dacpet: *De novo* analysis of ChIA-PET data

User's Guide

Aaron Lun

First edition 10 July 2014

Last revised 18 November 2014

Contents

Chapter 1

Introduction

1.1 Scope

This document describes the use of the dacpet package for *de novo* identification of protein-mediated interactions from CHIA-PET data. In particular, specific interactions are distinguished from non-specific ligation events by comparing the homo-linker and hetero-linker counts. Analyses can also be performed to identify differential interactions between biological conditions. This is done in a statistically rigorous manner using the methods in the edgeR package [Robinson et al., 2010]. Knowledge of edgeR is useful, but not necessary.

1.2 How to get help

Most questions about individual functions should be answered by the documentation. For example, if you want to know more about `preparePET`, you can bring up the documentation by typing `?preparePET` or `help(preparePET)` at the R prompt. If that doesn't help, reading this guide or contacting one of the authors is probably the best approach. Bug reports and thoughtful suggestions for improvements are occasionally appreciated.

1.3 Quick start

The analysis of ChIA-PET data with dacpet involves a number of steps. For simplicity, assume that the BAM files have already been processed to tag files in `curfiles`. Also assume that self-ligation events have been removed. The code itself can then be written as:

1. converting BAM files to tag files

2. removing self-ligation events

3. counting tag pairs for each interaction

```
> require(dacpet)
> countwidth <- 5000
> data <-countPET(curfiles, width=countwidth, filter=10)
> freqs <- compressMatrix(data)
```

4. filtering out low-abundance interactions

```
> require(edgeR)
> ab <- aveLogCPM(asDGEList(freqs))
> freqs <- freqs[ab > -1,]
```

5. normalizing the linker combinations

```
> binned <-countPET(curfiles, width=1e7, filter=1)
> rebin <- compressMatrix(binned)
> inters <- rebin[is.na(getDistance(rebin)),]
> normfacs <- normalize(inters)
```

6. modelling biological variability

```
> y <- asDGEList(freqs, norm.factors=normfacs)
> design <- model.matrix(~factor(info(freqs)$hetero))
> y <- estimateDisp(y, design)
> fit <- glmQLFit(y, design, robust=TRUE)
```

7. testing for significant differences between homo-/hetero-linker counts

```
> result <- glmQLFTest(fit)
```

In the various examples for this guide, data will be used from a ChIA-PET experiment targeting RNA polymerase II in MCF7 cells [Li et al., 2012]. Obviously, readers will have to modify the code according to their own circumstances.

Chapter 2

Preparing tag files

2.1 Splitting tags and aligning reads

As its name suggests, the ChIA-PET procedure generates paired-end sequencing data. Each read in a pair has a tag and linker component as the 5′ and 3′ segment, respectively. The linker component for each read must be identified as either A or B, based on the known linker sequence. Then, it must be removed so that the tag can be aligned to the reference genome. Removal is achieved with the `splitLinkers` function, which takes two FastQ files of tag-linker pairs and generates several paired FastQ files. Each pair of files contains the tag pairs corresponding to each linker combination, i.e., AA, BB, AB or other/unknown.

```
> require(dacpet)
> splitLinkers(fastq1="SRR372741_1.fastq", fastq2="SRR372741_2.fastq",
+     linkerA="GTTGGATAAGATATCG", linkerB="GTTGGAATGTATATCG")
```

In older ChIA-PET datasets, reads were 36 bp long to accommodate a 20 bp tag and a 16 bp linker. The `splitLinkers` function will automatically assume that the last 16 bp of the read represents the linker. In newer experiments, longer read lengths are typically used. The read position at which the linker starts must then be specified manually with the `start` parameter. This should be set to 20 - 21 bp if the restriction enzyme used is *Mme*I.

