 5x the average number of reads.

Filtering Reads based on Restriction Sites

All the options in this section depend on restriction sites and require the following to be part of the command:
-restrictionSite <seq> -genome <genome>

Restriction Site Proximity: The following options can be used to specify which reads should be kept with respect to the location of nearby restriction sites. The distance used to assign reads to restriction site is **1.5x** the fragment length by default (set it with **-restrictionSiteLength <#>**):

-both : Only keep reads if both ends of the paired-end read have a restriction site within the fragment length estimate 3' to the read. This is the most conservative, and probably **recommended** for most types of analysis.
-one : Only keep reads if one or both ends of the paired-end read are near restriction sites.
-onlyOne : Only keep reads if one (and only one) of the reads in a paired-end read are near a restriction sites (mainly used for troubleshooting).
-none : Only keep read if neither of the ends are near a restriction site (mainly used for troubleshooting)

-removeSelfLigation : Remove reads if their ends form a self ligation with adjacent restriction sites.
-removeRestrictionEnds : Removes reads if one of their ends starts on a restriction site.
-restrictionSiteLength <#> : Maximum distance to a restriction site for assignment.

Recommended Settings:

The following is an example of how an average experiment should probably be analyzed. In this example we'll make an initial tag directory and remove all clonal reads, then make a copy of the directory, and then filter reads based on HindIII restriction sites:

analyzeHiC ES-HiC-unfiltered reads1-1.sam,reads1-2.sam -tbp 1

cp -r ES-HiC-unfiltered ES-HiC-filtered

analyzeHiC ES-HiC-filtered -update -genome hg19 -removePEbg -restrictionSite AAGCTT -both -removeSelfLigation -removeSpikes 10000 5

Command Line Options:

```
Usage: makeTagDirectory <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.  Program will also
unzip *.gz, *.bz2, and *.zip files and convert *.bam to sam files on the fly
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.
To perform QC/manipulations on an existing tag directory, add "-update"

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:+/-)
                  -force5th (5th column of BED file contains # of reads mapping to
position)
            sam - SAM formatted files (use samTools to covert BAMs into SAM if you
have BAM)
                  -unique (keep if there is a single best alignment based on mapq)
                        -mapq <#> (Minimum mapq for -unique, default: 10, set
negative to use AS:i:/XS:i:)
                  -keepOne (keep one of the best alignments even if others exist)
                  -keepAll (include all alignments in SAM file)
                  -mis (Maximum allowed mismatches, default: no limit, uses MD:Z:
tag)
            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:misInfo)
            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[,])
mCpGbed - encode style mCpG reporting in extended BED format, no auto-detect
(1:chr,2:start,3:end,4:name,5:,6:+/-,7:,8:,9:,10:#C,11:#mC)
-minCounts <#> (minimum number of reads to report mC/C ratios, default: 10)
HiCsummary - minimal paired-end read mapping information
(1:readname,2:chr1,3:5'pos1,4:strand1,5:chr2,6:5'pos2,7:strand2)
-force5th (5th column of BED file contains # of reads mapping to position)
-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)
-update (Use current tag directory for QC/processing, do not parse new alignment files)
-tbp <#> (Maximum tags per bp, default: no maximum)
-precision <1|2|3> (number of decimal places to use for tag totals, default: 1)

GC-bias options:
-genome <genome version> (To see available genomes, use "-genome list")
-or- (for custom genomes):
-genome <path-to-FASTA file or directory of FASTA files>

-checkGC (check Sequence bias, requires "-genome")
-freqStart <#> (offset to start calculating frequency, default: -50)
-freqEnd <#> (distance past fragment length to calculate frequency, default: +50)
-oligoStart <#> (oligo bias start)
-oligoEnd <#> (oligo bias end)
-normGC <target GC profile file> (i.e. tagGCcontent.txt file from control experiment)
Use "-normGC default" to match the genomic GC distribution
-normFixedOligo <oligoFreqFile> (normalize 5' end bias, "-normFixedOligo default" ok)
-minNormRatio <#> (Minimum deflation ratio of tag counts, default: 0.25)
-maxNormRatio <#> (Maximum inflation ratio of tag counts, default: 2.0)

Paired-end/HiC options
-illuminaPE (when matching PE reads, assumes last character of read name is 0 or 1)
-removePEbg (remove paired end tags within 1.5x fragment length on same chr)
-PEbgLength <#> (remove PE reads facing on another within this distance, default: 1.5x fragLen)
-restrictionSite <seq> (i.e. AAGCTT for HindIII, assign data < 1.5x fragment length to sites)
Must specify genome sequence directory too. (-rsmis <#> to specify mismatches, def: 0)
-both, -one, -onlyOne, -none (Keeps reads near restriction sites, default: keep all)
-removeSelfLigation (removes reads linking same restriction fragment)

-removeRestrictionEnds (removes reads starting on a restriction fragment)

-assignMidPoint (will place reads in the middle of HindIII fragments)

-restrictionSiteLength <#> (maximum distance from restriction site, default: 1.5x fragLen)

-removeSpikes <size bp> <#> (remove tags from regions with > than # times the average tags per size bp, suggest "-removeSpikes 10000 5")

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

HOMER

Software for motif discovery and next-gen sequencing analysis

---

Hi-C Background Models: Normalizing Genomic Interactions for Linear Distance and Read Depth

Chromosome conformation capture (3C) and related technologies measure the frequency that two genomic fragments appear in close physical proximity (cross-linked in the same complex). One of the goals when analyzing Hi-C data is to understand which loci in the genome tend to "interact" or "don't interact" in a biologically meaningful way. Unfortunately, the total number Hi-C reads between any two loci is dependent on many factors, many of which need to be normalized before meaningful conclusions can be reached.

Primary Sources of Technical Bias in Hi-C Interaction Counts

Read depth per region: Since Hi-C is an unbiased assay of genomic structure, we expect to observe equal read coverage across the genome. However, factors such as the ability to map reads uniquely (e.g. density of genomics repeats), the number of restriction sites, and genomic duplications/structural variation in the experimental sample will all influence the number of reads.

Linear distance between loci along the chromosome: If two loci are along the same polymer/chromosome of DNA, the loci are constrained with respect to one another independent of any specific structures adopted by the chromosome. More to the point, loci closely spaced along a chromosome are almost guaranteed to be 'near' one another if for no other reason their maximal separation is the length of DNA between them. As a result, closely spaced loci with have very high Hi-C read counts, regardless of their specific conformation. This is generally true of all 3C-based assays.

Sequencing Bias: GC% bias, ligation preferences during library construction, normal sequencing problems.

Chromatin compaction: This source of bias isn't really an artifact as it reflects biologically meaningful configuration of chromatin fibers. The idea is that different types of chromatin environments tend to "fold" in different ways, with heterochromatic/inactive/inert chromatin displaying a different average interaction frequencies as a function of distance than euchromatin/active/permissive regions. However, when looking for specific interaction in a particular chromatin environment, it can be useful to understand the properties of your local region of DNA to help interpret what

is meaningful or just normal for that type of chromatin environment.

Analyzing Hi-C data at different resolutions

Most Hi-C experiments to date are severely under-sequenced. For example, there are approximately a million HindIII restriction sites in the genome. If have 100 million [filtered] paired-end reads then we will only find 200 reads ends at each HindIII restriction fragment. Since (in theory) each HindIII fragment may interact with each of the other fragments, we would observe on average 0.0002 reads connecting any two HindIII fragments in the genome. In reality, most of these reads are between HindIII fragments that are within close proximity. However, for many of the observed interactions, it is difficult to know if a single observed interaction is a significant event or simply noise. Extremely high sequencing depth may help resolve this, but for the most part we are force to come up with more clever ways to analyze the data.

One of the simplest approaches is too analyze reads across large regions to boost the total number of reads considered. This approach was used in the original Hi-C paper (Lieberman-Aiden et al.), and it remains a powerful strategy. The resolution of the analysis is therefore an important consideration.

HOMER uses two (related) notions of resolution. The first, "-**res <#>**" represents how frequent the genome is divided up into regions. The second, the super resolution "**-superRes <#>**" (poorly named), represents how large the region is expanded when counting reads. Usually the "**-res <#>**" should be smaller than the "**-superRes <#>**". For example a res of 50000 and a superRes of 100000 would mean that HOMER will analyze the regions 0-50k, 50k-100k, 100k-150k etc., but at each region it will look at reads from a region the size of 100k, so it would look at reads from -25k-75k, 25k-125k, 75k-175k, etc. This means HOMER will analyze data with overlapping windows. The principle advantage to this strategy is that you don't penalize features that span boundaries.

Normalizing Hi-C Data

Since Hi-C analysis is an unbiased assay of nuclear topology, the expectation is that roughly equal numbers of Hi-C reads should originate from each region of equal size in the genome. If different numbers of Hi-C reads are observed, this is likely due to bias in mapping (e.g. repeats or duplicated genomic sequences in the region where no reads can be mapped), a variable number of restriction enzyme recognition sites (e.g. HindIII), or a technical artifact (i.e. inaccessibility of HindIII to DNA). For this reason, Hi-C reads between any two regions of a given size must be normalized for sequencing depth by dividing the total number of interaction reads that each region participates in. To calculate the expected Hi-C reads between two given regions, we use the following equation:

$$e_{ij} = (n_i)(n_j)/N$$

Where N is the total number of reads in the Hi-C experiment and n the total number of reads at each region i and j. This formulation assumes that each region has a uniform probability of interacting with any other region in the genome.

In addition to sequencing depth, it is also useful to normalize data based on the distance between interacting regions. The constrained proximity of regions along linear DNA is by far the strongest signal in Hi-C data, with regions found at close linear distances much more likely to generate Hi-C reads than regions located at large linear distances along the chromosome or located on separate chromosomes. By computing the average number of Hi-C reads as a function of distance and sequencing depth, read frequencies between specific regions can be reported relative to the average Hi-C read density for their linear distances to help reliably identify proximity relationships that are not simply a result of the general linear compaction along the DNA.

The general idea was to modify our calculation of the expected number of Hi-C reads between any two loci to account for both their linear distance and sequencing depth. However, this calculation requires that we know the true number of interaction reads originating from a given locus, and not the measured number of interaction reads. For example, if region A (mapped) is adjacent to a region B (repeat region, unmappable), the total number of Hi-C reads mapping to A will likely be much less than a scenario in which B can be mapped as well. This is particularly important of regions adjacent to regions that cannot be mapped where their linear proximity would predict a significant numbers of Hi-C reads to connect the regions. This phenomenon causes a problem when trying to estimate the expected number of reads connecting region A to another region C. Since the total number of reads mapped to A is less because of region B adjacent to A that cannot be mapped, we likely underestimate the number of Hi-C reads mapping A to C.

To account for this, we find the expected number of Hi-C reads can be found with the following equation:

$$e_{ij} = f(i-j)\ (n^*_i)(n^*_j)/N^*$$

Where f is the expected frequency of Hi-C reads as a function of distance, N* is estimated total number of reads, and n* is the estimated total number of interaction reads at each region. The goal was to identify n^*_i such that the total number of expected reads at each region i as a function of sequencing depth and distance ($S_i=sum(e_{ij})$) is equal to the observed number of reads at each region n_i. Because the expected number of reads in any given region depends on the number of reads in all other regions, the resulting nonlinear system is difficult to solve directly. Instead, a simple hill climbing optimization was used to estimate inferred total reads. To calculate the inferred number of reads at each region, the model for expected interactions above is used to compute the expected number of

read totals for each region, using the actual numbers of interaction reads as the initial values. The difference between the observed number of reads and the expected number of reads is then used to scale the values for the estimated numbers of reads. This is computed for each region iteratively and repeated until the error between expected and observed Hi-C read totals per region is near zero.

Creating Hi-C Background Models

The primary/default normalization method attempts to normalize the data for sequencing depth and distance between loci. To speed-up analysis, HOMER creates a "background model" that saves important parameters from normalization so that the background model only has to be computed once for a given resolution. If you run HOMER routines with a certain resolution, HOMER will first check if the background model has been created (they are saved in the Hi-C Tag Directory). If the appropriate background model is missing, HOMER will create it. If for some reason you want to force the creation of a new model, use the "**-force**" option.

To create a background model, or force the creation of a new one, run **analyzeHiC** and use the "**-bgonly**" flag (this will create a background model and then quit):

analyzeHiC <HiC Tag Directory> -res <#> -bgonly -cpu <#>

i.e. **analyzeHiC ES-HiC/ -res 100000 -bgonly -cpu 8**

The output will be automatically saved in the file "<tag dir>/HiCbackground_#res_bp.txt" (i.e. "ES-HiC/HiCbackground_100000_bp.txt").

The background model creation process can be broken down into the following steps (example uses resolution of 100kb):

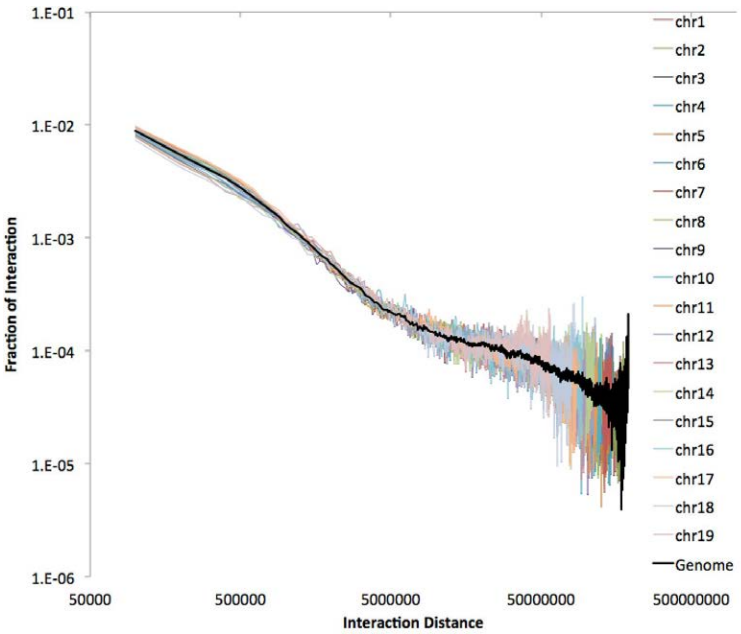
1. Divide the genome into putative regions (i.e. chr1:0-100kb, chr1:100kb-200kb, ... chrY:8900kb-9000kb)
2. Calculate the total read coverage in each region
3. Calculate the fraction of interactions spanning any given distance with respect to read depth
4. Optimize read count model to correctly assign expected interaction counts in regions with uneven sequencing depth
5. Calculate variation in interaction frequencies as a function of distance.

The process of background creation takes a while as it considers the interactions between every region at the size of the resolution. This means breaking the genome into 100000 bp regions and examining how many interactions they have between one another. For large resolutions, this isn't much of a problem, but for small resolutions the search space gets very large and the whole thing can take a long time. As a result, the process is [slightly] parallel to speed things up (i.e. use "**-cpu <#>**" to use multiple processors).

Part of the reason this takes so long is that it takes care to consider the read coverage between each set of elements and take read depths at each location into account. In the future, I would like to speed this up with some approximations that are probably 95% as good, just haven't gotten to it yet...

HiC Background Output File

There is a bunch of information in the file, including normalized interaction frequencies (both for the whole genome and for individual chromosomes). The values, including interchromosomal ones, are meant to describe frequencies between regions, so they reflect the total number of such regions in the genome. This file is interesting to graph in excel - for example, below is an example of a 100kb resolution file using a log-log plot on the interaction frequencies.



Creating Custom Background Models

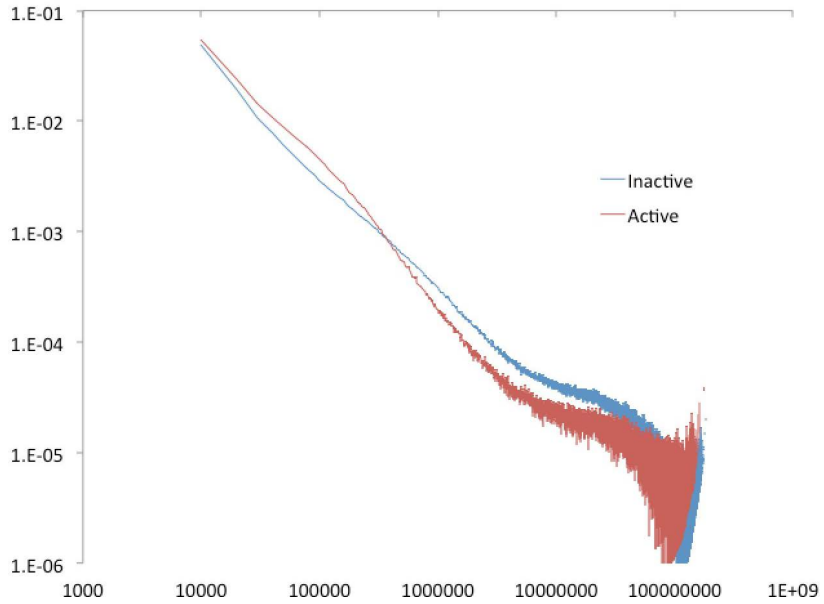
By default, HOMER creates a background using the entire genome. Sometimes you may want to understand the general properties in certain regions. A classic example fro Lieberman-Aiden et al. is to look at the connectivity in "active" and "inactive" compartments. To create a custom background model, use the option "**-model <model output file>**" to specify where to save the background model. To use custom regions instead of the whole genome, create a peak file with the regions (i.e. active regions), and supply it as an argument with "**-p <peak file>**".

For example, lets say we have two peak files, one composed of active 100kb regions (with enrichment for H3K4me2) and another file with

"inactive" regions. Running the following commands:

analyzeHiC ES-HiC/ -createModel active.model.txt -p active.peaks.txt -vsGenome -res 100000 -cpu 8 -bgonly
analyzeHiC ES-HiC/ -createModel inactive.model.txt -p inactive.peaks.txt -vsGenome -res 100000 -cpu 8 -bgonly

We can open the "active.model.txt" and "inactive.model.txt" files, and compare their genome interaction profiles:



Custom background models can be used in downstream analysis by specifying the "**-model <bgmodel.txt>**". For example, to use the "active.model.txt" from above, we would use:

analyzeHiC ES-HiC -model active.model.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

---

Creating and Normalizing Hi-C interaction Matrices

The most basic way to represent Hi-C data is in matrix format, where the number of interactions can be reported between sets of regions. Since it's difficult to extract this data from the raw mapped reads, HOMER provides tools create matrices from tag directories.

Most Hi-C tasks in HOMER revolve around the **analyzeHiC** command. Below is a description of how to use it to create matrices. **analyzeHiC** requires a Hi-C tag directory (direction on creating one can be found here). Many of the commands below require a "Hi-C background model" to efficiently normalize and calculate expected interaction counts (If one doesn't exist, it will be created automatically). Creation of Hi-C background models can be found here.

Quick Note about Hi-C Analysis

Hi-C is an unbiased assay of chromatin conformation, resulting in even read coverage across the entire genome. This, coupled with the fact that most Hi-C reads describe interactions at close linear distance along the chromosomes, results in relatively sparse read coverage for interactions between individual restriction fragments separated by great distance. This makes it difficult to find high-resolution interactions across long distances. To help describe how distal and/or inter-chromosomal regions co-localize, two general approaches are utilized:

1. Increase the size of the regions [i.e. decrease the resolution] to boost the number of Hi-C reads used for interaction analysis. This is the most common and simplest technique used by most approaches (including HOMER). It's much easier to identify significant inter-chromosomal interactions between regions that are 100kb in size than regions that are only 5kb in size (essentially 20x more Hi-C reads to work with).