The alignment itself can be performed with an appropriate aligner like Bowtie2 [Langmead and Salzberg, 2012]. Users should choose a mapping program that can handle 20 bp tags. This should also be done in a manner that does not enforce similar mapping locations for paired tags, as each pair represents an interaction between distinct loci. Duplicate marking with the MarkDuplicates tool from the Picard suite (`http://broadinstitute.github.io/picard`) is recommended. The ultimate aim is to produce a name-sorted BAM file for each linker combination in each library, i.e., 3 BAM files. For the MCF7 dataset, this results in 6 files in total after processing and mapping (ignoring those with unknown linkers):

```
> bamfiles <- c("SRR372741_AA.bam", "SRR372741_AB.bam", "SRR372741_BB.bam",
+         "SRR372742_AA.bam", "SRR372742_AB.bam", "SRR372742_BB.bam")
```

Note that the three BAM files named `SRR372741_*` correspond to a single ChIA-PET library, as do those named `SRR372742_*`. These two libraries represent biological replicates, according to the entry in the NCBI Gene Expression Omnibus (accession number GSE33664). They will be used later to assess the reproducibility of the identified interactions.

2.2 Converting BAM files to tag files

Repeated parsing of the BAM file is quite laborious for routine analyses. Instead, the BAM file is parsed once to generate a processed tag file in HDF5 format. Each tag file contains a number of `data.frame` objects, each of which contains all tag pairs mapped between a particular pair of chromosomes. Information can then be extracted efficiently from the tag file for downstream processing. This parsing is performed using the `preparePET` function to generate a tag file for each BAM file in the dataset.

```
> incoming <- bamfiles[1]
> outcoming <- sub("\\.bam", ".h5", incoming)
> out <- preparePET(incoming, outcoming, minq=20, dedup=TRUE)
```

The value of `minq` describes the minimum mapping quality score (MAPQ) for aligned tags. Both tags in each pair must have a MAPQ above `minq` for that pair to be retained in the tag file. A reasonably stringent value is recommended given that alignment of short tags is likely to be error-prone. Similarly, the `dedup` value specifies whether or not marked duplicates should be removed. This is recommended as the chance of randomly obtaining pairs with the same positions in the genome-by-genome space is low. Any overlaps are likely to be technical artifacts from PCR duplication during library preparation. The numbers of pairs with at least one poorly mapped or duplicate tag can be examined in the output:

```
> out$pairs

   total   marked filtered    valid
17734488   954225  9095921  7867801
```

In most cases, the duplicate tags are also those with low mapping qualities as the aligner tends to place unknown sequences within alignable regions. The total number of tag pairs is also shown, along with the number of pairs that are retained in the final tag file. You can also have a look at the number of single tags or those with more than two aligned segments. These should be non-existent, otherwise it suggests that the BAM file is malformed.

```
> out$other

singles   multi
      0       0
```

For brevity, this processing step is only performed here for one BAM file. Real analyses should run this function over a loop for all files (and store diagnostics) as required.