2. Compare the profile of interactions regions participate in when comparing regions. The best example of this is calculating the correlation coefficient of the interaction profiles between two regions. The idea is that if two regions share interactions with several other regions, they are probably similar themselves (even if they do not have direct interaction evidence between them). This forms the basis of the PCA analysis, and uses the transitive property to link regions. If A interacts with C and D, and B interactions with C and D, perhaps A and B interact as well.

Running analyzeHiC

Basic usage:

analyzeHiC <Hi-C Tag Directory> [options] > outputMatrixFile.txt

By default, HOMER generates a normalized interaction matrix and sends it to stdout. There are many other options covered below

Changing the Resolution and Super Resolution

The default resolution is 10000000 (10 Mb). This is to make sure the command finishes quickly if you forget to specify the correct resolution. To specify a different resolution, use "**-res <#>**".

analyzeHiC ES-HiC -res 10000 > output.10kResolution.txt

If the output requires normalization, HOMER will construct a background model for the specified resolution. If it is the first time you've used the designated resolution, HOMER will generate a background model (more on that below). For small resolutions, this can take a while...

HOMER uses two (related) notions of resolution. The first, "-**res <#>**" represents how frequent the genome is divided up into regions. The second, the super resolution "**-superRes <#>**" (poorly named), represents how large the region is expanded when counting reads. Usually the "**-res <#>**" should be smaller than the "**-superRes <#>**". For example a res of 50000 and a superRes of 100000 would mean that HOMER will analyze the regions 0-50k, 50k-100k, 100k-150k etc., but at each region it will look at reads from a region the size of 100k, so it would look at reads from -25k-75k, 25k-125k, 75k-175k, etc. This means HOMER will analyze data with overlapping windows. The principle advantage to this strategy is that you don't penalize features that span boundaries. For example:

analyzeHiC ES-HiC -res 10000 -superRes 20000 > output.10kby20kResolution.txt

Keep in mind that the resolution will also dictate the size of the output matrix. The command will warn you if your parameters seem unreasonable...

Specifying specific Regions to Analyze

By default HOMER will analyze interactions across the entire genome. You can restrict the analysis to a specific chromosome, or part of a chromosome using the following options:

-chr <chr name> : will restrict analysis to this chromosome
-start <#> : starting position for analysis
-end <#> : end position for analysis
-pos <chr:start-end> : UCSC browser formatted position - if you're lazy like me, takes place of the -chr/-start/-end

If you only "**-chr chr1**" and do not specify a start and end, HOMER will simply visualize all of chr1. Regions will be created starting at position 0 to 1*resolution, then from 1*resolution to 2*resolution, etc. (i.e. 0-10kb, 10kb-20kb, 20kb-30kb) If an alternative start is specified, then regions will be created at start+0*resolution to start+1*resolution, start+1*resolution to start+2*resolution, etc (i.e. 205kb-215kb,215kb-225kb,...).

HOMER will normally make a symmetric matrix by default. If you want to specifically look at a matrix between two different regions:

-chr2 <chr name> : will restrict analysis to this chromosome
-start2 <#> : starting position for analysis
-end2 <#> : end position for analysis
-pos2 <chr:start-end> : UCSC browser formatted position - if you're lazy like me, takes place of the -chr2/-start2/-end2
-vsGenome : compare to the rest of the genome

NOTE: Don't use -chr2/-start2/-end2/-pos2/-vsGenome unless you specified something with -chr/-start/-end/-pos etc. first.

Using these regions, HOMER will divide them into #-bp regions, where # is the resolution. HOMER doesn't allow you to cherry-pick several regions from different chromosomes. At least not using the -chr, -start, -end, and -pos options. (You could do this with peaks and the **-p** option).

You can arbitrarily define the regions you want to examine by providing a peak/BED file of the regions. However, these options work a little differently. In the case above (with -chr/-start/-end/-pos), HOMER will chop up the region into resolution sized chunks and perform the analysis. I.e. you provide a locus you want a detailed picture of. When providing a peak file, HOMER will only consider the center of the peak file and the surrounding "resolution-sized" region. It will not chop-up your peaks into resolution-sized chunks (unless you specify the "**-chopify**" option). In general, the **"-p"** option is more useful for looking at all CTCF peaks, for example, to see which are interacting. You can provide a peak/BED that effectively tiles a region, in which case it would mimic how -chr/-start/-end/-pos works, but give you the flexibility to define any region(s) you wanted. To specify a peak file containing regions to analyze with analyzeHiC:

-p <peak/BED file> : peak/BED file to use to search for interactions between.
-p2 <peak/BED file> : A second peak/BED file for non-symetrical matricies.

Keep in mind that an interaction matrix is probably only useful at 2000 x 2000 points - much bigger you can't really visualize it easily anymore (The file will also get really big...)

Matching the resolution of Peak/BED Files and Resolution of analyzeHiC

Another important point, often if you're using Transcription factor peaks for analysis, they may be located less than the resolution apart from one another. (i.e. two PU.1 peaks may be less than 1000 bp from each other, but the resolution for analyzeHiC is 50000) - this means that the two PU.1 peaks will give essentially the same results. To avoid this redundancy, It's best to run **mergePeaks** with a single peak/BED file to collapse peaks within the size of the resolution. For example:

mergePeaks pu1.peaks -d 50000 > newPu1Peaks.txt

The resulting file will contain only peaks at least 50000 bp from one another. Use this resulting file with **analyzeHiC**.

Creating an Interaction Matrix

By default, **analyzeHiC** creates an interaction matrix file. This is a tab-delimited text file formatted to be easy opened with Java Tree View (or any other software that generates Heat Maps). The rows and columns correspond to genomic regions, and the values correspond interaction information between each locus. The type of information shown depends on the options choosen when running **analyzeHiC**. The genomic positions reported correspond to the beginning of the region. Normally, the output is sent to *stdout*, but you can also specify "**-o outputfilename.txt**".

Output Information Options (choose only *one*):

-raw
Outputs the raw interactions counts between the regions

-simpleNorm
Outputs the ratio of observed to expected interactions assuming each region has an equal chance of interacting with every other region in the genome. This normalization assumes each region should contain roughly the same number of total interactions, and essentially controls for the sequencing depth at each region.

-norm (*default*)
Outputs the ratio of observed to expected interactions by assuming each region has an equal chance of interacting with every other region in the genome AND that regions are expected to interact depending on their linear distance along the chromosome. This attempts to take into account the "proximity ligation" effect, where adjact regions are expected to have interactions regardless of the specific 3D genomic structure.

-logp

Outputs the natural log of the p-value describing the likelihood of observeing the actual number of interactions relative to the expected number of interactions between to two loci. This is calculated conservatively as a cumulative binomial distribution.

-expected

Outputs the expected number of interactions instead of the observed number of interactions. To get the "simpleNorm" version of the expected interactions, include "**-simpleNorm**" too.

-corr (can be used with any of the above options)

Instead of outputing the matrix as is, the value of each cell is replaced with the Peason's Correlation Coefficient between the row and column. This can be useful as it adds transitive information to the problem. Instead of just using the number of interaction that directly span between to loci, the correlation option will consider how each region interacts with all of the other loci too. If they have similar interaction profiles, the correlation will be high (i.e. 1). If "**-logp**" or "**-expected**" is used, those values are the ones that will be used for the correlation calculation. The matrix must be symmetric for this option to work.

-nomatrix

Don't create a matrix

Visualizing the Interaction Matrix

The interaction matrix output is a tab-delimited text file. Below is an example opened in Excel (resolution of 1 Mb). The coordinates are given as "chr-position", where the position indicates the **beginning** of the region. If a peak file is given as input, the columns/rows will be labeled with the peak names.

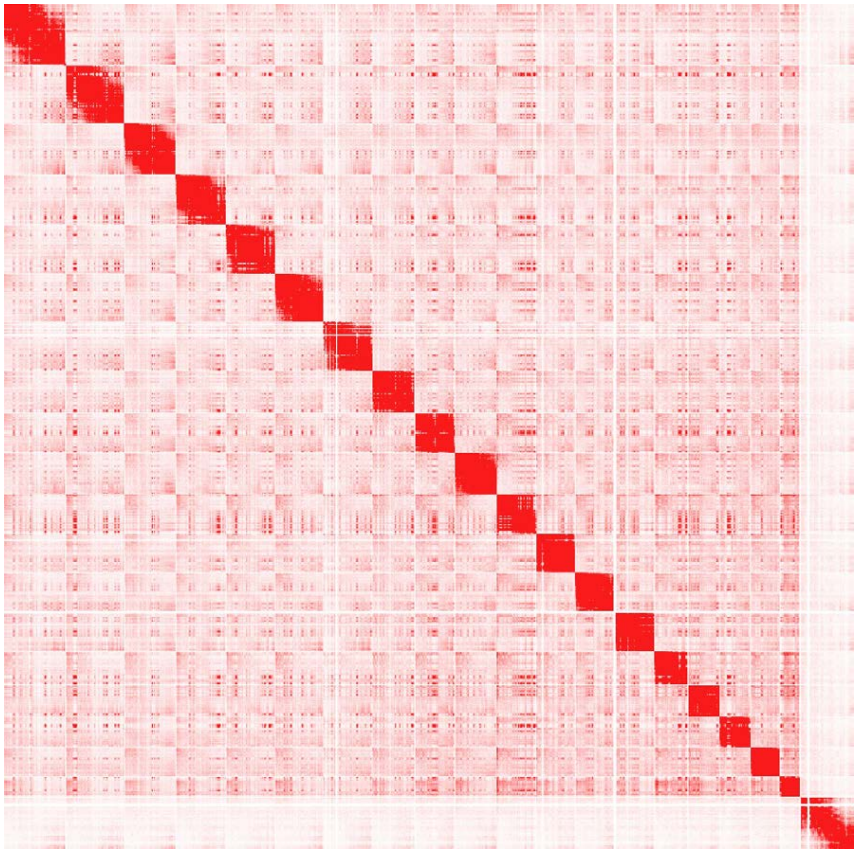
	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P
1	HiCMatrix (director	Regions	chr1-0	chr1-100000(	chr1-200000	chr1-300000	chr1-400000	chr1-500000	chr1-600000	chr1-700000	chr1-800000	chr1-900000	chr1-100000	chr1-110000	chr1-120000	chr1-13000(
2	chr1-0	chr1-0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
3	chr1-1000000	chr1-1000000	0	0	0	0	0	0	0	0	0	0	0	0	0	0
4	chr1-2000000	chr1-2000000	0	0	0	0	0	0	0	0	0	0	0	0	0	0
5	chr1-3000000	chr1-3000000	0	0	0	11740	1329	594	246	276	386	200	233	454	418	100
6	chr1-4000000	chr1-4000000	0	0	0	1329	12240	1564	585	418	313	335	384	308	288	201
7	chr1-5000000	chr1-5000000	0	0	0	594	1564	8768	950	558	477	318	348	338	306	129
8	chr1-6000000	chr1-6000000	0	0	0	246	585	950	15660	2155	380	454	431	237	206	228
9	chr1-7000000	chr1-7000000	0	0	0	276	418	558	2155	9115	1086	574	425	326	260	163
10	chr1-8000000	chr1-8000000	0	0	0	386	313	477	380	1086	8235	994	561	595	377	101
11	chr1-9000000	chr1-9000000	0	0	0	200	335	318	454	574	994	15460	2546	382	288	314
12	chr1-10000000	chr1-10000000	0	0	0	233	384	348	431	425	561	2546	13520	987	437	297
13	chr1-11000000	chr1-11000000	0	0	0	454	308	338	237	326	595	382	987	10170	1358	178
14	chr1-12000000	chr1-12000000	0	0	0	418	288	306	206	260	377	288	437	1358	12080	1406
15	chr1-13000000	chr1-13000000	0	0	0	100	201	129	228	163	101	314	297	178	1406	19170

To view the output as a heatmap, open it in Java Tree View (or any other software that generates Heat Maps). You may have to adjust the pixel settings to get the desired level of brightness. If viewing normalized data of observed vs. expected log2 ratios, you may want to view the matrix so that interacting regions are positive (one color) and non-interacting regions are negative (a different color).

By default, **analyzeHiC** sorts the chromosomes numerically and places the X, Y, and M at the end of the list (You can tell in the example below that the cells are from a male because the X chromosome is lighter than the others...)

Example 1: Raw interactions across the genome at 1 Mb resolution

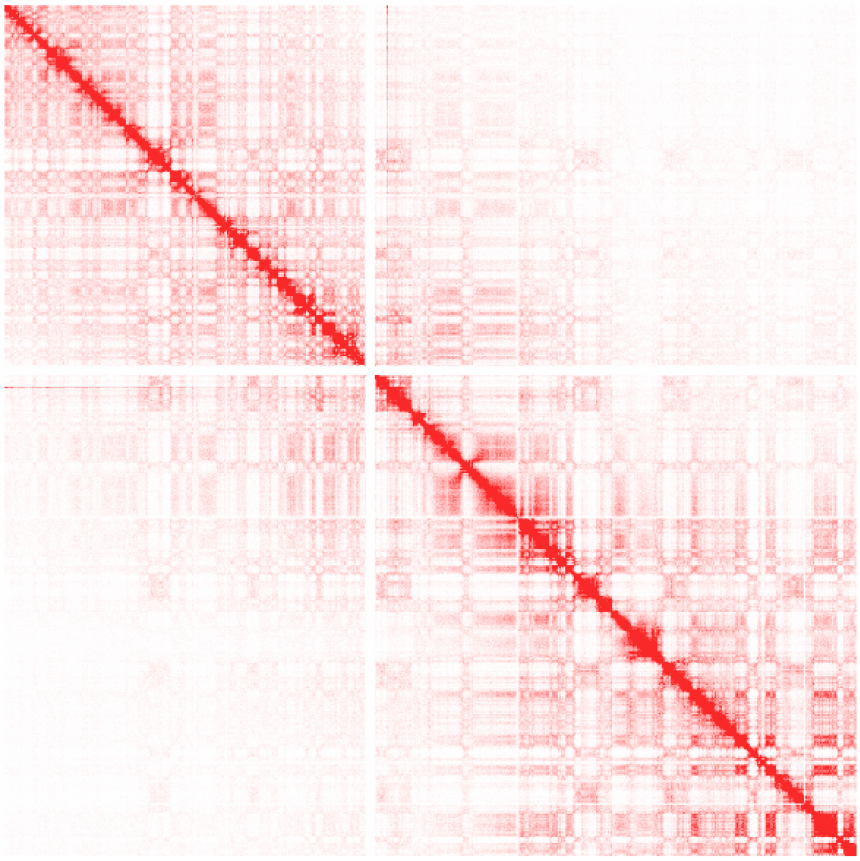
analyzeHiC ES-HiC -res 1000000 -raw > outputFile.txt



As you can see, interactions are much more frequent within chromosomes than between.

Example 2: Raw interactions across chr1 at 100 kb resolution

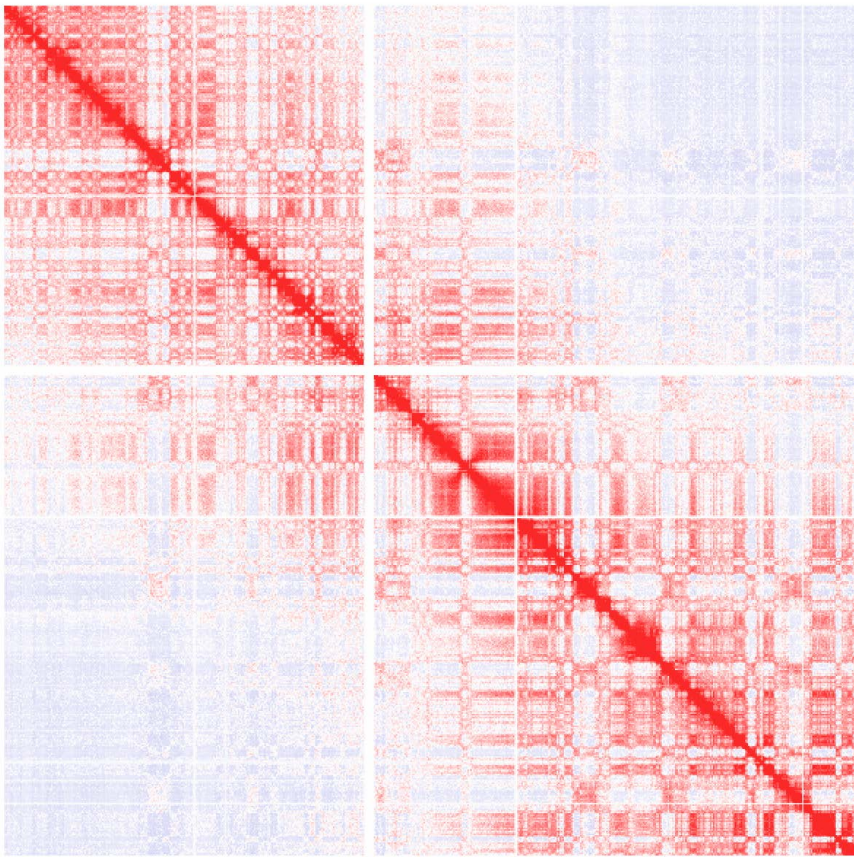
analyzeHiC ES-HiC/ -raw -chr chr1 -res 100000 > outputFile.txt



Here you'll start to notice the compartment patterns.

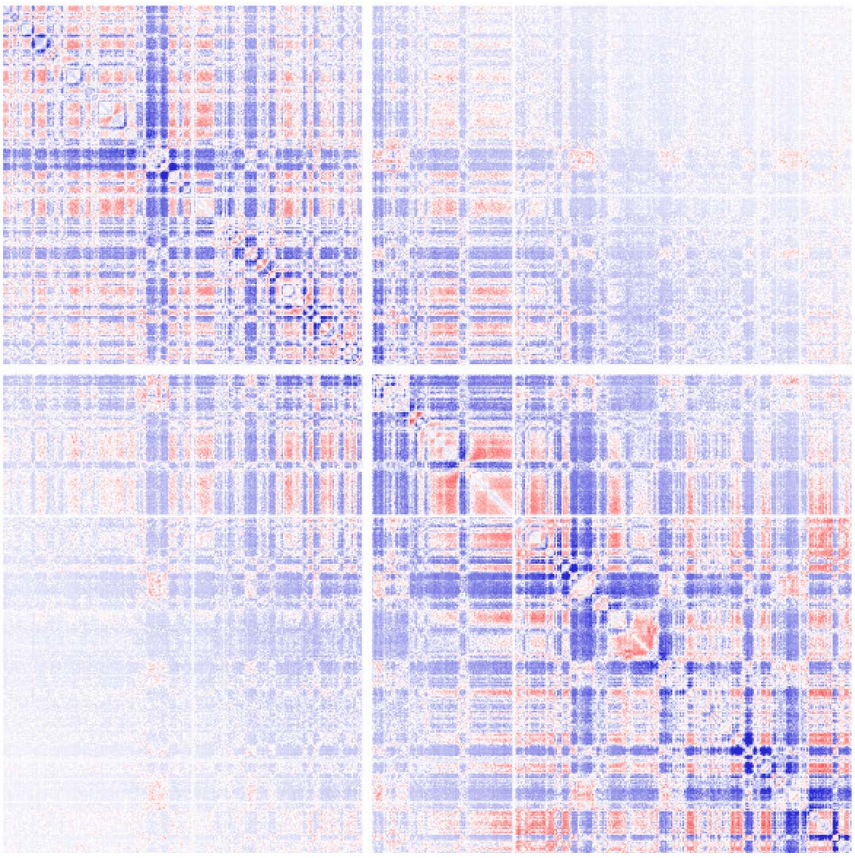
Example 3: Ratio of observed vs. expected interactions across chr1 at 100 kb resolution using simple normalization (normalizing for sequencing coverage only). Visualized as log2 ratios of observed interactions relative to expected interactions (red being positive, blue negative)

analyzeHiC ES-HiC -chr chr1 -res 100000 -simpleNorm > outputFile.txt



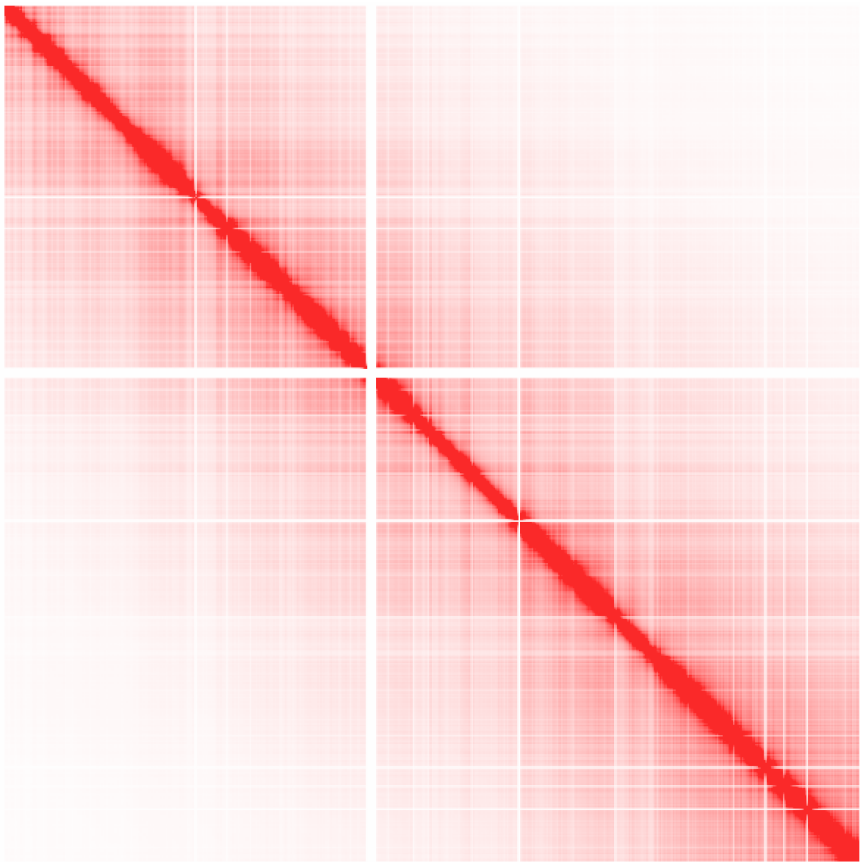
Example 4: Ratio of observed vs. expected interactions across chr1 at 1 kb resolution (normalizing for sequencing coverage and distance between loci - this is the default). Visualized as log2 ratios of observed interactions relative to expected interactions (red being positive, blue negative). NOTE: If a 100kb background model does not exist, it will be created the first time (may take some time).

analyzeHiC ES-HiC -chr chr1 -res 100000 -norm > outputFile.txt



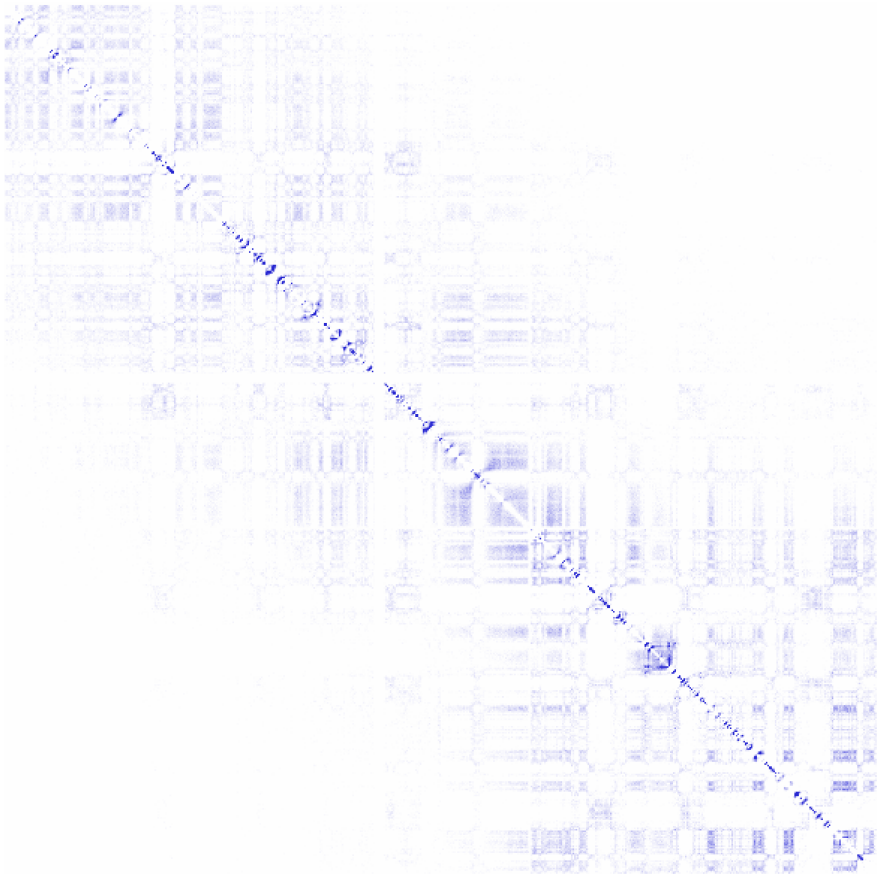
Example 5: You might be curious what the "expected" interactions are from the background model. Use the "**-expected**" option for that:

analyzeHiC ES-HiC -chr chr1 -res 100000 -expected > outputFile.txt



Example 6: log p-values for interactions across chr1 at 100 kb resolution (normalizing for sequencing coverage and distance between loci). More information about the p-value calculation is available in the "Identifying Significant Interactions" section.

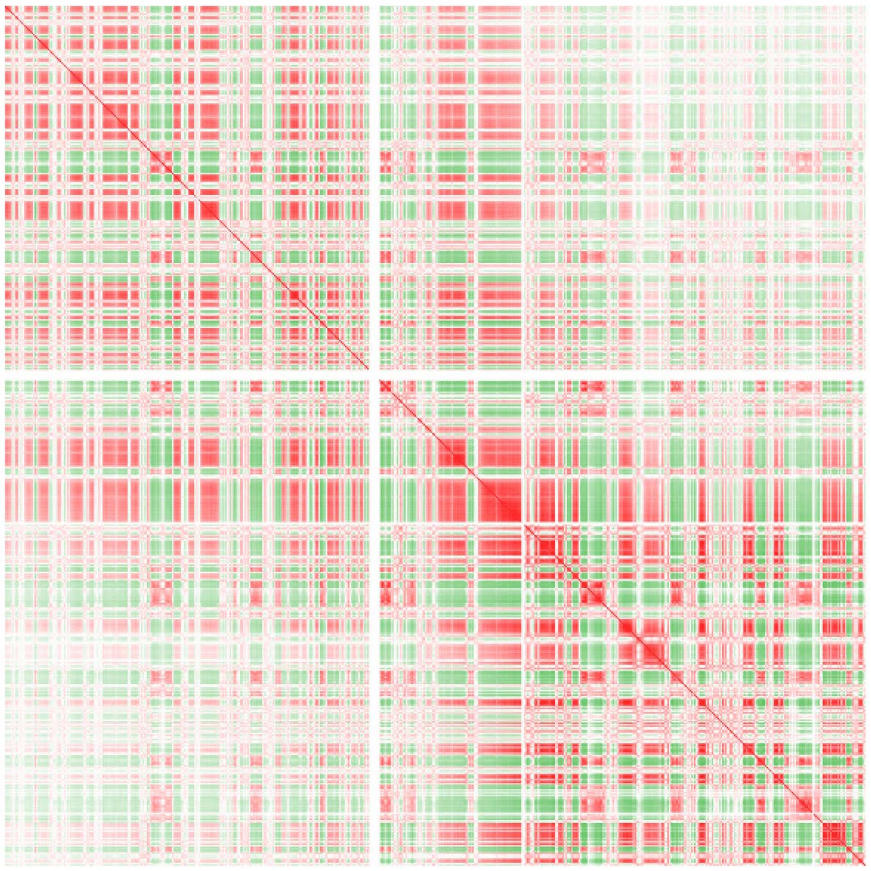
analyzeHiC ES-HiC -chr chr1 -res 100000 -logp > outputFile.txt



As you might notice, most of the significant interactions appear closer to the diagonal where the read coverage is high enough to confidently identify overrepresented interactions.

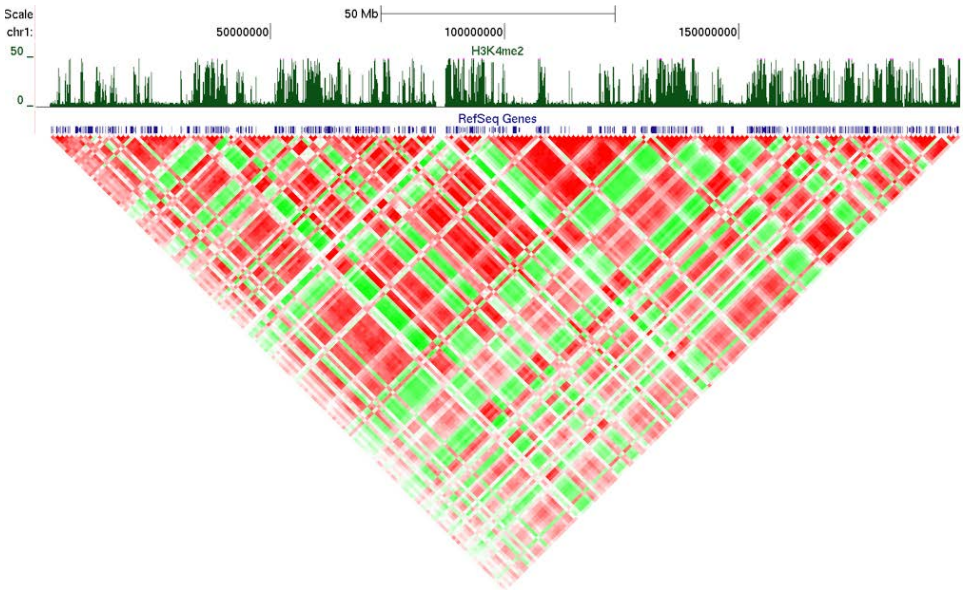
Example 7: Correlation of normalized interaction ratios across chr1 at 100 kb resolution. More about correlation can be found in the "Sub-nuclear compartment analysis/PCA" section. Positive correlation is shown as red, negative correlation as green.

analyzeHiC ES-HiC -chr chr1 -res 100000 -corr > outputFile.txt



Example 8: Poor man's data integration.

Chuck whipped this up faster than you can spell "ROUNDHOUSE". Using powerpoint as professional image arranging software, import a picture such as the one from the previous example, and a picture of a chromosome wide track of H3K4me2 for chr1. You can then rotate the interaction matrix by 45 degrees, and visualize them together to see how all the active regions of the genome "interact" together, and all the inactive regions interact together. Pretty cool...



Example 9: Analyzing Peak Files

When analyzing peak files, use the "**-p <peak/BED file>**". The key difference here is that the matrix that is returned will show each row/column as a peak, and not a continuous section of the genome. Peak regions will be sorted based on their position in the output. For example:

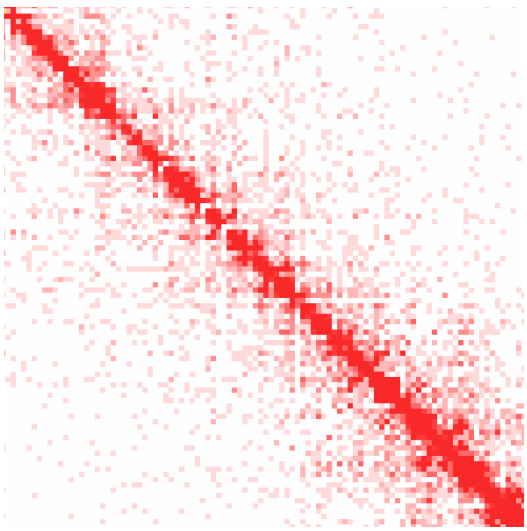
mergePeaks CTCFchr10.peaks.txt -d 50000 > inputPeaks.txt

analyzeHiC ES-HiC -p inputPeaks.txt -res 50000 > outputFile.txt

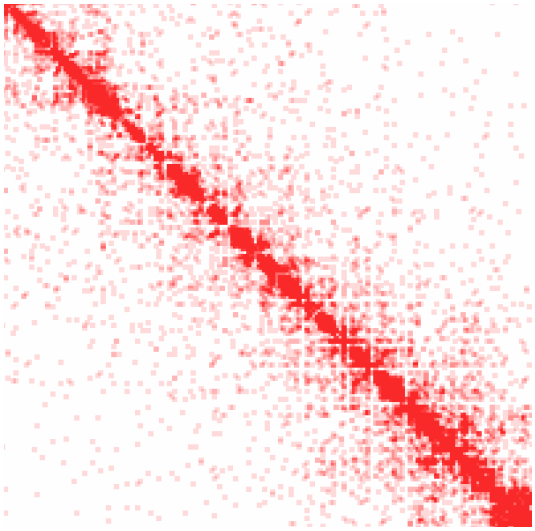
Using -res and -superRes

The "**-superRes <#>**" option is basically a way to make moving windows (explained in more detail at the beginning of this section). In summary, instead of analyzing the data at resolution of 10kb for every 10kb of sequence, you can analyze at a resolution of 10kb (-superRes) but perform that analysis every 1kb of sequence (-res) Normally, you want to superRes to be larger than the resolution, or the same (default). The idea is that it will smooth out features and be more resilient to boundary effects. Below is an example of how it looks with raw interactions.

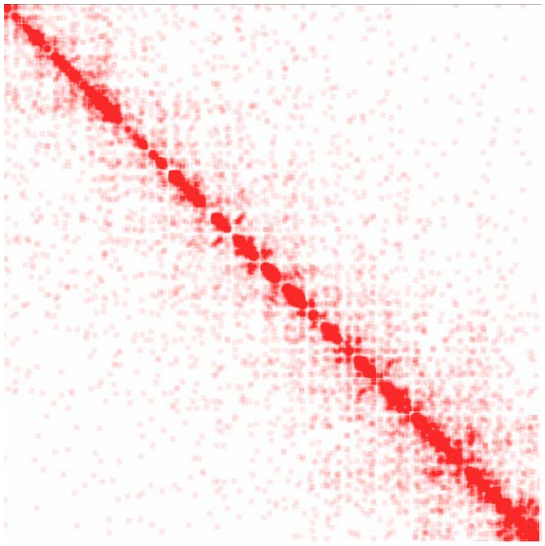
analyzeHiC ES-HiC -res 10000 -superRes 10000 -pos chr1:20000000-21000000 -raw > output.txt



analyzeHiC ES-HiC -res 5000 -superRes 10000 -pos chr1:20000000-21000000 -raw > output.txt



analyzeHiC ES-HiC -res 1000 -superRes 10000 -pos chr1:20000000-21000000 -raw > output.txt



Creating 2D Histograms of Interactions

HOMER can create 2D histograms by examining the interactions in the vicinity of specified locations, such as looking at the interaction profiles around transcription factor peaks, or the Transcription Start Site. The idea is that you specify locations in a peak file (with "-p <peak/BED file>"), specify a resolution and a size, and then HOMER will create several "mini-interaction matrices" around each peak, with the peak position in the center, and in the end adds them together to get a global profile. This is similar to looking at the distribution of a histone mark around a set of transcription factor binding sites, accept in this case you will see a 2D histogram that you would view in the same way as a normal interaction matrix.

To make a 2D histogram, you MUST specify a peak file ("**-p <peak/BED file>**"), a total size of the interaction matrix ("**-size <#>**"), and the histogram resolution ("**-hist <#>**") ("-hist" takes the place of "-res" in this case).

analyzeHiC <HiC Tag Directory> -size <#> -hist <#> -p <peak/BED file> >

output2dHistogram.txt
i.e. **analyzeHiC ES-HiC -size 1000000 -hist 10000 -p tssPositions.txt > output2dHistogram.txt**

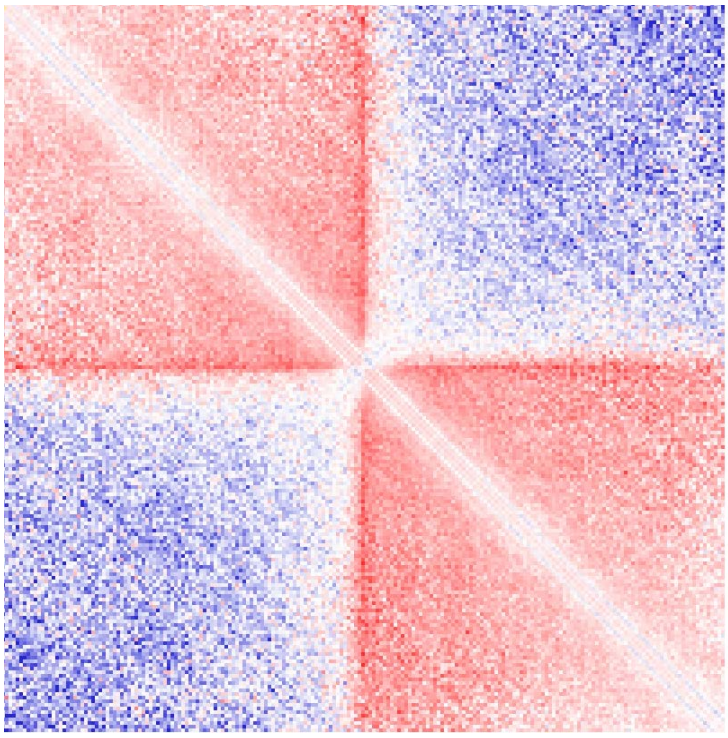
If **-logp**, **-simpleNorm**, etc. are used then those values will be added together for use in the histogram. If you choose to view too many peaks, too large of a size or small of a resolution, the memory requirements can get large, so be careful.

NOTE: If the data is sparse (or analyzed with high resolution, the resulting histogram may have a negative shift (this is due to the pseudo counts that are added to ratios to avoid dividing by zero, etc.). In most cases, it is better practice to make a histogram of raw counts, then make a histogram of expected counts, and then use excel or similar to normalize the values yourself. I will try to integrate this into the command at a later time, don't have time now...

For example:

analyzeHiC ES-HiC -size 1000000 -hist 5000 -p CTCFpeaks.chr10.bed -raw > output1.txt
analyzeHiC ES-HiC -size 1000000 -hist 5000 -p CTCFpeaks.chr10.bed -expected > output2.txt

Then open output1.txt and output2.txt in Excel (or write a quick script) and make a new matrix by dividing each cell in output1.txt by it's value in output2.txt. Save that and visualize:



In this example you can see how CTCF tends to sit at the boundaries between interconnected "domains".

Command Line Options for analyzeHiC