2.3 Diagnosing ligation quality

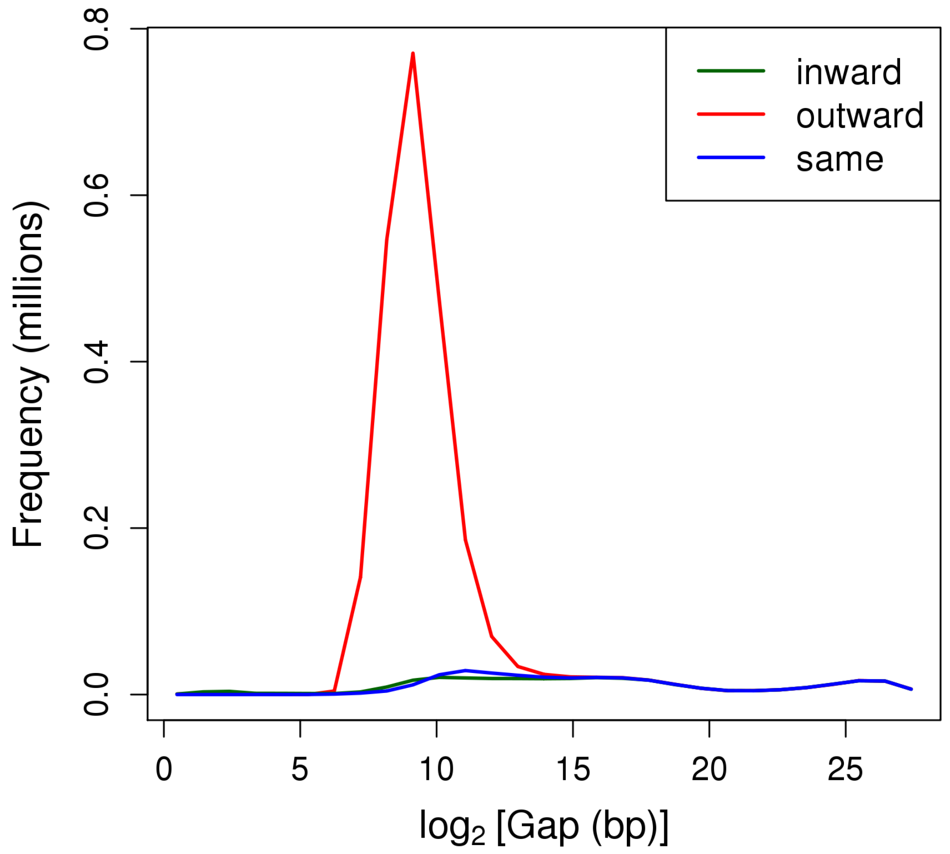
2.3.1 Making strand orientation plots

Some diagnostics can be pulled out of each processed tag file using the `diagnosePET` function. This extracts the strand information for inter-chromosomal tag pairs, i.e., those with tags mapped to different chromosomes. Each code refers to the combination of strands to which the tags in each pair are mapped, i.e., both forward (`0`), forward-reverse (`1`), reverse-forward (`2`) and both reverse (`3`). As you can see, these numbers are fairly balanced as ligation between different pieces of DNA can generate any combination of strands.

```
> diag <- diagnosePET(outcoming)
> diag$inter

      0       1       2       3 
1125899 1125172 1124961 1125080 
```

The function will also collect strand and gap information for intra-chromosomal tag pairs. In this case, the "first" tag in a pair is the one that maps at a lower genomic position on that chromosome. This means that a code of `1` marks an inward-facing tag pair (forward-reverse) whereas a code of `2` marks an outward-facing tag pair (reverse-forward). The gap distance refers to the distance between tag positions on the same chromosome. The distribution of gaps can then be examined for each strand orientation [Jin et al., 2013]. The average distribution of codes `0` and `3` is shown as both involve alignment to the same strand.

```
> lgap <- log2(diag$gap+1)
> breaks <- seq(min(lgap), max(lgap), length.out=30)
> iwin <- hist(lgap[diag$flag==1L], plot=FALSE, breaks=breaks)
> owin <- hist(lgap[diag$flag==2L], plot=FALSE, breaks=breaks)
> ssin <- hist(lgap[diag$flag==0L | diag$flag==3L], plot=FALSE, breaks=breaks)
> plot(owin$mids, owin$counts/1e6, type="l", xlab=expression(log[2]~"[Gap (bp)]"),
+     ylab="Frequency (millions)", col="red", lwd=2, cex.lab=1.4, cex.axis=1.2)
> lines(iwin$mids, iwin$counts/1e6, col="darkgreen", lwd=2)
> lines(ssin$mids, ssin$counts/2e6, col="blue", lwd=2)
> legend("topright", c("inward", "outward", "same"), col=c("darkgreen", "red", "blue"),
+     cex=1.3, lwd=2)
```



The spike in outward-facing tag pairs is consistent with self-ligation events. This occurs when a DNA fragment ligates to itself to form a circle. Subsequent cleavage and sequencing over the ligation junction can only generate outward-facing pairs upon alignment. In contrast, inter-ligation events between DNA fragments can generate any of the strand orientation. This results in the equality that is observed between the distributions at higher gap sizes.

In theory, no skews in strand orientation should exist for the AB library. This is because all AB pairs should be formed by inter-ligation events. Nonetheless, self-ligation spikes will still be observed for the AB library. This is probably due to spillover from linker assignment errors. The majority of AB tag pairs are inter-chromosomal so the absolute number of spillover pairs is usually negligble, regardless of the size of the peak in those plots.

2.3.2 Mopping up outward-facing tag pairs

Self-ligation events provide no information with respect to interactions between DNA fragments. Moreover, they can confound the analysis by masking genuine interactions during counting. Any tag pairs corresponding to self-ligation events should be removed with the `stripOutwardPET` function. This is done by removing all pairs contributing to the spike in the plot above, i.e., all outward-facing pairs at gap distances below 25 kbp.

```
> stripfile <- sub("\\.h5", "_strip.tsv", outcoming)
> stripped <- stripOutwardPET(outcoming, min.gap=25000, discard.to=stripfile)
> stripped
```

```
[1] 2259071
```

This will overwrite the old tag file with the stripped results. The stripped results can be diverted to a new tag file by setting the `file.out` parameter, if further work on the original results is required. The return value of the function represents the number of tag pairs discarded in this manner. This is usually a substantial proportion of all pairs in the original tag file. Here, the discarded pairs are also routed into `stripfile` for later use.

Chapter 3

Counting tag pairs into interactions

3.1 Motivation

The aim of the ChIA-PET data analysis is to identify specific interactions between pairs of genomic binding sites. The evidence for an interaction can be summarized in terms of the number of tag pairs connecting the corresponding binding sites. Recall that each tag pair is associated with a linker combination. Specific interactions are those where the homo-linker count (i.e., AA or BB) is significantly greater than the hetero-linker count (i.e., AB). This is because the latter can only be formed from non-specific ligation between complexes.

The analysis will be demonstrated using the full MCF7 dataset. Some work is necessary to determine which files contain the hetero-linker counts, and which files correspond to a single library. These identifiers will be used later to collapse counts for each linker combination into homo- or hetero-linker counts for each library.

```
> curfiles <- c("SRR372741_AA.h5", "SRR372741_AB.h5", "SRR372741_BB.h5",
+              "SRR372742_AA.h5", "SRR372742_AB.h5", "SRR372742_BB.h5")
> is.het <- grepl("AB", curfiles)
> libname <- sub("_[AB][AB].h5", "", basename(curfiles))
> data.frame(curfiles, is.het, libname)
```

```
         curfiles is.het   libname
1 SRR372741_AA.h5  FALSE SRR372741
2 SRR372741_AB.h5   TRUE SRR372741
3 SRR372741_BB.h5  FALSE SRR372741
4 SRR372742_AA.h5  FALSE SRR372742
5 SRR372742_AB.h5   TRUE SRR372742
6 SRR372742_BB.h5  FALSE SRR372742
```