```
Usage: analyzeHiC <PE tag directory> [options]
    ...

Resolution Options:
    -res <#> (Resolution of matrix in bp or use "-res site" [see below], default:
10000000)
    -superRes <#> (size of region to count tags, i.e. make twice the res, default:
same as res)

Region Options:
    -chr <name> (create matrix on this chromosome, default: whole genome)
    -start <#> (start matrix at this position, default:0)
    -end <#> (end matrix at this position, default: no limit)
    -pos chrN:xxxxxx-yyyyyy (UCSC formatted position instead of -chr/-start/-end)
    -chr2 <name>, -start2 <#>, -end2 <#>, or -pos2 (Use these positions on the
        y-axis of the matrix.  Otherwise the matrix will be sysmetric)
    -p <peak file> (specify regions to make matrix, unbalanced, use -p2 <peak file>)
    -vsGenome (normally makes a square matrix, this will force 2nd set of peaks to
be the genome)
    -chopify (divide up peaks into regions the size of the resolution, default: use
peak midpoints)
    -fixed (do not scale the size of the peaks to that of the resolution)
    -pout <filename> (output peaks used for analysis as a peak file)

Interaction Matrix Options:
    -norm (normalize by dividing each position by expected counts [log ratio],
default)
    -zscoreNorm (normalize log ratio by distance dependent stddev, add "-nolog" to
return linear values)
    -logp (output log p-values)
    -simpleNorm (Only normalize based on total interactions per location [log ratio],
not distance)
    -raw (report raw interaction counts)
    -expected (report expected interaction counts based on average profile)
    -corr (report Pearson's correlation coeff, use "-corrIters <#>" to recursively
calculate)
        -corrDepth <#> (merge regions in correlation so that minimum # expected
tags per data point)
    -o <filename> (Output file name, default: sent to stdout)
    -cluster (cluster regions, uses "-o" to name cdt/gtr files, default: out.cdt)
    -clusterFixed (clusters adjacent regions, good for linear domains)
        (# defined the threshold at which to define clusters/domains, i.e. 0.5)
    -std <#> (# of std deviations from mean interactions per region to exclude,
default:4)
    -min <#> (minimum fraction of average read depth to include in analysis,
default: 0.2)