3.2 Using windows or bins

3.2.1 Overview

Tag pairs must be summarized into counts for interactions prior to statistical analysis. The simplest strategy for doing so is to count tag pairs into pairs of bins. Partition the genome into contiguous and non-overlapping bins of equal size. For each pair of bins, the number of tag pairs with one tag mapped to each bin can be counted. This is performed for each linker combination in each library such that a set of counts is obtained for each bin pair.

```
> countwidth <- 5000
> actual <- countPET(curfiles, width=countwidth, filter=10)
> actual

IList object for 6 libraries with 27364 pairs across 627478 regions

Counts:
     [,1] [,2] [,3] [,4] [,5] [,6]
[1,]    3    0    3    2    0    5
[2,]    6    0    3    4    0    3
[3,]   11    0    5    7    0    7
[4,]   15    0   16   12    0   16
[5,]    3    0    6    4    0    2
... and 27359 more rows

Information:
   totals          files
1 6158575 SRR372741_AA.h5
2 2312522 SRR372741_AB.h5
3 6052074 SRR372741_BB.h5
4 6224677 SRR372742_AA.h5
5 2342675 SRR372742_AB.h5
6 6120574 SRR372742_BB.h5

Anchors:
   Chr  Start    End
1 chr1 845001 850000
2 chr1 845001 850000
3 chr1 850001 855000
4 chr1 850001 855000
5 chr1 855001 860000
... and 27359 more rows

Targets:
   Chr  Start    End
1 chr1 840001 845000
2 chr1 845001 850000
3 chr1 845001 850000
4 chr1 850001 855000
```

```
5 chr1 845001 850000
... and 27359 more rows
```

The returned `IList` object contains the counts for each bin pair in each tag file. Each row of the count matrix represents one bin pair, and each column represents a corresponding entry in the input `curfiles`. Each bin pair consists of one anchor and one target bin, both of which can be obtained with the `anchors` and `targets` methods. To avoid redundant permutations, the anchor bin is defined as that with the higher genomic coordinate. Information about each tag file, such as the total number of PETs, is contained in the `info` slot.

The function can also be used to count tag pairs across pairs of sliding windows across the genome. This provides greater spatial resolution than bins as optimal counting can be achieved for features that cross bin boundaries. However, it is more laborious and generates overlapping window pairs that are difficult to interpret. The gains in resolution are minor given the sparsity of tag pairs throughout the genome-by-genome interaction space.

3.2.2 Consequences of reverse read extension

The strategy described above considers a tag as "within" a bin if the $3'$ end of the tag is contained within the bin. The $3'$ end is more important than the $5'$ end as the former marks the end of the sonicated DNA fragment whereas the latter just marks the *Mme*I cut site [Fullwood et al., 2010]. If a bin overlaps a $3'$ end, it means that the bin spans part of the immunoprecipitated fragment. In contrast, overlaps to the $5'$ end do not have any obvious interpretation. Of course, this makes little practical difference for short 20 bp tags.

The gap between two $3'$ ends in an outward-facing read pairs represents the interval spanned by the DNA in the immunoprecipitated chromatin fragment. This suggests that the location of the peak in the strand orientation plot of Section 2.3.1 represents the average fragment length in the chromatin complexes after sonication and immunoprecipitation. In this case, the average fragment length is around 1 kbp. Each tag can then be extended from its $3'$ end *in the reverse direction of the strand* to this average length. The reverse-extended tag then represents the putative interval spanned by the original fragment.

This reverse extension can be performed by setting the `ext` parameter to the average fragment length in `countPET`. Counts are then defined for each bin pair as the number of tag pairs where one extended tag overlaps each bin. In practice, simply increasing the width of each bin by twice the average fragment length will have the same effect (see below).

3.2.3 Choosing an appropriate width

The width of each bin in `countPET` must be specified by the user during counting. A large `width` will count more tag pairs and increase detection power in downstream testing. This comes at the expense of spatial resolution whereby adjacent events can no longer be distinguished. In general, loss of resolution can be damaging as the count for the feature of interest becomes contaminated with irrelevant counts from adjacent features. This can dilute or mask significant differences between the homo- and hetero-linker counts.

For most ChIA-PET datasets, modest overestimation of the width will have little effect as the degree of contamination is limited by the sparsity of tags in the surrounding interaction space. This argument also justifies the use of a larger `width` rather than `ext`. The only difference between the two parameters is that reverse extension is strand-specific whereas the width is not. This will not have a major effect on the results due to the aforementioned sparsity. Given that the average fragment length is around 1 kbp, a width of 5 kbp is used here so that all tags associated with a binding site are likely to be counted in a bin.

3.2.4 Filtering for computational efficiency

Only bin pairs with a sum of counts across all libraries above `filter` will be retained. A reasonably large filter is necessary to avoid reporting an excessive number of uninteresting bin pairs that contain very few (e.g., single-digit) tag pairs. Otherwise, the downstream analyses will be very computationally intensive for little practical gain. Conversely, the filter value should not be too high, otherwise genuine interactions will be discarded. The appropriate choice of filter statistic and value is discussed in more depth in Section 3.5.2.

3.3 Using peaks

3.3.1 Peak calling from outward-facing pairs

An alternative approach is to identify binding sites using peak calling programs like MACS [Zhang et al., 2008]. These can be run on the outward-facing tag pairs that were previously pruned out of the tag file. This is equivalent to the cluster identification step in the ChIA-PET software tool [Li et al., 2010]. Outward-facing pairs are more reliable as the presence of both tags at a site indicates that they are more likely to be properly mapped.

3.3.2 Pooling to a BED file

Here, the outward-facing pairs from all tag files are pooled together and stored in a BED file. Pooling is recommended as it ensures that a single set of peaks can be obtained from the entire dataset. This means that consolidation of multiple peak sets is not required. It also increases the number of tags involved for sensitive peak calling.

```
> require(rtracklayer)
> discard.files <- list.files(".", pattern="_strip.tsv$")
> output.file <- "pooled.bed"
> start <- TRUE
> for (x in discardfiles) {
+     current <- read.table(x, stringsAsFactors=FALSE)
+     chosen.strand <- rbinom(nrow(current), 1, 0.5)==1L
+     chosen.pt <- ifelse(chosen.strand, current[,3], current[,2])
+     export(GRanges(current[,1], IRanges(chosen.pt, chosen.pt), strand=chosen.strand),
+         con=output.file, format="bed", append=!start)
+     start <- FALSE
+ }
```

Technically, all libraries should be pooled to maintain statistical validity during peak calling. This ensures that the called peaks are independent of the downstream hypothesis tests for specificity. In practice, self-ligation events should dominate the outward-facing tag pairs. This means that only homo-linker tag pairs need to be used for peak calling. Indeed, very few hetero-linker tag pairs should be present in the discarded set.

The tags themselves are stored as if they were generated from a ChIP-seq experiment. For each outward pair, the left tag is mapped to the reverse strand for a ChIA-PET experiment, but must be switched to the forward strand to mimic a ChIP-seq experiment. The same logic applies for the right tag. Only one tag is randomly taken from each fragment to mimic single-end data, given that many peak-callers cannot directly handle paired-end data.

3.3.3 Reading peaks and checking widths

So, assume that you got your peak-caller to save the results in `peak.file`. In this case, MACS [Zhang et al., 2008] was called with the options `-t pooled.bed -f BED -n here --nomodel`. The peaks can then be loaded into a `GRanges` object after peak calling, as shown below:

```
> peak.file <- "here_peaks.bed"
> peak.tab <- read.table(peak.file, stringsAsFactors=FALSE)
> peaks <- GRanges(peak.tab[,1], IRanges(peak.tab[,2], peak.tab[,3]))
> peaks

GRanges object with 6076 ranges and 0 metadata columns:
         seqnames                 ranges strand
            <Rle>              <IRanges>  <Rle>
     [1]     chr1     [ 851585,  857223]      *
     [2]     chr1     [ 900857,  902727]      *
     [3]     chr1     [ 934166,  936667]      *
     [4]     chr1     [ 955012,  956300]      *
     [5]     chr1     [1003968, 1006188]      *
     ...      ...                    ...    ...
  [6072]     chrX [153744105, 153744815]      *
  [6073]     chrX [153773862, 153776739]      *
  [6074]     chrX [153990407, 153991359]      *
  [6075]     chrX [154444087, 154445451]      *
  [6076]     chrX [154842051, 154842837]      *
  -------
  seqinfo: 27 sequences from an unspecified genome; no seqlengths
```