Background Options:
```

-force (force the creation of a fresh genome-wide background model)
-bgonly (quit after creating the background model)
-createModel <custom bg model output file> (Create custome bg from regions specified, i.e. -p/-pos)
-model <custom bg model input file> (Use Custom background model, -modelBg for -ped)
-randomize <bgmodel> <# reads> (and the output is a PE tag file, initail PE tag directory not used
Use makeTagDirectory ... -t outputfile to create the new directory)

Non-matrix stuff:
-nomatrix (skip matrix creation - use if more than 100,000 loci)
-loci <filename> (calculate collective p-value of each loci)
-interactions <filename> (output interactions, add "-center" to adjust pos to avg of reads)
-pvalue <#> (p-value cutoff for interactions, default 0.001)
-zscore <#> (z-score cutoff for interactions, default 1.0)
-minDist <#> (Minimum interaction distance, default: resolution/2)
-maxDist <#> (Maximum interaction distance, default: -1=none)

Making Histograms:
-hist <#> (create a histogram matrix around peak positions, # is the resolution)
-size <#> (size of region in histogram, default = 100 * resolution)

Comparing HiC experiments:
-ped <background PE tag directory>

Creating Circos Diagrams:
-circos <filename prefix> (creates circos files with the given prefix)
-d <tag directory 1> [tag directory 2] ... (will plot tag densities with circos)
-b <peak/BED file> (similar to visiualization of genes/-g, but no labels)
-g <gene location file> (shows gene locations)

Given Interaction Analysis Mode (no matrix is produced):
-i <interaction input file> (for analyzing specific sets of interactions)
-iraw <output BED filename> (output raw reads from interactions, or -irawtags <file>)
-4C <output BED file> (outputs tags interacting with specified regions)
-peakStats <output BED file prefix> (outputs several UCSC bed/bedGraph files with stats)
-cpu <#> (number of CPUs to use, default: 1)

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

HOMER

Software for motif discovery and next-gen sequencing analysis

---

Sub-nuclear Compartment Analysis (PCA/Clustering)

Hi-C data is very complex, and it's interpretation is not trivial. One strategy is to look for general patterns in an unbiased way. This section covers the application two general strategies, Principal Component Analysis and hierarchical clustering. Generally speaking, analysis of Hi-C data using these techniques tend to reveal the genome is spatially segregation in two compartments harboring "active" and "inactive" chromatin.

Principal Component Analysis of Hi-C Data

PCA is a powerful technique for analyzing high dimensional data, and first used on Hi-C data by Lieberman-Aiden et al. The basic idea behind PCA is to redefine the coordinate system such the data can be "described" with as few dimensions as possible (a type of clustering or dimensionality reduction). This axes of the coordinate system are referred to as the principle components, and the "1st" component is found such that it can be used to describe or discriminate as much of the system as possible, with the "2nd" component describing as much of the remaining variance as possible, and so on. We can then consider each region with respect to their values along the first couple principal components to get a simplified view of the data.

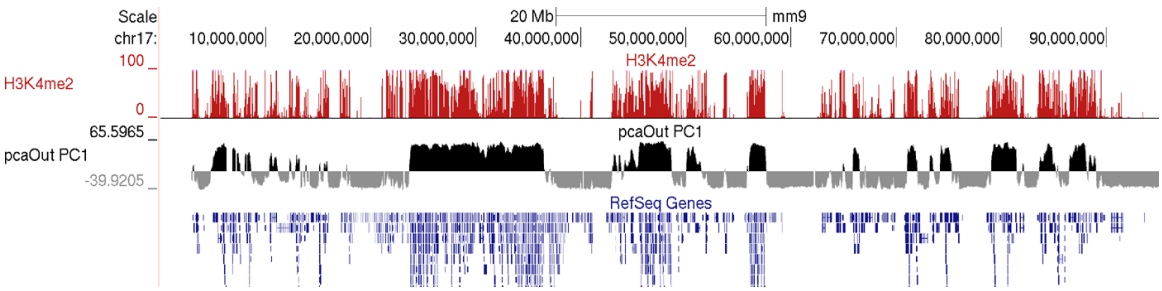
To perform PCA on Hi-C data, we apply it to the normalized interaction matrix. In this setting, each region along the chromosome represents a dimension in the analysis. Normally, PCA is applied to full chromosome matrices to identify important features. You could apply PCA to the entire genome interaction matrix, but sparse interactions between chromosomes may introduce noise. You can also apply PCA to a small region, although the conclusions from such an analysis are less general with respect to general chromosome structure.

To perform PCA analysis with HOMER, use the program **runHiCpca.pl**. Below is an example:

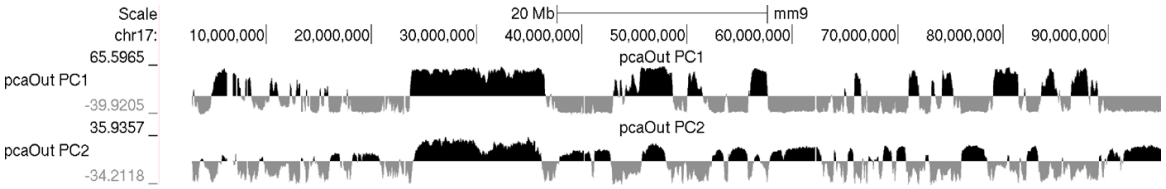
runHiCpca.pl <outputPrefix> <HiC Tag Directory> [options]

example: **runHiCpca.pl pcaOut ES-HiC/ -res 50000 -cpu 4 -genome mm9**

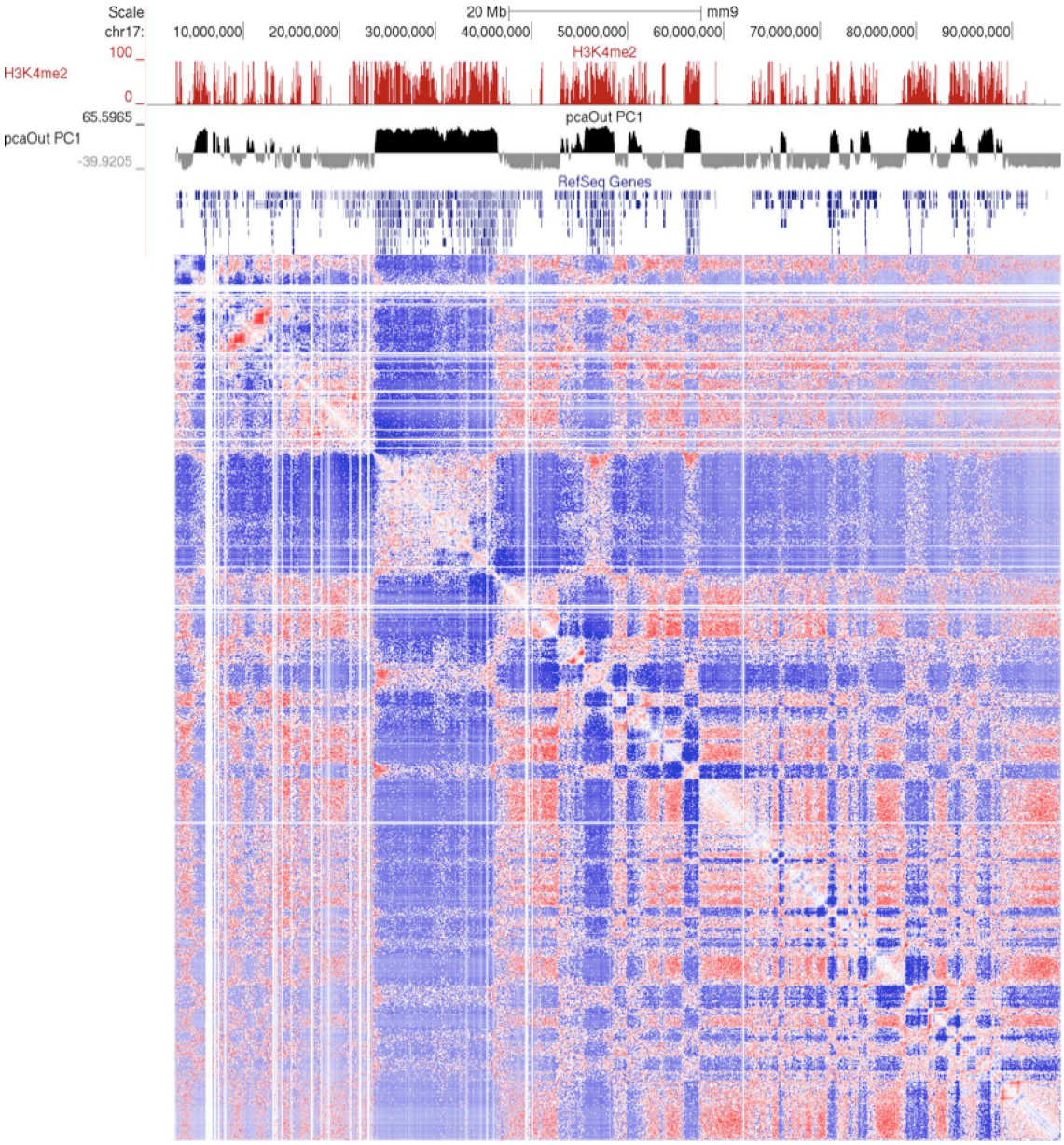
This command will produce two output files: *.PC1.txt and *.PC1.bedGraph (i.e. pcaOut.PC1.bedGraph). The bedGraph file can be directly uploaded to UCSC and represents the coordinates of each region with respect to the first two principle components. The txt file contains the same information in peak file format. Below is an example of the output on chr17:



Usually the values for the first principal component (PC1) are the most informative. These indicate the classification of each region of the chromosome with respect to the first component. Normally, gene-rich regions with 'active' chromatin marks have similar PC1 values, while gene deserts tend to have the opposite. You can also view additional principal components by changing the "**-pc <#>**" option. The 2nd component (PC2) usually describes the the location along the chromosome (e.g. near telomere vs. centromere, or different chromosome arms). The PC2 values tend look a lot like the PC1 values but usually "flip" along the length of the chromosome. Their similarity is partly due to the fact that they are mathematically linked - Each principal component must be orthogonal to one another, so unless the first PC1 perfectly describes a feature of the data, other PCs might reflect the feature as well. In the example below, PC1 and PC2 values are more similar for the first 40 Mb, and then they tend to be more opposite for the rest of the chromosome.



In general, this analysis tends to highlight the fact that the genome can be divided into two compartments, creating a "checkerboard" pattern, with positive PC1 regions reflecting "active/permissive" chromatin and negative PC1 regions indicative of "inactive/inert" chromatin



What does runHiCpca.pl do:

runHiCpca.pl is a wrapper script that will perform PCA analysis on each chromosome individually and combine all the results. Below are each of the steps:

1. Check for the appropriate background model, create it if necessary.

2. For each chromosome:

   - Generate normalized interaction matrix (**analyzeHiC**)
   - Calculate correlation between the contact profiles from each region against each other region
   - Using R, compute the principal components from the correlation matrix
   - Output the values of reach region with respect to the first two principal components

3. If "seed" regions are provided, HOMER checks the overlap between active (and/or inactive) regions to decide if the coordinates should be multiplied by -1 to flip the signs. HOMER will try to make it so regions in "active" zones are positive. If no seed regions are used, but a genome is specified, HOMER will look for

overlap with TSS to assign "active" vs. "inactive".

4. Concatenate results from all chromosomes, format, and print the output

runHiCpca.pl Options:

-pc <#> : Number of principal components to return (default: 1).

-res <#> : resolution of analysis (if it's too small, the program may take too long or use up too much memory). A resolution of 25000 is about the limit for mammalian genomes, unless your computer is ridiculous... The sequencing depth and quality of an experiment may dictate which resolutions are possible. If the PC1 values are very noisy you may need to increase the resolution size. Default: 50000

-superRes <#> : window size of analysis, should be equal to or larger than **-res**. Default: 100000.

-active <peak/BED file> : seed regions of "active" chromatin to use when assessing the proper sign of the PC1 results.

-inactive <peak/BED file> : seed regions of "inactive" chromatin to use when assessing the proper sign of the PC1 results.

-genome <genome> : If no seed regions are given, HOMER will get the TSS locations and treat them as "active" seeds.

-corrDepth <#> : Minimum number of expected reads per region to use when computing correlation (pools adjacent regions), default: 3

-std <#> : exclude regions with sequencing depth exceeding # std deviations, default: 4. The idea here is to remove regions that may throw off the PCA calculation

-min <#> : exclude regions with sequencing depth less than this fraction of mean, default: 0.15. The idea is to get rid of regions with low coverage that don't behave as well during the PCA analysis.

-rpath <path to R executable> : If "R" is not available through the $PATH variable (i.e. can't just type "R" to run R)

-cpu <#> : number of CPUs to use. Memory consumption is also higher with more CPUs (work separated by chromosome). Please note that if multiple CPUs are configured, the screen output will spew all over the place - just ignore it...

Directly Comparing Two Hi-C Experiments

PCA analysis is a useful tool for analyzing single Hi-C experiments. However, comparing PC1 values between two experiments is a little dangerous. PCA is an unbiased analysis of data, and while PC1 values usually correlated with "active" vs. "inactive" compartments, the precise qualitative nature of this association may differ slightly between experiments. Because of this, it is recommended to directly compare the interaction profiles between experiments rather than simply looking at PC1 values.

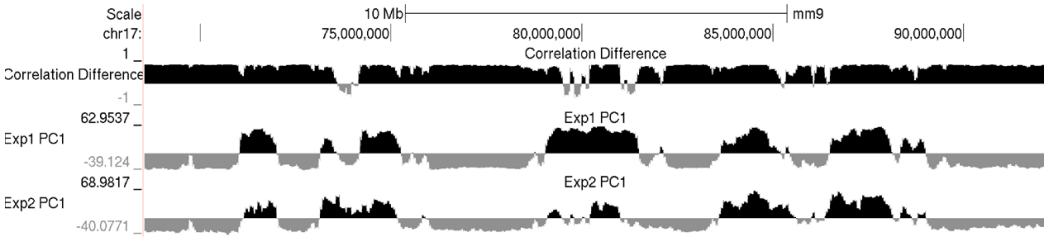
One way to do this is to directly correlate the interaction profile of a locus in one experiment to the interaction profile of that same locus in another experiment. If the locus tends to interact with similar regions in both Hi-C experiments, the correlation will be high. If the locus interacts with different regions in the two experiments, the correlation will be low. Note that this is NOT PCA analysis, cut a direct comparison between two experiments at each locus. By default,

this will compare the interaction profiles across each chromosomes (interchromosomal interactions are generally too sparse for this type of analysis)

To calculate the "correlation difference" between two experiments, use the **getHiCcorrDiff.pl**. Below is how it's used:

> **getHiCcorrDiff.pl <outputPrefix> <HiC Tag Directory1> <HiC Tag Directory2> [options]**
>
> example: **getHiCcorrDiff.pl out ES-HiC Bcell-HiC -cpu 8 -res 50000**

This program will produce two output files, much like **runHiCpca.pl**. One is a bedGraph what can be loaded into the UCSC Genome Browser as a custom track. The other is a peak file with the correlation information in the 6th column. Below is an example of the corrDiff output loaded with the PC1 values from two experiments:



Additional Options for getHiCcorrDiff.pl:

-res <#> : resolution of analysis (if it's too small, the program may take too long or use up too much memory). A resolution of 25000 is about the limit for mammalian genomes, unless your computer is ridiculous... The sequencing depth and quality of an experiment may dictate which resolutions are possible. If the values are very noisy you may need to increase the resolution size. Default: 50000.

-superRes <#> : window size of analysis, should be equal to or larger than **-res**. Default: 100000.

-corrDepth <#> : Minimum number of expected reads per region to use when computing correlation (pools adjacent regions), default: 3

-std <#> : exclude regions with sequencing depth exceeding # std deviations, default: 4. The idea here is to remove regions that may throw off the PCA calculation

-min <#> : exclude regions with sequencing depth less than this fraction of mean, default: 0.15. The idea is to get rid of regions with low coverage that don't behave as well during the PCA analysis.

-maxDist <#> : By default, getHiCcorrDiff.pl compares each locus by looking at it's contact profile across the whole chromosome. Setting this option will limit the profile comparison to the region +/- this distance from the locus (sort of like a 'local' correlation)

-cpu <#> : number of CPUs to use. Memory consumption is also higher with more CPUs (work separated by chromosome). Please note that if multiple CPUs are configured, the screen output will spew all over the place - just ignore it...

Downstream PC1 analysis

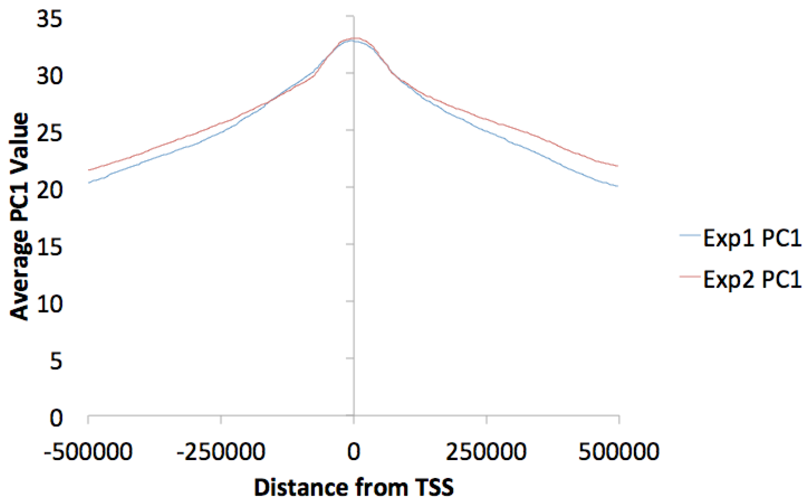
There are several ways to use HOMER to analyze PC1 values (or corrDiff values) with respect to other data. Some are very general ways to manipulate bedGraphs, while others are very specialized to PC1 values (i.e. **getDomains.pl**).

Histograms/Quantifying PC1 values at genomic features

You can make histograms of PC1 values around areas of interest by using **annotatePeaks.pl** and the "**-bedGraph**" option. Lets say you'd like to see the distribution of PC1 values around TSS. (alternatively you could look at PC1 values near ChIP-Seq peaks or other features) Using the command below:

annotatePeaks.pl tss mm9 -size 1000000 -hist 1000 -bedGraph exp1.PC1.bedGraph exp2.PC1.bedGraph > output.histogram.txt

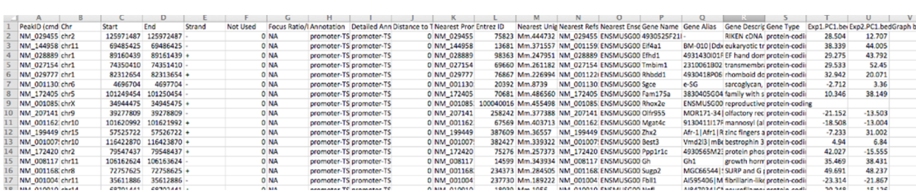
Opening the histogram output in Excel and graphing the "coverage" columns:



You can also omit the "**-hist <#>**" option and adjust the size from the **annotatePeaks.pl** command, and **annotatePeaks.pl** will instead quantify the PC1 values at each locus, which can be useful for classifying which TSS are in the 'active/permissive' compartment vs. which TSS are in the 'inactive/inert' compartment:

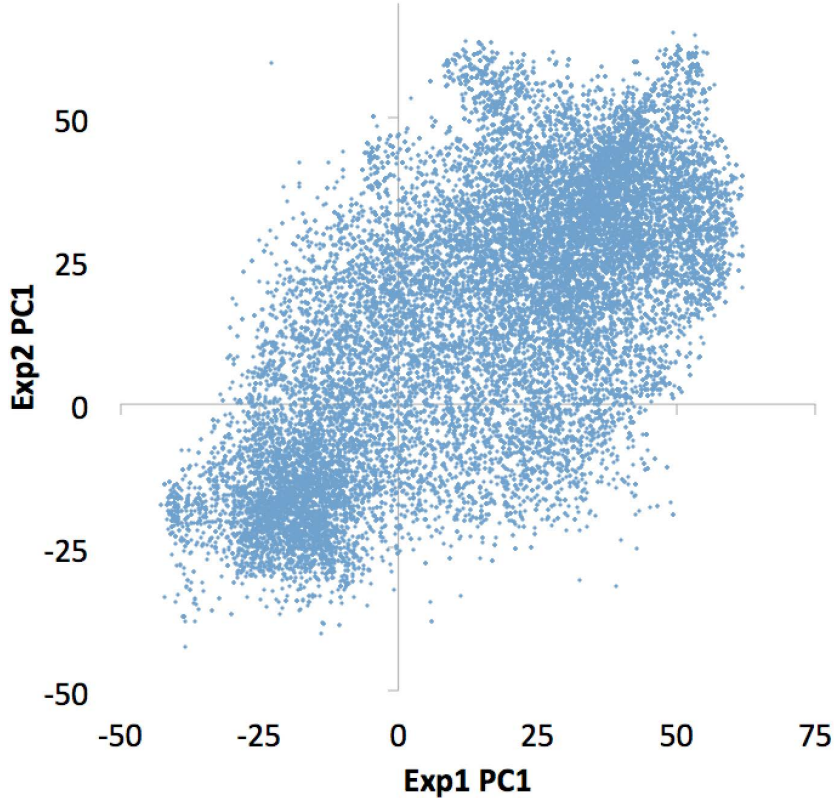
annotatePeaks.pl tss mm9 -size 1000 -bedGraph exp1.PC1.bedGraph exp2.PC1.bedGraph > output.table.txt

Opening this file with Excel:



PeakID (cmd	Chr	Start	End	Strand	Not Used	Focus Ratio/I	Annotation	Detailed Ann	Distance to T	Nearest Pror	Entrez ID	Nearest Unig	Nearest Refs	Nearest Ense	Gene Name	Gene Alias	Gene Descrip	Gene Type	Exp1.PC1.be	Exp2.PC1.bedGraph be
NM_029455	chr2	125971487	125972487	-	0	NA	promoter-TS	promoter-TS	0	NM_029455	75823	Mm.444732	NM_029455	ENSMUSG00	4930525F21I	-	RIKEN cDNA	protein-codi	28.504	12.707
NM_144958	chr11	69485425	69486425	-	0	NA	promoter-TS	promoter-TS	0	NM_144958	13681	Mm.371557	NM_001159	ENSMUSG00	Eif4a1	BM-010|Ddx	eukaryotic tr	protein-codi	38.339	44.005
NM_028889	chr1	89160439	89161439	+	0	NA	promoter-TS	promoter-TS	0	NM_028889	98363	Mm.247951	NM_028889	ENSMUSG00	Efhd1	4931430I01F	EF hand dom	protein-codi	29.275	43.792
NM_027154	chr1	74350410	74351410	-	0	NA	promoter-TS	promoter-TS	0	NM_027154	69660	Mm.261182	NM_027154	ENSMUSG00	Tmbim1	2310061B02	transmembr	protein-codi	29.533	52.45
NM_029777	chr1	82312654	82313654	+	0	NA	promoter-TS	promoter-TS	0	NM_029777	76867	Mm.226994	NM_001122	ENSMUSG00	Rhbdd1	4930418P06	rhomboid do	protein-codi	32.942	20.071
NM_001130	chr6	4696704	4697704	-	0	NA	promoter-TS	promoter-TS	0	NM_001130	20392	Mm.8739	NM_001130	ENSMUSG00	Sgce	e-SG	sarcoglycan,	protein-codi	-2.712	3.36
NM_172405	chr5	101249454	101250454	-	0	NA	promoter-TS	promoter-TS	0	NM_172405	70681	Mm.486560	NM_172405	ENSMUSG00	Fam175a	3830405G04	family with s	protein-codi	10.346	38.149
NM_001085	chrX	34944475	34945475	+	0	NA	promoter-TS	promoter-TS	0	NM_001085	100040016	Mm.455498	NM_001085	ENSMUSG00	Rhox2e	ENSMUSG00	reproductive	protein-coding		
NM_207141	chr9	39277809	39278809	-	0	NA	promoter-TS	promoter-TS	0	NM_207141	258242	Mm.377388	NM_207141	ENSMUSG00	Olfr955	MOR171-34|	olfactory rec	protein-codi	-21.152	-13.503
NM_001162	chr10	101620992	101621992	+	0	NA	promoter-TS	promoter-TS	0	NM_001162	67569	Mm.403713	NM_001162	ENSMUSG00	Mgat4c	9130411I17F	mannosyl (al	protein-codi	-18.508	-13.004
NM_199449	chr15	57525722	57526722	+	0	NA	promoter-TS	promoter-TS	0	NM_199449	387609	Mm.36557	NM_199449	ENSMUSG00	Zhx2	Afr-1|Afr1|R	zinc fingers a	protein-codi	-7.233	31.002
NM_001007	chr10	116422870	116423870	+	0	NA	promoter-TS	promoter-TS	0	NM_001007	382427	Mm.339322	NM_001007	ENSMUSG00	Best3	Vmd2l3|mBe	bestrophin 3	protein-codi	4.94	6.84
NM_172420	chr2	79547437	79548437	+	0	NA	promoter-TS	promoter-TS	0	NM_172420	75276	Mm.257373	NM_172420	ENSMUSG00	Ppp1r1c	4930565M2	protein phos	protein-codi	42.027	-15.555
NM_008117	chr11	106162624	106163624	-	0	NA	promoter-TS	promoter-TS	0	NM_008117	14599	Mm.343934	NM_008117	ENSMUSG00	Gh	Gh1	growth horm	protein-codi	35.469	38.431
NM_001168	chr8	72757625	72758625	+	0	NA	promoter-TS	promoter-TS	0	NM_001168	234373	Mm.284505	NM_001168	ENSMUSG00	Sugp2	MGC66544|	SURP and G	protein-codi	49.691	48.237
NM_001004	chr11	35611886	35612886	-	0	NA	promoter-TS	promoter-TS	0	NM_001004	237730	Mm.189222	NM_001004	ENSMUSG00	Fbll1	AI595406|M	fibrillarin-lik	protein-codi	-23.314	-21.867

Plotting the bedGraph data columns as a scatter plot:



This file can also be used to figure out which individual TSS are changing the most, etc. Similar operations could be performed on ChIP-Seq peaks or other features of interest.

Finding PC1 Based Compartments

Regions of continuous positive or negative PC1 values set the stage for identifying "compartments", in this case referring to linear regions of DNA that are in the "active/permissive" or "inactive/inert" compartments. (This is a little different than the concept of *topological domains* reported in Dixon et al., which are more of a local phenomenon instead of a compartment association). A custom tool is available in HOMER to help with this task named **findHiCCompartments.pl**.

Unlike the **annotatePeaks.pl** command avoid, **findHiCCompartments.pl** works on the *.PC1.txt files that are produced by **runHiCpca.pl**, not the *.PC1.bedGraph files.

To identify compartments, run the command:

findHiCCompartments.pl exp1.PC1.txt > compartments.txt

By default, this command will identify all regions that have a continuous positive PC1 value. The output is a peak file that contains each of these regions, their average PC1 value across the region, and their length in the 6th and 7th columns, respectively. To adjust the threshold used to call compartments, use the "**-thresh <#>**" option. To identify regions with negative PC1 values, add the "**-opp**" option to the command. Also, if you prefer to return compartments as the original sub-peaks (instead of stitching them together into a single, long peak), add the "**-peaks**" option.

Differential Compartments (i.e. Flipping)

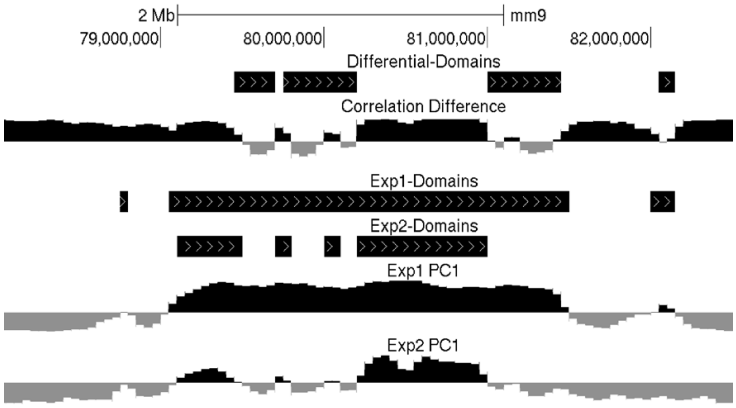
The **findHiCCompartments.pl** command also has some built in

support for identifying regions that change their compartment between experiments. Two sources of information can be used to identify changes: the PC1 values from a second Hi-C experiment, and/or the correlation difference (i.e. getHiCcorrDiff.pl output) between two experiments. If both options are used, the identified regions must pass both criterion.

-bg <PC1.txt file> : specify a background experiment's PC1 results.
-diff <#> : minimum difference between PC1 values to define region as different, default: 50

-corr <corrDiff.txt file> : specify a correlation difference file
-corrDiff <#> : maximum correlation for a region to be defined as different, default: 0.4

Below is an example of these files uploaded to the UCSC Genome Browser. NOTE: If UCSC complains that you chromsome coordinates exceed the end of the chromosome (could happen due to the resolution), run the command "**removeOutOfBoundsReads.pl compartments.txt <genome> > compartments.new.txt**" to adjust the coordinates at the end of the chromosome.



Clustering Regions Based on their Interactions

HOMER also offers more "direct" ways to group regions (most of these are experimental). For now, it's recommended you stick to the PCA analysis from above, however, there may be situations where the routines below are more useful.

analyzeHiC has two types of clustering routines to help identify sets of regions that are "related" in 3D-space. They are:

-cluster : Clustering of regions regardless of genomic locations (pure clustering based on interaction frequency)

-clusterFixed : Clustering of regions based on adjacent, linear, regions on the chromosome (for finding "linear domains")

The clustering is performed as hierarchical clustering, and the output is stored in files "out.cdt" and "out.gtr". You can use Java Tree View to open the "out.cdt" file and view the clustering result. From there you can select your own clusters if you like. Selection of "**-corr**", "**-logp**" or "**-simpleNorm**" etc. during the command will cause those values to be used in the clustering instead of the default

("**-norm**").

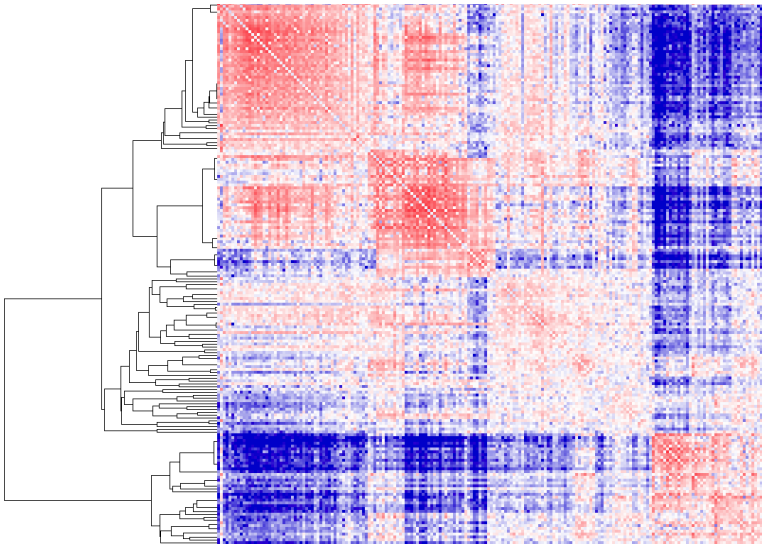
To name the clustering output something other than "out", use "**-o <filename>**", which will use <filename> as the prefix for the clustering files instead of "out".

Example of -cluster

As an example, lets try clustering the normalized matrix of chr1 at 1 Mb resolution. Using the command:

analyzeHiC ES-HiC/ -chr chr1 -res 1000000 -cluster > outputmatrix.txt

This produces several files, named out.cdt and out.gtr (and outputmatrix.txt, which we'll ignore). Opening out.cdt with Java Tree View will give us:

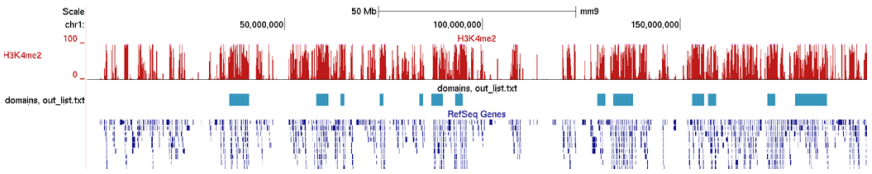


In this example, we allow "free clustering", and the algorithm will group together loci that are considered "close together" - in this case, loci whose interaction profiles have high interaction log2 ratios. As you can see, there are essentially two groups. To visualize which regions are in the groups, highlight the sub-tree of interest, and then export the list (under Export->Save List in Java Tree View). HOMER contains a tool called cluster2bed.pl to allow you to visualize these clusters in the genome browser. Run the following command on the saved list file (be default this is called "out_list.txt"):

cluster2bed.pl <cluster list> <#resolution> [# minimum to cluster size to keep] > out2.bed
i.e. cluster2bed.pl out_list.txt 1000000 > out.bed

You can now upload the output file to the UCSC genome browser. The 3rd option in the **cluster2bed.pl** command is optional, which will remove clusters containing only a couple regions (be default, it removes cluster containing fewer than 5% of the total). Colors are randomly assigned to the clusters to help differentiate them.

Visualizing these clusters in the UCSC genome browser along with H3K4me2 ChIP-Seq and the interaction matrix, you can see that the 2 major clusters from above correlate with H3K4me2 very nicely (would look better at higher resolution, say 100kb)

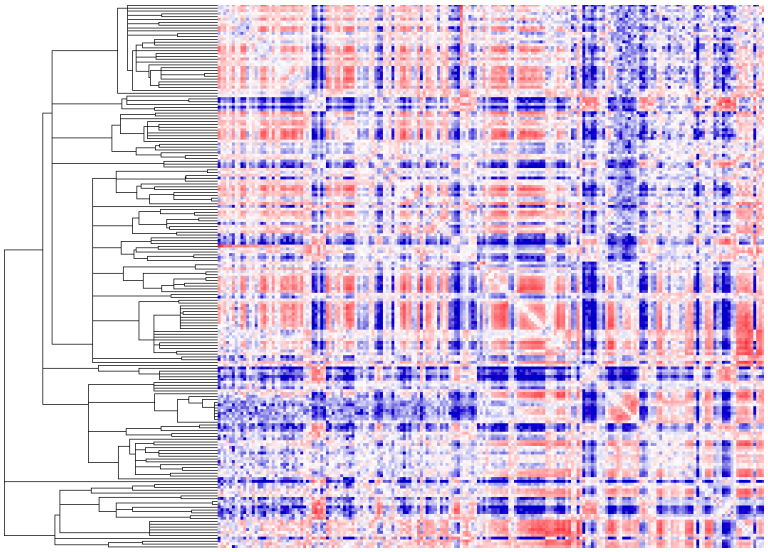


Example of -clusterFixed

As an example, lets try clustering again, this time using "fixed" clustering, which will force adjacent positions to be grouped. Using the command:

analyzeHiC ES-HiC/ -chr chr1 -res 1000000 -clusterFixed > outputmatrix.txt

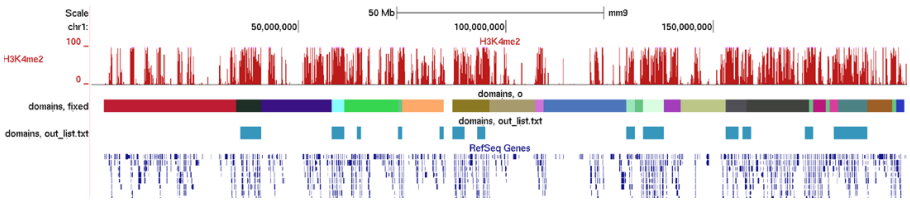
This produces the same basic output files as "**-cluster**", named out.cdt and out.gtr (and outputmatrix.txt, which we'll ignore). Opening out.cdt with Java Tree View will give us:



Here you'll notice that the heatmap look just as it would if you did not cluster the data. The only difference is that it grouped regions together based on there similarity. This can be useful for finding linear domains. To extract domains from the clustering result, you must first choose a cutoff threshold to identify cluster (i.e. 0.5 - related to correlation) and use the "**homerTools cluster**" program, and then use the **cluster2bed.pl** tool to format for the UCSC Genome Browser:

homerTools cluster -gtr out.gtr -thresh 0.5 > cluster.txt
cluster2bed.pl cluster.txt 1000000 > out.bed

Uploading this to the browser will give us:



Command Line Options for runHiCpca.pl

Usage runHiCpca.pl <output prefix> <HiC directory> [options]

Options:
-res <#> (resolution in bp, default: 50000)
-superRes <#> (super resolution in bp, i.e. window size, default: 100000)
-active <K4me+ regions> (Regions to use to help decide sign on principal component [active=+])
-inactive <K4me- regions> (Regions to use to help decide sign on principal component [inactive=-])
-genome <genome> (If you don't have seed regions, this will use the TSS file as active seeds)
-corrDepth <#> (number of expected reads needed per data point when calculating correlation, default: 3)
-std <#> (exclude regions with sequencing depth exceeding # std deviations, default: 4)
-min <#> (exclude regions with sequencing depth less than this fraction of mean, default: 0.15)
-rpath <path to R executable> (If R is not accessible via the $PATH variable)
-cpu <#> (number of CPUs to use, default: 1)

Output files:
<prefix>.PC1.txt - peak file containing coordinates along the first 2 principal components
<prefix>.PC1.bedGraph - UCSC upload file showing PC1 values across the genome

Command Line Options for getHiCcorrDiff.pl

Usage getHiCcorrDiff.pl <output prefix> <HiC directory1> <HiC directory2> [options]

Options:
-res <#> (resolution in bp, default: 50000)
-superRes <#> (super resolution in bp, i.e. window size, default: 100000)
-corrDepth <#> (number of expected reads needed per data point when calculating correlation, default: 3)
-std <#> (exclude reads with sequencing depth exceeding # std deviations, default: 4)
-min <#> (exclude reads with sequencing depth less than this fraction of mean, default: 0.1)
-maxDist <#> (maximum distance around regions to calculate similarity metrics, default: none)
-cpu <#> (number of CPUs to use, default: 1)

Output files:
<prefix>.corrDiff.txt - peak file containing correlation values for each region
<prefix>.corrDiff.bedGraph - UCSC upload file showing correlation values across the genome

Command Line Options for getDomains.pl

Usage: getActiveDomains.pl <PC1.txt file> [options]

Options:
-opp (return inactive, not active regions)
-thresh <#> (threshold for active regions, default: 0)
-bg <2nd PC1.txt file> (for differential domains)
-diff <#> (difference threshold, default: 50)
-corr <corrDiff.txt file> (for differential domains)
-corrDiff <#> (correlation threshold, default: 0.4)
-peaks (output as peaks/regions, not continuous domains)

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 Significant Interactions in Hi-C Data

HOMER can search for pairs of loci that have a greater number of Hi-C reads that expected by chance, which below will be referred to as a 'significant interaction'. The enhanced proximity of these regions may have relevant biological interpretation. It is important to note up front that it is basically unheard of for two loci to ALWAYS co-localize next to one another (i.e. in same cross-linked complex). Usually regions simply show "enrichment" for their co-localization, meaning that evidence for their co-fixation comes from only a fraction of the total cells used in the experiment. This also means that some regions may for 'significant interactions' with multiple other loci in the same experiment.

The section below describes the HOMER commands that can be used to find significant interactions from Hi-C experiments, followed by routines and techniques to analyze their biological significance. The next section describes how to visualize interactions in Circos Diagrams. The section after that describes the analysis of interactions with respect to other features and their annotation. These sections focus on the analysis of specific interactions between any two loci, while the section after that describes the use of Structured Interaction Matrix Analysis (SIMA), which boosts analysis power by considering interactions between multiple regions simultaneously.

Finding Significant Interactions in Hi-C Data

To find significant interactions between any two loci with HOMER, use the **analyzeHiC** command with the option "**-interactions <outputfilename.txt>**". This will trigger homer to perform significance calculations between each loci described in the command. For example:

analyzeHiC ES-HiC -res 1000000 -interactions significantInteractions.txt -nomatrix

This command will search for significant interactions in the genome at 1Mb resolution. The "**-nomatrix**" is optional, but since the default of **analyzeHiC** is to produce a matrix, you may want (or need) to stop creation of the matrix as it could be very large and consume too much memory. Opening the "significantInteractions.txt" files with Excel will reveal a file that looks like:

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T
1	InteractionID	PeakID(1)	chr(1)	start(1)	end(1)	strand(1)	Total Reads(1)	PeakID(2)	chr(2)	start(2)	end(2)	strand(2)	Total Reads(2)	Distance	Interaction Reads	Expected Reads	Z-score	LogP	FDR(Benjamini,	Circos Thickness
2	interaction1	chr18-55000000	chr18	55000000	56000000	+	65369	chr18-54000000	chr18	54000000	55000000	+	59156	827404	8017	2844.717409	2.758042	-3384.2552	0	38
3	interaction2	chr2-46000000	chr2	46000000	47000000	+	49538	chr2-44000000	chr2	44000000	45000000	+	59178	1933194	3508	736.558047	2.094374	-2788.863	0	32
4	interaction3	chr13-6000000	chr13	6000000	7000000	+	53092	chr13-4000000	chr13	4000000	5000000	+	39340	1732151	3339	691.382485	2.11301	-2707.9667	0	30
5	interaction4	chr15-63000000	chr15	63000000	64000000	+	54049	chr15-61000000	chr15	61000000	62000000	+	58833	1802098	3546	772.795341	2.044411	-2707.5473	0	30
6	interaction5	chr16-93000000	chr16	93000000	94000000	+	53664	chr16-92000000	chr16	92000000	93000000	+	61076	672241	7050	2786.411658	2.470993	-2469.4225	0	28
7	interaction6	chr18-56000000	chr18	56000000	57000000	+	59550	chr18-54000000	chr18	54000000	55000000	+	59156	1845562	3352	831.989798	1.869891	-2210.896	0	24
8	interaction7	chr8-118000000	chr8	118000000	119000000	+	62988	chr8-117000000	chr8	117000000	118000000	+	66052	812627	7438	3193.567647	2.250585	-2203.2677	0	24
9	interaction8	chr10-17000000	chr10	17000000	18000000	+	55883	chr10-16000000	chr10	16000000	17000000	+	49590	764255	6246	2458.540337	2.48187	-2197.3447	0	24
10	interaction9	chr10-96000000	chr10	96000000	97000000	+	56336	chr10-95000000	chr10	95000000	96000000	+	57888	667928	6394	2582.466989	2.413316	-2128.5783	0	24
11	interaction10	chr13-83000000	chr13	83000000	84000000	+	53013	chr13-82000000	chr13	82000000	83000000	+	41127	812535	5307	1980.919726	2.623121	-2053.9882	0	22
12	interaction11	chr13-23000000	chr13	23000000	24000000	+	40780	chr13-21000000	chr13	21000000	22000000	+	53931	2038903	2815	650.968731	1.964692	-2021.8847	0	22
13	interaction12	chr3-81000000	chr3	81000000	82000000	+	51704	chr3-80000000	chr3	80000000	81000000	+	47782	735626	5306	2066.007459	2.510682	-1887.0592	0	20
14	interaction13	chr4-95000000	chr4	95000000	96000000	+	56037	chr4-94000000	chr4	94000000	95000000	+	57774	665429	6062	2533.981116	2.321825	-1883.2354	0	20

This file is formatted as a HOMER interaction file. It is a tab-delimited text file with a header and one interaction per line. The definitions of the columns are as follows:

HOMER Interaction File Format Column Types (tab-separated text file)

1) Interaction ID (must be unique)

2) Peak ID for region 1
3) chr for region 1
4) start position for region 1
5) end position for region 1
6) strand for region 1
7) total reads for region 1

8) Peak ID for region 2
9) chr for region 2
10) start position for region 2
11) end position for region 2
12) strand for region 2
13) total reads for region 2

14) Distance between regions (or "interchromosomal")
15) Interaction Reads (total Hi-C reads connecting the regions)
16) Expected Interaction Reads (total expected Hi-C reads based on background model)
17) Modified Z-score
18) Natural log of the p-value for the interaction (binomial)
19) False Discovery Rate (based on Benjamini correction)
20**) Circos Thickness (used for visualization by Circos)

The interaction file can be further analyzed with commands in the sections below, or it can be the starting point of your own custom analysis.

How HOMER finds Significant Interactions

The premise behind finding significant interactions is simple enough: Look for pairs of regions that have more Hi-C reads between them than would be expected by chance. The expected number of reads is calculated using the background model (covered here). The background model is used to model how many reads we expect to connect to each other region in the genome. If regions are far away (or on separate chromosomes), the we expect only a small number of reads to connect them. Likewise, if two regions are close to one another, we would expect a large number of interactions. These expectations are also dependent on the number of total reads mapping to each locus (which may depend on the number of restriction sites etc. in the region). The background model attempts to take all of these factors into account.

Since the total number of reads per region is fixed (and more or less constant for each region due to the unbiased nature of Hi-C), we test how these reads are distributed relative to the expectation. For two given loci that could potentially interact, we model their randomly expected read counts using the cumulative binomial distribution, where the total number of trials is the number of reads that could possibly map between the loci (i.e. the region total), the rate of success is the expected interaction frequency, and the number of observed successes is the number of observed reads mapping between the loci. In this setting, regions with only 1 or 2 reads between them will have high p-values, regardless of their expected interaction frequency, while regions with many interactions above expected will have low p-values.

Parameters to Consider when Finding Significant Interactions

Regions to Analyze

The **analyzeHiC** command will look for interactions across the entire genome unless otherwise specified. To specify a specific subset of the genome to analyze, use the "**-chr/-start/-end/-pos/-p**" options.

Super resolution and congruent interactions

When using a "**-superRes <#>**" value that is larger than the resolution ("**-res <#>**"), HOMER analyzes the Hi-C data in overlapping windows. If there is a truly significant interaction in the data, there is a good chance that multiple overlapping windows may observe the same reads and call the interaction multiple times with slightly different offsets. By default, HOMER looks through the identified interactions before writing them to the output and removes overlapping interactions (defined as overlapping at both endpoints), keeping only the interaction with the most significant logP value.

Limiting the Search Space

The further apart regions get, the less reads that are likely to map between them, and the less likely that a significant interaction is to be found. Depending on the resolution and the sequencing depth, it can be next to impossible to identify "significant interactions" past a certain distance. To control the space that is searched, specify "**-maxDist <#>**" or "**-minDist <#>**". This can dramatically speed up the search for high resolution interactions (i.e. "-res 10000 -maxDist 3000000").

Increasing Accuracy by Centering Interactions

If you add the option "**-center**", analyzeHiC will replace the coordinates of the regions in the output file by re-centering the regions on the average of the position of the Hi-C reads participating in the interaction. Lets say a region spans from position 10,000 to 20,000 (10k resolution), and it interacts with the region at 100,000 to 110,000. If most of the interacting regions originate from a regulatory

elements at 17,000, the output file will be re-centered such that the output coordinates are 12,000 to 22,000 for the first region.

Interaction Reporting Filtering

By default, HOMER will report all interactions with a p-value less than 0.001. To modify this threshold, use "**-pvalue <#>**". You can also change the modified z-score cutoff using the parameter "**-zscore <#>**".

Multiple CPUs

Using the "**-cpu <#>**" option will speed up the analysis if analyzing multiple chromosomes.

Finding Differential Interactions between two Hi-C Experiments

To assess the interactions in a second Hi-C experiment, use the "**-ped <HiC Tag Directory>**" option. When this command is used with the "**-interaction <outputfile.txt>**" option, interactions are first found just like normal. After interactions are found, HOMER will go back and quantify the Hi-C reads for each interaction in the second Hi-C experiment. Independent statistics will be calculated for that experiment based on it's background model, and then it will be compared to the first Hi-C experiment. For example:

analyzeHiC ES-HiC/ -res 100000 -ped MEF-HiC/ -interactions significantInteractions.txt -nomatrix

The output file is slightly different in the case of a background Hi-C experiment. There are 8 extra columns as described below:

20**) Background Experiment Interaction Reads
21) Background Experiment Expected Reads
22) Background Modified Z-score
23) Background LogP
24) Background total reads region 1
25) Background total reads region 2
26) LogP of Primary Experiment vs. Background
27) Modified Z-score of Primary Expeirment vs. Background
28) Circos Thickness (used for visualization by Circos)

(Column 20 is the Circos Thickness normally without specifying "**-ped <tag dir>**", see above)

Note that the background Hi-C experiment is not used to "find" interactions, just score the ones found in the primary Hi-C experiment. To find interactions that are specific to the background Hi-C experiment, reverse their positions in the command.

Finding Intra-chromosomal Interactions Genome-wide at High Resolution

A common task is to find all the interactions in the genome. With more sequencing and better data quality, the hope is to find interactions between individual regulatory elements, at a resolution of 10kb or less (maybe 1kb or 500bp! depending on restriction enzyme used). At these small resolutions it is near impossible to identify significant inter-chromosomal interactions, and the search space for inter-chromosomal interactions is massive.

To speed up the process, HOMER will look at chromosomes individually which allows it to save on resources. A separate program called **findHiCInteractionsByChr.pl** automates this process and helps speed up the calculation. Here's an example:

findHiCInteractionsByChr.pl ES-HiC/ -res 2000 -superRes 10000 -cpu 8 > outputInteractions.txt

The first argument MUST be the Hi-C tag directory. The output interaction formatted file is sent to *stdout*, so make sure to capture the output in a file. Other options are as follows:

-res <#> : resolution of analysis, see above, default: 2000

-superRes <#> : super resolution (window size), see above, default: 10000

-minDist <#> : Minimum distance between regions to consider for an interaction (default: -superRes value)

-maxDist <#> : Maximum distance between regions to consider for an interactions (default: 10,000,000) Changing this parameter may change the running time a bit

-pvalue <#> : pvalue cutoff, default: 0.01

-zscore <#> : modified z-score cutoff, default: 1.5

-cpu <#> : number of CPUs to use, default: 1 (remember that more CPUs will require more memory too)

-ped <background HiC directory> : Will quantify background experiment reads at significant interactions.

-std <#> : exclude regions with sequencing depth exceeding # std deviations, default: 4)

-min <#> : exclude regions with sequencing depth less than this fraction of mean, default: 0.2)

Miscellaneous Interaction Actions

Quantifying Hi-C reads Given Predetermined Interactions

Lets say you have known interactions, or interactions from another experiment. You can supply these as an input file to **analyzeHiC** and it will quantify their read counts and significance for you. The input file should look exactly like the HOMER formatted interaction file. It should be a tab-delimited text file. However, most of the columns can have a value of 0. The only values that MUST have values are the interaction IDs and region information (columns 1-12). The read totals to not need to be specified - HOMER will fill that in. To have HOMER analyze you given interactions, use the "**-i <inputInteractions.txt>**" option. Below is an example:

```
analyzeHiC ES-HiC -i inputInteractions -res 100000 -interactions
outputInteractions.txt -nomatrix
```

The interaction quantification will be in the new outputInteractions.txt file from this example.

Retrieving Hi-C reads at Interacting Loci

To investigate the exact evidence that contributes to a given interaction or set of significant interactions, use the "**-i <inputInteractions.txt>**" option to specify your interactions of interest in conjunction with the "**-iraw <outputFile.bed>**" or "**-irawtags <outputFile.tags.tsv>**" options. These options will output the raw interactions from the Hi-C experiment that participate in the interaction in the input file. The "-iraw" output file is in BED format and can be uploaded to the UCSC Genome Browser to visualize your interactions.

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 and Analyzing Significant Interactions From Hi-C Data

Once you have found significant interactions in your data, it's time to figure out what it means. HOMER contains a tool called **annotateInteractions.pl** that can be used to execute a variety of different types of analysis (analogous to **annotatePeaks.pl** in some ways).

The **annotateInteractions.pl** command takes a HOMER-style interaction file as input (see here). The interactions do not need to be produced by HOMER - they can come from any tool or created manually, but they must have the same format. The basic syntax of the command is as follows:

> **annotateInteractions.pl <interaction file> <genome version> <output directory> [additional options...]**
>
> example: **annotateInteractions.pl bcell-Interactions.txt mm9 AnnotationOutputDirectory/**

By default, this command will produce a bunch of output files and place them in the given output directory. Below is a description of the default output files. Additional options enable the assessment of feature enrichment, and are covered further down.

Many of the options for the **annotateInteractions.pl** command are used to filter the interactions such that a subset is analyzed or annotated. Filtering options are described below. One of the main purposes for the filtering options is so that liberal cutoffs for the p-value and z-score can be selected when running the **analyzeHiC** or **findHiCInteractionsByChr.pl** command to find the initial set of interactions. Once this large list of possible interactions is found, the p-value and z-score cutoffs can be changed/optimized in the **annotateInteractions.pl** command to avoid rerunning the other commands, which can be very time consuming.

Default Annotation Output:

Each of these output files will be placed in the chosen output directory

interactions.txt

This file is a "HOMER interaction" formatted file that contains the interactions used in the analysis. Several of the **annotateInteractions.pl** options are used to filter the input interactions, and this file will only contain the interactions that pass the filters and used for analysis and annotation.

interactionAnnotation.txt

This file is an extension of the HOMER interaction format that includes 16 additional columns with annotation information describing the regions at each end of the interaction:

Column#) Description(Peak/Region 1 or 2)
21) Total Number of Significant Interactions at region(1) - how many other interactions use the same region endpoint
22) Total Number of Significant Interactions at region(2)
23) Annotation(1) - basic annotation from **annotatePeaks.pl**
24) Detailed Annotation(1) - detailed annotation from **annotatePeaks.pl**
25) Distance to TSS(1)
26) Nearest PromoterID(1)

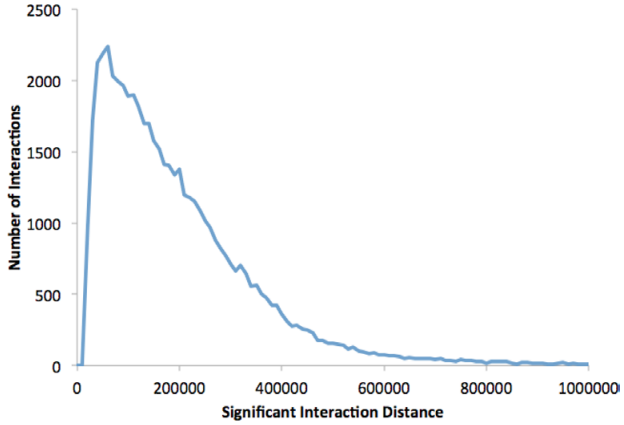
27) Gene Name(1)
28) Gene Alias(1)
29) Gene Description(1)
30) Annotation(2)
31) Detailed Annotation(2)
32) Distance to TSS(2)
33) Nearest PromoterID(2)
34) Gene Name(2)
35) Gene Alias(2)
36) Gene Description(2)

The "Total Number of Significant Interactions at region" described how many other interactions originate from the same region/endpoint. The "Annotation" and "Detailed Annotation" correspond to the basic and full annotations from annotatePeaks.pl.

NOTE: If you don't care too much about the annotation part, you can put "none" as the genome, and this part will be skipped (fields will be replaced with "NA").

lengthDist.txt

This file contains a histogram showing the distribution of interaction lengths. Graphing the file in Excel will produce something like the following:

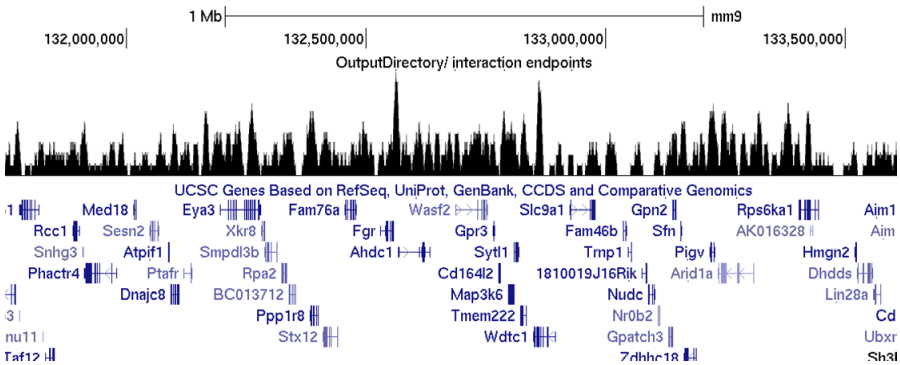


peaks.txt

Peak file containing all of the unique endpoint positions. The 7th column indicates how many interactions connect to that peak position.

endpoint.bedGraph

This file produces a coverage track of interaction endpoints. If you load this up to UCSC, it will look something like this:



endpoint.bedGraph.peaks

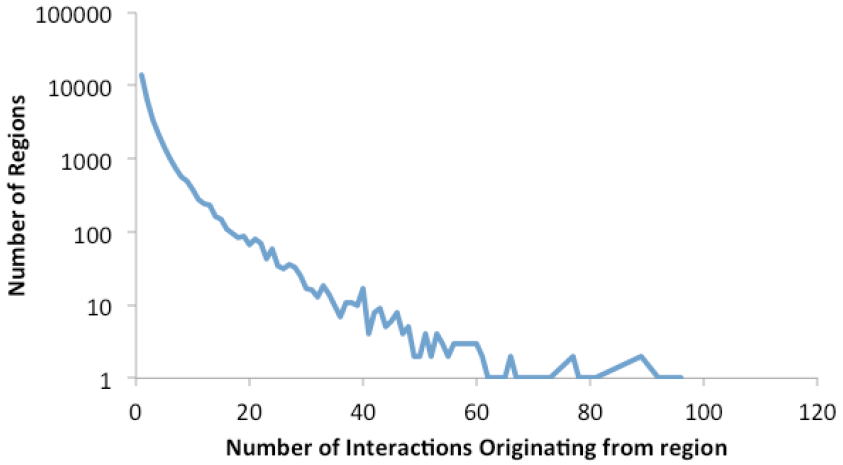
This peak file is an exploratory file that attempts to identify hubs directly from the peaks in the endpoint.bedgraph file that exceed the "**-hubCount <#>**" threshold (default: 5)

hubs.gt#.interactions.txt

This file is a peak file that contains regions that have more than 5 interactions connecting to that region. This allows you to focus your attention on highly connected regions if you want. To change the number of interactions required to designate a hub, use the "**-hubCount <#>**" option.

hubs.distribution.txt

This file contains a histogram describing the distribution of interactions per unique region. Graphing it in Excel looks something like:



Filtering Options

As mentioned above, one of the main purposes for the filtering options is so that liberal cutoffs for the p-value and z-score can be selected when running the **analyzeHiC** or **findHiCInteractionsByChr.pl** command to find the initial set of interactions. Once this large list of possible interactions is found, the p-value and z-score cutoffs can be changed/optimized in the **annotateInteractions.pl** command to avoid rerunning the other commands, which can be very time consuming.

Interaction Confidence:
-pvalue <#> : filter out interactions with p-value greater than #
-zscore <#> : filter out interactions with z-score less than #

Interaction Length:
-**minDist <#>** : filter out interactions spaced less than # bp - set > 300 million for only inter-chromosomal interactions
-maxDist <#> : filter out interactions spaced more than # bp, will remove inter-chromosomal interactions if set

Interaction Confidence Vs. Background:
-dpvalue <#> : filter out interaction with p-value vs. background Hi-C experiment greater than #
-dzscore <#> : filter out interaction with z-score vs. background Hi-C experiment less than #

Filtering Regions:
-filter <peakfile> : only look at interactions with endpoints in overlapping with peak in the file
-filter2 <peakfile> : only look at interactions connecting peaks in "-filter" file to the "-

filter2" file

Feature Enrichment at Interaction Endpoints

HOMER can compute feature enrichment calculations with your interactions. Simply add the "**-p <peak/BED file1> [peak/BED file2]...**" to the **annotateInteractions.pl** command. You can add as many peaks as you want to be considered. Below is an example:

annotateInteractions.pl bcell-Interactions.txt mm9 AnnotationOutputDirectory/ -p tss.peaks.txt ctcf.peaks.txt pu1.peaks.txt

Adding the "**-p ...**" option will initiate three changes to the output of the program:

1. Creation of a "**featureEnrichment.txt**" file in the output directory. This uses **mergePeaks** to assess the significance of overlap between the interaction endpoints and each of the peak files. When opened in Excel, the output looks like this:

	A	B	C	D	E
1	Feature Name	Number of overlapping Features	Enrichment Ratio	Enrichment (ln P-value,	file
2	tss	10464	1.163766	-2235.598844	tss.peaks.txt
3	ctcf	23980	1.610998	-8875.031153	ctcf.peaks.txt
4	pu1	21499	1.426652	-6858.604532	pu1.peaks.txt

2. Creation of a "**pairwiseFeatureEnrichment.txt**" file in the output directory. The program annotate the interaction endpoints of each significant interaction and check if any of the feature peak files overlaps with the interaction endpoints. It will then quantify how often each feature is "connected" to each other feature by an interaction. annotateInteractions.pl will then assess if a connection between the features is over- or under-represented given the general enrichment for each feature in the data set. The output file looks like this (+ logp enrichment indicate under-represented connections):

	A	B	C	D	E	F	G	H	I	J	K	L	M	N
1	FeatureSetID	Feature1	Feature2	Interactions(31918)	Expected	Ratio	log P-value	P-value	Total Feature	Total Feature	Feature Over	Feature 1	Feature 2	Total Interactions
2	0x0	tss	tss	1249	857.629676	0.37592612	-83.973697	3.39E-37	10464	10464	10464	-2235.6	-2235.6	31918
3	0x1	tss	ctcf	4029	3715.0819	0.08111748	-17.262047	3.19E-08	10464	23980	5248	-2235.6	-8875.03	31918
4	0x2	tss	pu1	3682	3310.69202	0.10629884	-25.30333	1.03E-11	10464	21499	5220	-2235.6	-6858.6	31918
5	1x1	ctcf	ctcf	5539	4504.04474	0.20683815	-134.11968	5.66E-59	23980	23980	23980	-8875.03	-8875.03	31918
6	1x2	ctcf	pu1	7679	7341.86757	0.04489608	-12.381713	4.19E-06	23980	21499	9682	-8875.03	-6858.6	31918
7	2x2	pu1	pu1	3909	3620.26913	0.07673322	-15.236237	2.42E-07	21499	21499	21499	-6858.6	-6858.6	31918

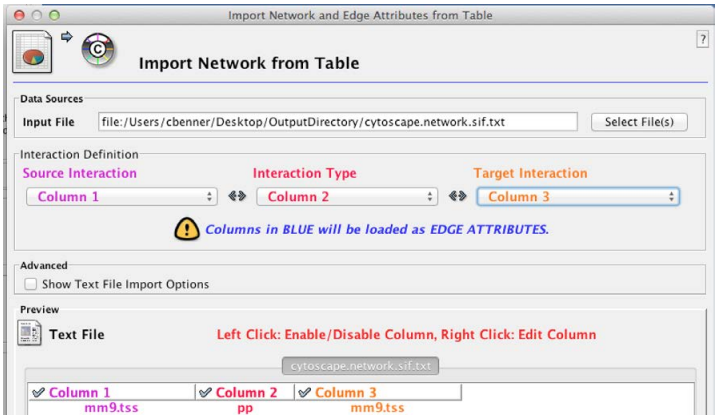
3. The **interactionAnnotation.txt** file will have one new column at the far right that will give the interaction codes that match the first column of the pairwiseFeatureEnrichment.txt file. In brief, each file is peak file is given an ID number (starting with 0, 1, 2...). Interactions that link peak features are assigned with the code (#x#, e.g. 0x0, 0x2, 1x2, etc.). For example, if an interaction is assigned the code 0x2 from above, then the interaction connects a TSS and PU.1 peak.

Visualizing Feature Enrichment with Cytoscape

It's hard to appreciate the pairwise feature enrichment in an excel table. The good news is that there is a great visualization tool called Cytoscape that can help out with that. HOMER's support for Cytoscape is a little clumsy (feedback on a more efficient way to setup the input files would be appreciated!).

When HOMER calculates feature enrichment, it will produce six files in the output directory staring with the name "cytoscape". These files are tab-delimited text files formatted to be used with Cytoscape. To load the network, follow these instructions:

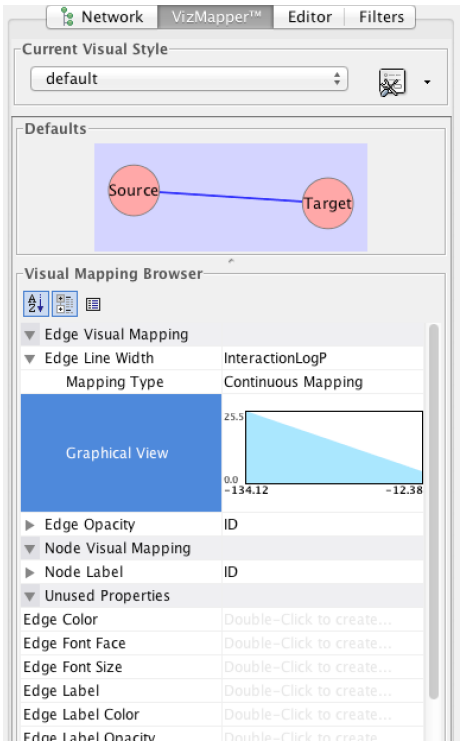
1. Go to File -> Import -> Network From Table (Text/MS Excel) For the file, select the "**cytoscape.network.sif.txt**" file. In the "Interaction Definition" box, set the Source Interaction to column 1, the Interaction Type to column 2, and the Target Interaction to column 3:



2. Next go to File -> Import -> Node Attributes... and load the "**cytoscape.node.logp.txt**" file. Do the same for the other cytoscape.node.* files (**cytoscape.node.size.txt**, **cytoscape.node.ratio.txt**). These files correspond to the values for general feature enrichment (i.e. the featureEnrichment.txt file [size is the total number of regions overlapping with each set of peaks)

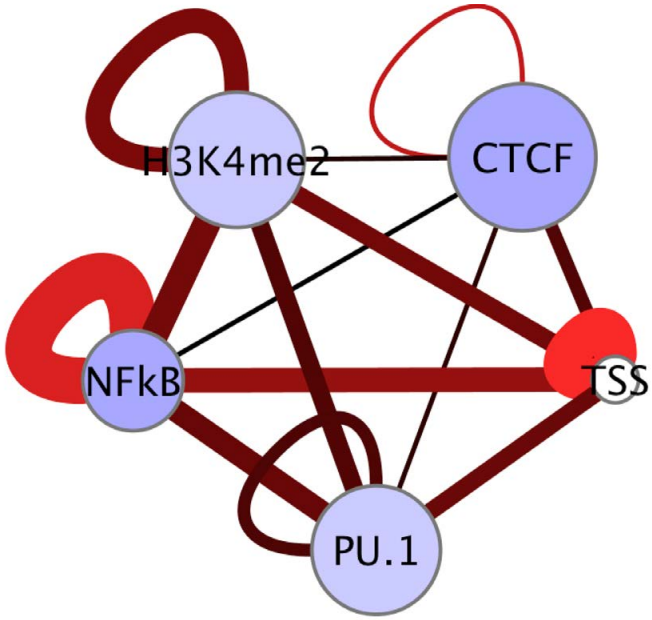
3. Then go to File -> Import -> Edge Attributes... and load the "**cytoscape.edge.logp.txt**" file. Do the same for the other cytoscape.edge.* files (**cytoscape.edge.ratio.txt**). These values correspond to the pairwise feature enrichment values.

4. Clicking on the network diagram should now reveal attributes in the "Data Panel" at the bottom of the screen. Next, to make the network pretty, click on the "VisMapper" on the left side of the screen. From here you can customize how your network displays the data. For example, click on the "Edge Line Width" and double click to activate it.



You can choose which attribute you want to visualize and select the appropriate parameters.

5. Takes awhile the first time to play with and make it look right... Good luck!



Modifying the Background for Feature Enrichment

By default, annotateInteractions.pl assumes your interactions were found by searching the entire genome. If you are analyzing a subset, you need to specify what was used with one of the following options:

-pos <chrN:XXX-YYY> : specific a specific region used for analysis
-gsize <#> : set the genome size used for significance calculations
-bgp <peak/BED file> : peaks used to find interactions from.

Miscellaneous/Specialized Analysis

Compare Interactions

If you have two interaction files, add "**-i <HOMER interaction file2>**" to the end of your **annotateInteractions.pl** command, and the 2nd interaction file will be compared to the first one, and only the common interactions will be analyzed. Common interactions are defined as interactions where the endpoints at each end are overlapping with each other (with the resolution size).

Connecting Features with Interactions

Lets say you have two peak files, maybe one called "enhancers.txt" and the other "promoters.txt", and you want to see which of the peaks in one of the files is connected to peaks in the other by significant interactions. If you add the "**-connect <peak/BED file1> <peak/BED file2>**" to the command, a "mapping" file will be sent to *stdout*. For example:

annotateInteractions.pl bcell-Interactions.txt mm9 AnnotationOutputDirectory/ -connect TSS.txt Enhancers.txt > outputMap.txt

The outputMap.txt file will contain 3 columns, the peakID from the first peak file (column1), the peakID from the 2nd file (column2), and the distance between them (3rd column). There could be many mappings for each peak, so each peak may appear multiple times in the file.

Manually Specifying Circos Edge Widths

The next sections discusses how to visualize interactions using Circos. However, you might find that it's difficult to set the width of the edges yourself. If you modify a HOMER interaction file manually and put the desired width of the edges in the file, you can run **annotateInteractions.pl** with the "**-circos**" option. This will output a Circos formatted interaction file. More on this in the next section...

Command Line options for annotateInteractions.pl

Usage: annotateInteractions.pl <interaction file> <genome version> <output directory>
[additional options]

General Options:
-res <#> (Resolution of analysis, default: auto detect)
-hubCount <#> (Minimum number of interactions to define a hub, default: 5)

Filtering Options:
-minDist <#> (filter out interactions spaced less than # bp - set high for only interchr)
-maxDist <#> (filter out interactions spaced more than # bp, will remove interchr)
-pvalue <#> (filter out interactions with p-value greater than #)
-dpvalue <#> (filter out interactions with p-value (vs bg) greater than #)
-zscore <#> (filter out interactions with zscore less than #)
-dzscore <#> (filter out interactions with zscore (vs bg) less than #)
-filter <peakfile> (only look at interactions with endpoints in peakfile)
-filter2 <peakfile2> (only look at interactions connecting -filter and -filter2 peak files)

Enrichment Options:
-p <peak file 1> [peak file 2] ... (Check overlap with peak files)

Special Operations:
-circos (Convert interactions to circos interactions format - stdout)
-i <interaction file2> [interaction file3] ... (Compare 1st file interactions to these)
-connect <peakFile1> <peakFile2> (returns association table between sets of peaks)
-pout (Convert interactions to a non-redundant peak file, sent to stdout)

Specifying Background (i.e. regions used to find interactions - default: whole genome)
-gsize <#> (size of genome, default: 2e9)
-pos chrN:XXX-YYY (specific, continuous region)
-bgp <peak file> (peak 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

---

Visualizing Hi-C Interactions with HOMER and Circos

Circos is a great program for visualizing interactions and integrating other data sources. To streamline the process of creating circos diagrams from Hi-C data and combining it with other types of sequencing, HOMER integrates several routines for preparing Circos input files. These configuration/data files are part of the output, allowing the user to modify/tweak them to produce the perfect image in Circos. To get perfect looking pictures, you have to learn a thing or two about Circos, but HOMER will do it's best to give you a quality image without needing to become an expert with Circos.

Required if not done already: Install Circos.

Finding Significant Hi-C Interactions with HOMER

Circos is simply a visualization tool (It *does not* analyze your data). To use it with HOMER, you must first understand how HOMER defines significant interactions, and should have a general idea how the analyzeHiC program works (see here and here), otherwise some of what follows may not make much sense. You can also feed HOMER interactions to visualize that may have nothing to do with Hi-C (use the "-i <interactionFile>" option with a HOMER interaction formatted file).

Generating Circos Diagrams with HOMER

To generate a Circos diagram, add "**-circos <prefix>**" to you **analyzeHiC** command. For example:

analyzeHiC <HiC Tag Directory> -res <#> -pos chrN:X-Y -circos <prefix> -nomatrix

i.e. **analyzeHiC ES-HiC -res 50000 -pos chr1:20,000,000-50,000,000 -circos chr1Interactions -nomatrix**

The "**-nomatrix**" option is optional, but normally you don't care about the normal matrix produced by **analyzeHiC**. This command will produce several files. In this case, the <*prefix*> was "chr1Interactions", so all files will start with that:

chr1Interactions.circos.conf
chr1Interactions.circos.interactions.txt
chr1Interactions.circos.karyotype.txt
chr1Interactions.circos.png

chr1Interactions.circos.svg

(more files may be present with additional options...)

The final two of which are the actual output images. The PNG is nice for normal viewing, the SVG is better for importing into graphics applications like Illustrator.

Often, the circos diagram is not perfect - you may want to change the font, or the color. To do this, edit the "***prefix*.circos.conf**" file with the appropriate settings, and then rerun the circos image generation using:

circos -conf *prefix*.circos.conf

i.e. **circos -conf chr1Interactions.circos.conf**

This will produce new PNG and SVG files. Circos is all about creating your ***.circos.conf** file - the program has very few command line options that are regularly used. Circos is also a very feature rich program that could take you years to fully explore. To learn more about how to edit circos.conf files, check out the Circos tutorials.

Important Interaction Parameters

Often your interactions may not look that great. Could be you need to adjust these parameters to clean up the image:

-pvalue <#> : Will filter out interactions with a pvalue greater thatn # (default, 0.001). You may need to try different values to clean up the image

-res <#> / -superRes <#> : controls the resolution of the analysis

-minDist <#> : Useful to set this to exclude trivial interaction between adjacent regions, etc.

Adding Sequencing Data to Circos Output

analyzeHiC has three general options for automating the visualization different types of data:

-d <tag directory1> [tag directory2] ...

Add ChIP-Seq, RNA-Seq, or really any data from a HOMER tag directory. makeUCSCfile will be called to generate a bedGraph

-b <peak/BED file1> [peak/BED file2] ...

Add peak/BED files to identify were certain features are located

-g <named peak/BED file>

Similar to the "-b" option, but in this case the regions will be named. This is most useful for annotating gene positions in

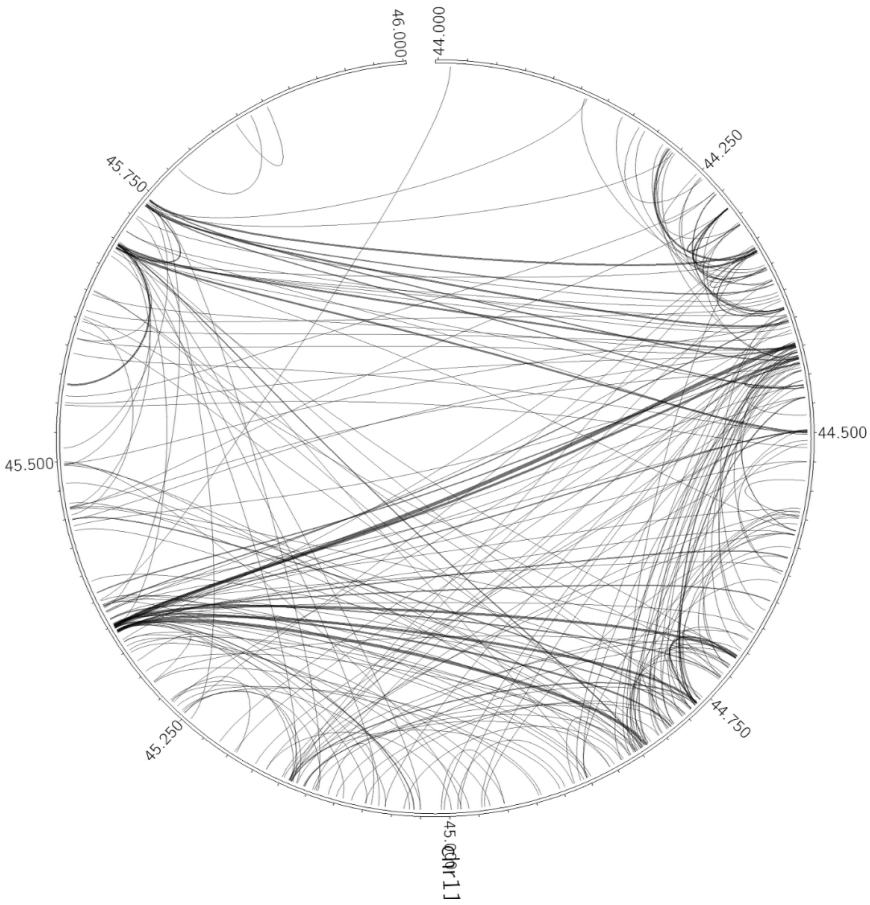
the file. Gene files you may want to use (you can also easily make your own):

RefSeq genes for mm9: mm9.genes.txt
RefSeq genes for hg18: hg18.genes.txt
RefSeq genes for hg19: hg19.genes.txt

Circos is a very powerful program, and there are lots of other types of things you can do with your data as well. These are only the ones covered by HOMER, but don't let that limit you if you want to use other types of data.

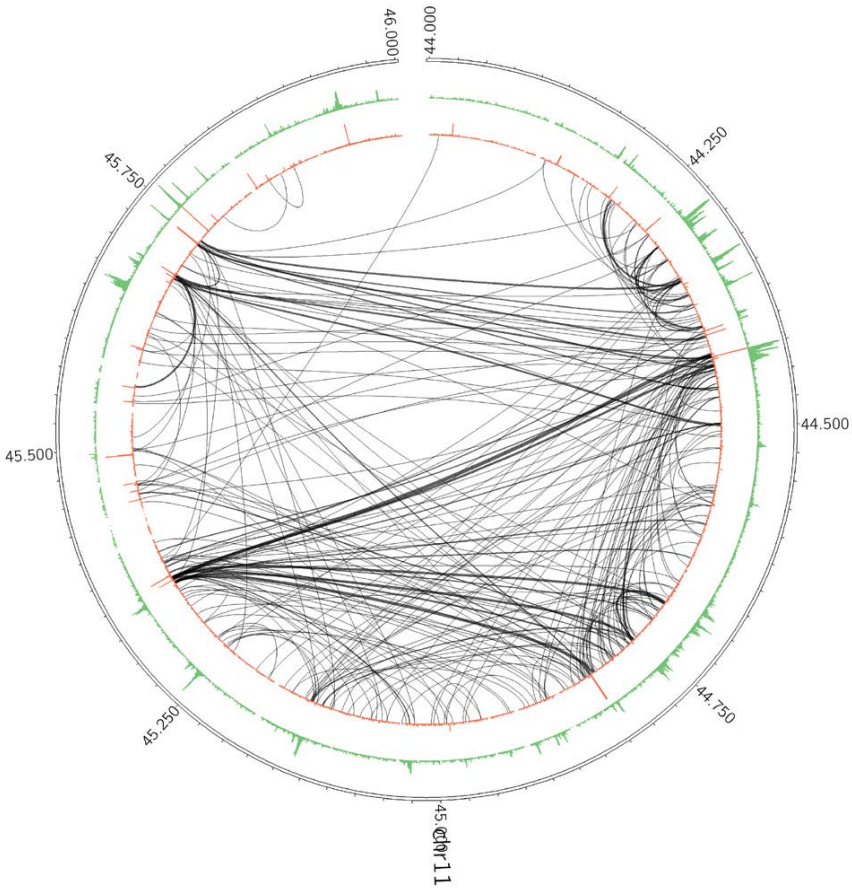
Below is a demonstration of how you can add additional data to your circos output. First, we will visualize the B cell interactions formed in along a region of chr11:

analyzeHiC proB-HiC -pos chr11:44,000,000-46,000,000 -res 2500 -superRes 10000 -circos cirOutput -nomatrix -minDist 20000



Next, lets add some ChIP-Seq data for CTCF (inslutor/boundary transcription factor) and H3K4me2 (epigenetic histone modification found at promoters and enhancers):

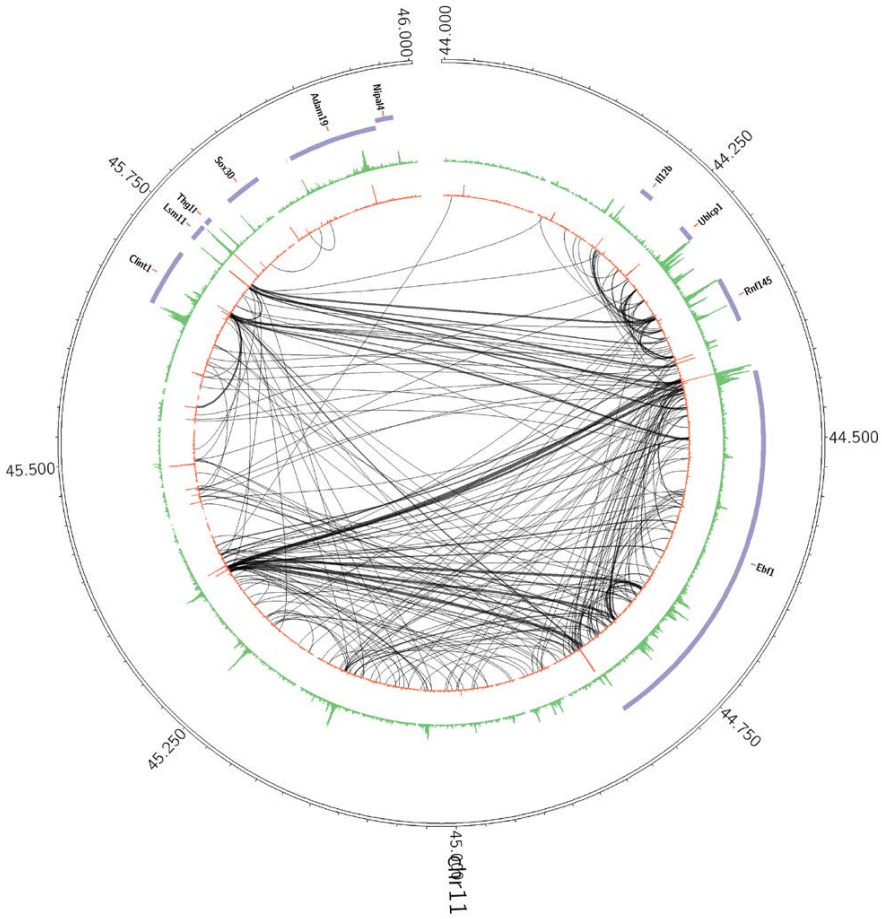
analyzeHiC proB-HiC -pos chr11:44,000,000-46,000,000 -res 2500 -superRes 10000 -circos cirOutput -nomatrix -minDist 20000 *-d CTCF-chipseq/ H3K4me2-chipseq/*



In this case the interactions didn't change, but the layout was adjusted to make room for ChIP-Seq data.

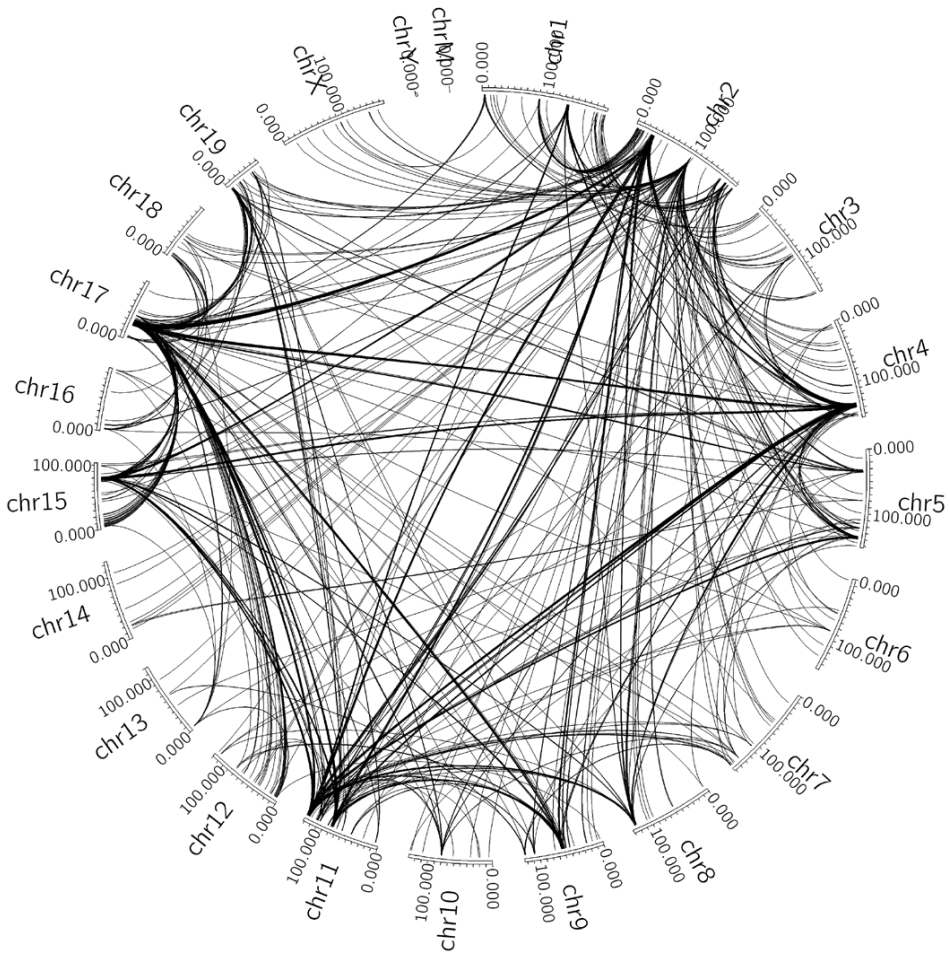
Now lets show where the genes are - do with by adding the "-g" option.

analyzeHiC proB-HiC -pos chr11:44,000,000-46,000,000 -res 2500 -superRes 10000 -circos cirOutput -nomatrix -minDist 20000 -d CTCF-chipseq/ H3K4me2-chipseq/ -g mm9.genes.txt



In the end, there are lots of ways to generate circos figures. For example, lets say you want to visualize interchromosomal interactions? Consider the following (the -minDist was set very high such that intrachromosomal interactions will be ignored):

analyzeHiC proB-HiC -res 1000000 -cpu 8 -pvalue 1e-7 -circos interChrom -minDist 2000000000 -nomatrix



Modifying Circos output without *rerunning* analyzeHiC

Often you'll run **analyzeHiC** with the "**-circos <prefix>**" option and get a giant mess of interactions, especially if the "**-threshold <#>**" is set low. Sometimes things are not clear, or the ChIP-Seq signal is too low low to see. For these types of problems, it is necessary to directly modify the input files for Circos instead of rerunning analyzeHiC with different parameters. Below is a set of modifications that may be helpful. There are a ton of other things you can do with Circos, so it might be worth learning more about it from the source.

To regenerate the Circos output after making modifications, simply run:

circos -conf <prefix>.circos.conf

If you ran: **analyzeHiC ES-HiC -res 50000 -chr chr1 -circos chr1Interactions > output.txt**
Then run: **circos -conf chr1Interactions.circos.conf**

Visualizing Subsets of Interactions

Lets say you want to see only interactions that are interchromosomal, or only the interactions that are greater than 50Mb apart. You can control this by

adding "Rules" to the Circos configuration file. If you open the <prefix>.circos.conf file in a text editor, you should see something like this:

```
...
   <rules>
      <rule>
         importance = 200
         condition = _THICKNESS1_ > 20
         thickness = 20
      </rule>
...
```

The idea is to add additional rules to the file such as:

```
...
      <rules>
         <rule>
            importance = 200
            condition = _THICKNESS1_ > 20
            thickness = 20
         </rule>
         <rule>
            importance = 250
            condition = _INTRACHR_ && abs(_POSITION1_ -
_POSITION2_) < 50Mb
            show = no
         </rule>
...
```

This rule with check the condition - is the interaction intra chromosomal and are the interacting positions less then 50Mb - then Circos will modify the way the interaction is shown. In this case, it says "show = no", which will hide the interaction. Make sure the "importance" is higher than the other importance numbers - this specifies which rules take priority - 250 is fine. Lets say we only want to show only interchromosomal interactions. We'd add the rule:

```
         <rule>
            importance = 250
            condition = _INTRACHR_
            show = no
         </rule>
```

I this case, if the interaction is intrachromosomal, then it won't show it. If you'd prefer to color those interactions a different color instead of hiding them, try "color = red" instead of "show = no".

For more on rules, check out the Circos tutorial.

Changing the Range on tag directory (i.e. ChIP-Seq) tag pileups

By default, analyzeHiC will display tag densities ranging from 0 to 100 normalized tag counts. For some data this is the wrong range. To change this manually, edit the <prefix>.circos.conf file and change the **min** and **max** parameters (you can also change the color and other things):

```
...<plots>
```

```
    <plot>
      show = yes
      z=5
      type = histogram
      r0 = 0.90r
      r1 = 1.00r
      color = red
      fill_color = red
      fill_under = yes
      thickness = 1
      extend_bin = no
      background = no
      axis = no
      file = yyy.circos.histogram1.txt
      min = 0
      max = 100
    </plot>
    ...
  </plots>
  ...
```

More info from Circos

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

HOMER

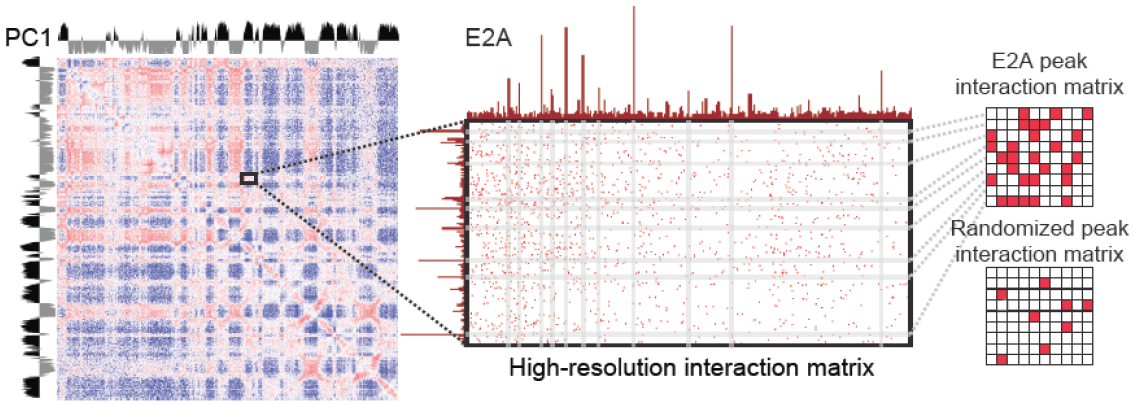
Software for motif discovery and next-gen sequencing analysis

---

SIMA: Structured Interaction Matrix Analysis

Hi-C is an unbiased assay of chromatin conformation, resulting in even read coverage across the entire genome. This, coupled with the fact that most Hi-C reads describe interactions at close linear distance along the chromosomes, results in relatively sparse read coverage for interactions between individual restriction fragments separated by great distance. This makes it difficult to find high-resolution interactions across long distances. To help identify which distal and/or inter-chromosomal regions co-localize, we generally increase the size of the regions [i.e. decrease the resolution] to boost the number of Hi-C reads used for interaction analysis. This is the most common and simplest technique used by most approaches (including HOMER). It's much easier to identify significant inter-chromosomal interactions between regions that are 100kb in size than regions that are only 5kb in size (essentially 20x more Hi-C reads to work with).

The major drawback to this methodology is that it is difficult to identify features that are responsible for mediating the interaction due to the large resolution size. To address this, HOMER implements an algorithm called Structured Interaction Matrix Analysis (or SIMA) to help identify which genomic features play a role in mediating interactions between two large domains. SIMA works by pooling interactions across a set of genomic features, such as ChIP-Seq peaks, and scores the relative enrichment of Hi-C interactions in these regions relative to randomly selected regions. The schematic below describes how this works:



SIMA takes a set of "domains" as input (could be only 1), and peak/BED files for assessing enrichment. SIMA then analyzes each domain against each of the other domains and calculates the number of interactions connecting peaks from on domain to peaks in the other domain (normalized for expected interactions based on the background model). It then randomizes the peak positions and recalculates the connecting interactions to generate a NULL distribution to use to gauge the significance of the interactions from the true peaks.

Generating Domains to use with SIMA

One common way to generate domains is to use compartment regions identified from PCA analysis. For example:

runHiCpca.pl esData ES-HiC/ -cpu 8
findHiCCompartments.pl esData.PC1.txt > activeDomains.txt

You could also use topological domains, custom regions, etc. The input domain file for SIMA is a peak/BED file. If you only want to analyze a single region, just include a single region in the domain file.

Running SIMA

SIMA is generally run using a two step process. First, the calculations are made generating a table of results. Then this table can be used to generate visualization files. To run SIMA, use the **SIMA.pl** program as shown below:

SIMA.pl <Hi-C Tag Directory> -d <domain peak/BED file> -p <peak/BED file1> ... [other options] > outputTable.txt

For Example:
SIMA.pl Bcell-HiC/ -d domains.txt -p pu1.peaks.txt -cpu 8 -chr chr19 > simaOutput.txt

By default, this will compare each domain to every other domain. For set of domain comparisons, the peak files supplied will be used to assess their significance as interaction anchors between domains.

SIMA.pl can take a while to run since it's performing high resolution analysis across the interaction space. If you have multiple CPUs, it's highly recommended to run **SIMA.pl** with the "**-cpu <#>**" option.

Below are important options to help control how SIMA is run:

-d <domain peak/BED file> : regions to use as domains

-minDsize <#> : removes domains smaller than this size (default: 500000)

-p <peak/BED file> [peak/BED file2] ... : regions to score interactions at. You can supply multiple peak files. Each peak file will be used separately to score interactions (file1 vs file1, file2 vs file2, etc.)

-AvsA : compare each of the peaks against one another.

-p2 <peak/BED file> [peak/BED file2] ... : Peaks must be "paired" with peaks specified with "-p <peak/BED> ...". This allows different types of peaks to be compared (file1[-p] vs. file1[-p2], file2[-p] vs. file2[-p2], etc.). If in doubt, use the "-AvsA" option.

-res <#> | -superRes <#> : Sets resolution of analysis (default: -res 2000 -superRes 10000)

-minDist <#> : Do not consider Hi-C interactions spanning less than this distance. Important for keeping analysis away from the diagonal of interaction matrix (default: 2x super resolution).

-min <#> : minimum distance between domains to analyze (default: -1).

-max <#> : maximum distance between domains to analyze (default: 1e9). To only analyze domains vs. themselves, set this value to something very low. To analyze all possible domains (including interchromosomeal interactions), set to -1.

-chr <chr> : only analyze domains found on this chromsome

-N <#> : Number of randomization for calculating significance scores.

-cpu < #> : Number of CPUs to use (Domain comparisons will be done on separate CPUs)

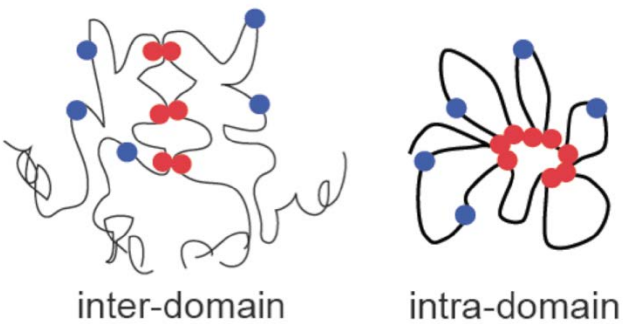
SIMA will send the output table to stdout. Below is a description of the output columns:

1. region name for domain1
2. chr for domain1
3. start position for domain1
4. end position for domain1
5. region name for domain2
6. chr for domain2
7. start position for domain2
8. end position for domain2
9. peak/BED file1 (for domain1)
10. peak/BED file2 (for domain2
11. Number of peaks in domain1
12. Number of peaks in domain2
13. p-value of comparison
14. Ratio of observed Hi-C Read enrichment vs. Randomized Average (e.g. col 15 vs. col 16)
15. Peak enrichment ratio vs. expected interactions
16. Average randomized peak enrichment ratio vs. expected interactions

Interpreting SIMA results

When comparing one domain versus another, a strong positive ratio [col 14] (and low p-value) implies that the peaks used in the analysis participate in a higher proportion of the inter-domain interactions then expected by chance (red below). If inter-domain interactions are depleted at the peak positions, the ratio will be well below 1.

When comparing a domain versus itself, a strong positive ratio implies that the peaks participate in a high proportion of the intra-domain interactions. Below is a schematic to illustrate the physical interpretation.



Visualizing SIMA Results

After running SIMA to generate the output table, you can run SIMA.pl again to generate visualization files. Generating visualization files is quick - the idea is to run the analysis once (which takes much longer), and then you can experiment with different visualization options.

Generally, there are two ways to visualize SIMA results. To visualize how SIMA scores vary across domains, the Matrix/Heapmap option can be used. To visualize how multiple peaks form interactions across a single domain (or set of domains), then the Cytoscape option should be used.

Domain Comparison Matrix/Heatmap

SIMA will visualize the domain interaction scores using the following command:

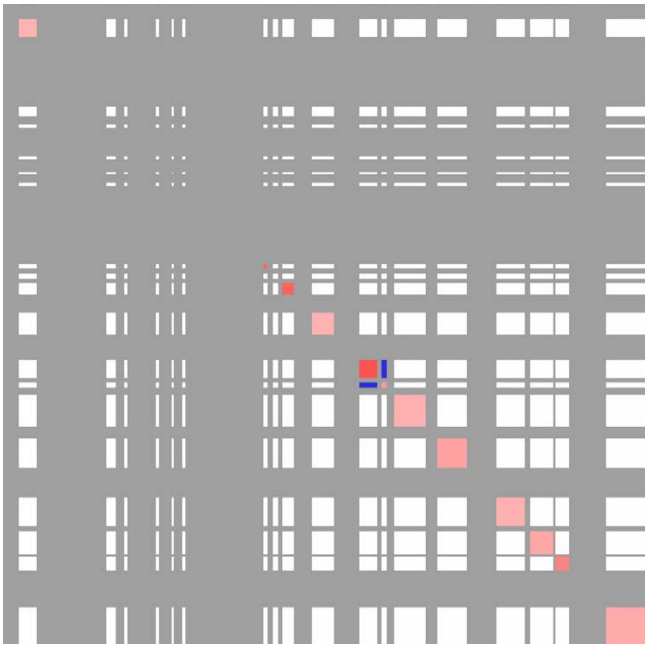
SIMA.pl -matrix <SIMA output from analysis> [options] > outputMatrix.txt

In practice, you would run SIMA like this (using PC1 compartments saved in the file "domains"):

#run SIMA first
SIMA.pl Bcell-HiC -d domains.txt -p ctcf.peaks.txt -cpu 8 -chr chr11 > simaOutputTable.txt

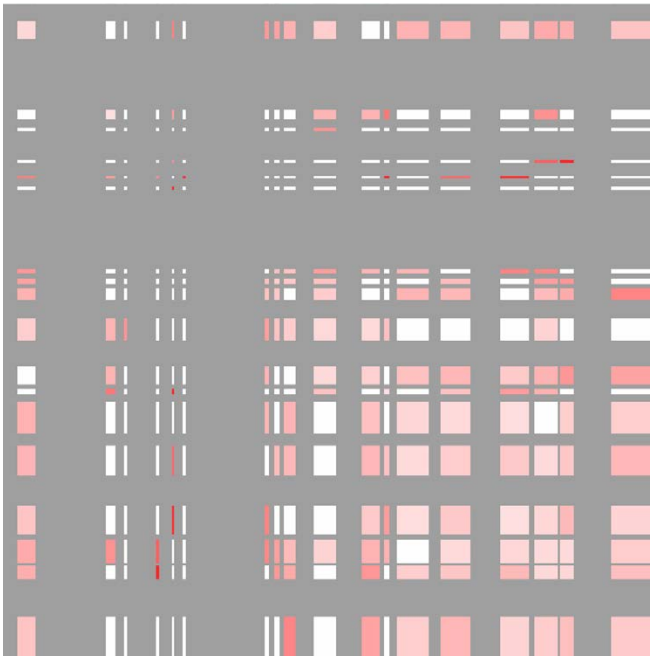
#create heatmap
SIMA.pl -matrix simaOutputTable.txt -p ctcf.peaks.txt -minPeaks 5 > outputMatrix.txt

Viewing the "outputMatrix.txt" with Java Tree View (visualizing values as log2):



Each row and column is an active compartment along chr11 (these regions correspond to the PC1 values in the SIMA figure at the beginning of this section)

If we use TSS for the SIMA analysis instead of CTCF peaks, we get something that looks more like this:



There are several option to tune how the result is shown:

-res <#> : Resolution of output matrix file (default: 200000). You'll need to adjust this if your domain sizes are small.

-p <peak/BED file> : display results for this peak file (if multiple peaks were analyzed at once)

-minPeaks <#> : only display regions with at least this many peaks.

-stat <pvalue|ratio> : display this value in the heatmap (default: ratio)

-pvalue <#> : only display ratio of domain relationships if the p-value is less than this value (default: 0.01)

Relationship between peaks in Cytoscape

To understand the relationship between peak files and Hi-C interactions across a single domain/region, SIMA has an option to output files that can be visualized using Cytoscape. Normally to run SIMA in this mode, you would collect several peak files and use the "-AvsA" option in the command like this:

```
# save a single region into the file domain.txt
SIMA.pl Bcell-HiC -p ctcf.peaks pu1.peaks h3k4me2.peaks
e2a.peaks -d domain.txt -AvsA > outputTable.txt
```

To generate the visualization files, run **SIMA.pl** with the "**-cytoscape**"

option:

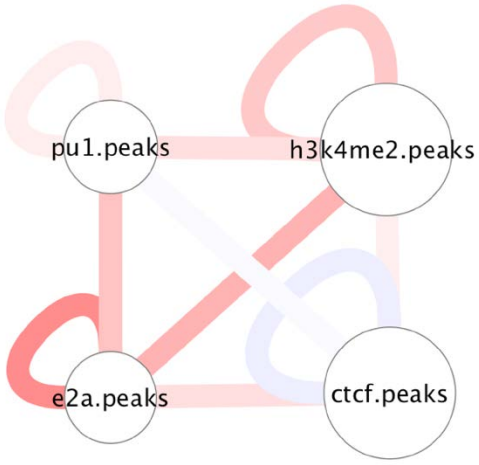
SIMA.pl -cytoscape <SIMA output from analysis> [options]

For example:
SIMA.pl -cytoscape outputTable.txt

Running SIMA.pl with the "-cytoscape" option will produce several output files:

cytoscape.InputFilename.network.txt
cytoscape.InputFilename.nodes.size.txt
cytoscape.InputFilename.edges.ratio.txt
cytoscape.InputFilename.edges.pvalue.txt

These files can be loaded into Cytoscape in a similar manner to that described for global interactions (see here). For example, visualizing the example above should look something like this:



If SIMA was original run with multiple domains, you can create cytoscape files specific to the domains of interest by using the "-dname <domain name1>" and "-dname2 <domain name2>" options.

Command Line options for SIMA.pl

Normal usage: SIMA.pl <HIC directory> [options]

Output table is sent to stdout
(See below for output visualization formatting)

Required Options:
-d <domain peak file> (Domains to perform analysis on)
-p <peak file1> [peak file2] ... (features to check for enrichment)

Options:
-res <#> (resolution, default=2500)
-superRes <#> (super resolution/window size, default=10000)
-minDist <#> (minimum interaction read distance, default: 2x superRes)
-minDsize <#> (minimum domain size, default: 500000)
-min <#> (minimum distance between domains to test significance, default=-1)

-max <#> (maximum distance, set to -1 to allow inter-chr, default=1e9)
-chr <chromosome> (only analyze this chromosome, default: all)
-p2 <peak file1> [peak file2] ... (heterogenous peak enrichments)
-AvsA (All versus All, compare -p peaks against one another)
-N <#> (Number of randomizations per domain, default: 1000)
-cpu <#> (number of CPUs to use for analysis, default: 1)

Output Visualization Formatting (Run SIMA first, then format the output)

Matrix Mode: Takes output and prints out a matrix for visualization
SIMA.pl -matrix <SIMA output from analysis> [options]

Matrix Mode Options: (defaults to resolution of 200000, output to stdout)
-stat <pvalue|ratio> (output stat for matrix mode, default: ratio)
-pvalue <#> (p-value cutoff to report the ratio, default: 0.01)
-minPeaks <#> (minimum number of peaks, default: none)
-res <#> (resolution of matrix, default=200000)
-p <peak file1> (features from initial analysis to show)
-p2 <peak file2> (features from initial analysis to show, if used/different)

Cytoscape Mode: Takes output from single domain and prints files
SIMA.pl -cytoscape <SIMA output from analysis>
(output to "cytoscape.filename.*" files)
-dname <name> (domain name to show)
-dname2 <name2> (domain name to show, if different)

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

HOMER

Software for motif discovery and next-gen sequencing analysis

---

Hi-C Analysis Tips

Hi-C is a beast of an assay. There is a lot of information in each run, and the size and complexity of the experiments can stretch the limits of your computer too. Below are some tips and "best practices" to help get Hi-C analysis with HOMER to turn out successful.

Resolution and Contact Matrices

Keep in mind that the size of the region you are looking divided by the resolution will yield the total size of your matrix. For example, mouse chr1 is about 200 Mb long. If you make matrix with chr1 at 100kb resolution, you matrix will be 2000x2000 in size. Even if one pixel is used to visualize each data point in the matrix, it's not going to fit on your computer screen. It will also be a very large file (tens of megabytes). If you try to visualize chr1 at 10kb resolution, your matrix will be 20000x20000, which in all likely hood will eat up all of your memory and disk. Homer has a checkpoint to warn you about this, but in general, if you want to visualize something at high resolution, use a smaller region.

Resolution and Running Time

The smaller the size of the regions used for analysis, the longer everything is going to take. In general, the analysis time scales $O(n^2)$ with the number of regions analyzed, so analysis with a resolution of 10000 will take approximately 4x longer than analysis with a resolution of 20000.

In practice it's much better to start with a low resolution (i.e. 100kb), and if everything works out, try a higher resolution (i.e. 10kb).

How small can the Resolution/Super Resolution Go?

Is it possible to analyze my data at 5kb resolution? What about 1kb or 250 bp resolution?

Number of CPUs and Memory

Several of the HOMER Hi-C programs support multiple processors to help speed up the computation. Unfortunately, as of now this works at the chromosome level. This means that if you use "**-cpu 3**" with the **analyzeHiC** command, it will spawn 3 threads to handle chr1, chr2, and chr3 at the same time. As each chromosome finishes, it will start on chr4 next, and so on. Ideally the parallel code would be deeper in the analysis (i.e. analyze chr1 at 3x speed), but I haven't had time to go back in and re-butcher everything.

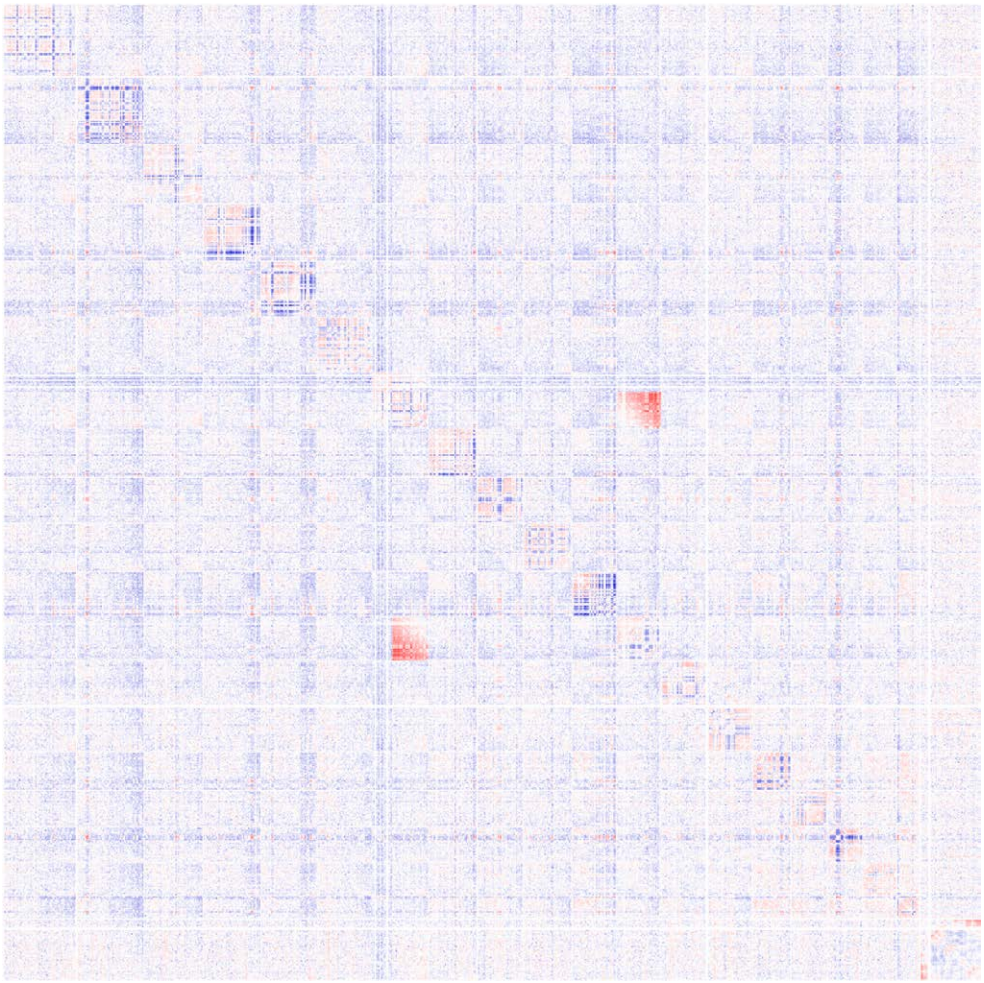
On important practical consideration: The total data for each chromosome is read into memory! If your experiment is very large, this may cause problems. For example, lets say you have 16 CPUs on you computer, so you run analyzeHiC with "-cpu 16". The data for each of the first 16 chromosomes will then be read into memory... If the experiment is very large (say a billion reads), you may run out of memory, particularly if you only have 50-100Gb). In these cases, you may be forced to use less CPUs...

Impact of Genomic Rearrangements and Copy Number Variants on Analysis

Genomic rearrangements in particular can wreak havoc on Hi-C analysis. Ideally, Hi-C reads should be mapped to the actual genome of the cells being analyzed. However, is option is not always available and difficult to perform in practice. How do you know if your cells have any major genome altercations? I'd recommend creating a genome-wide interaction matrix at 1Mb resolution. For exampe, consider the following:

analyzeHiC HiCexp1/ -res 1000000 -cpu 8 > outputMatrix.txt

The resulting matrix might look something like this:



Here you can see a large block of signal between chr7 and chr12. From this, its a good bet that the latter portion of chr7 is ligated to the end of chr12. You can't necessarily "prove" these cell have a translocation from Hi-C data, but it's likely that this is the case.

How does this effect your analysis? Ideally, the coordinates of the reads could be adjusted to reflect the real genome present in the cell. This is tricky (nearly impossible due to the potential heterozygosity of the rearrangements). The best (or easiest) course of action is to disregard significant interactions and results from these regions (i.e. remove inter-chromosomal interactions between chr7 and chr12 in this case). I will try to update this sections with more tricks and techniques as we encounter more data of this nature.

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