It is a good idea to check the widths of the identified peaks. For some peak callers, very small intervals will be identified so it may be necessary to expand the width of each peak. In this case, all peaks are expanded to 2 kbp in width and any overlapping peaks are merged. This minimum width is justified as being twice the (estimated) average fragment length for this dataset, such that all tags associated with a binding site will be counted.

```
> summary(width(peaks))

   Min. 1st Qu.  Median    Mean 3rd Qu.    Max. 
    289     916    1262    1464    1754    7238 

> peaks <- resize(peaks, fix="center", width=pmax(2000, width(peaks)))
> peaks <- reduce(peaks)
> summary(width(peaks))

   Min. 1st Qu.  Median    Mean 3rd Qu.    Max. 
   2000    2000    2000    2154    2000    7238 
```

For each pair of peaks, the number of tag pairs is counted based on the presence of the $3'$ end of one tag within each peak interval. This represents the number of connections between two putative binding sites. Counting can be performed using the `recountPET` function, along with some filtering on the count sum with `filter` as previously described. Reverse read extension can also be performed with `ext` but, again, this is probably unnecessary.

```
> peaked <- recountPET(curfiles, peaks, filter=10)
> peaked

IList object for 6 libraries with 2328 pairs across 6064 regions

Counts:
     [,1] [,2] [,3] [,4] [,5] [,6]
[1,]   21    0   26   32    0   20
[2,]   10    0    8    6    0   11
[3,]    3    0    1    2    0    6
[4,]    3    0    5    5    0    3
[5,]    9    0    4    5    0    4
... and 2323 more rows

Information:
   totals           files
1 6158575 SRR372741_AA.h5
2 2312522 SRR372741_AB.h5
3 6052074 SRR372741_BB.h5
4 6224677 SRR372742_AA.h5
5 2342675 SRR372742_AB.h5
6 6120574 SRR372742_BB.h5

Anchors:
   Chr   Start     End
1 chr1  851585  857223
2 chr1  934166  936667
3 chr1  954656  956655
4 chr1 1003968 1006188
5 chr1 1008607 1010606
... and 2323 more rows

Targets:
   Chr   Start     End
1 chr1  851585  857223
2 chr1  934166  936667
3 chr1  954656  956655
4 chr1 1003968 1006188
5 chr1 1003968 1006188
... and 2323 more rows
```

3.4 Aggregating homo- and hetero-linker counts

As previously mentioned, the aim of the analysis is to compare homo- and hetero-linker counts. The presence of separate AA and BB counts is redundant for this purpose. To simplify the count matrix, the homo-linker counts for each library can be added together. This is done with the `compressMatrix` function for either the binned or peak-based counts.

```
> freqs <- compressMatrix(actual, is.het, libname)
> freqs

IList object for 4 libraries with 27364 pairs across 627478 regions

Counts:
     SRR372741hom SRR372741het SRR372742hom SRR372742het
[1,]            6            0            7            0
[2,]            9            0            7            0
[3,]           16            0           14            0
[4,]           31            0           28            0
[5,]            9            0            6            0
... and 27359 more rows

Information:
    totals hetero   libname
1 14523171  FALSE SRR372741
2 14523171   TRUE SRR372741
3 14687926  FALSE SRR372742
4 14687926   TRUE SRR372742

Anchors:
   Chr  Start    End
```

```
1 chr1 845001 850000
2 chr1 845001 850000
3 chr1 850001 855000
4 chr1 850001 855000
5 chr1 855001 860000
... and 27359 more rows

Targets:
   Chr  Start    End
1 chr1 840001 845000
2 chr1 845001 850000
3 chr1 845001 850000
4 chr1 850001 855000
5 chr1 845001 850000
... and 27359 more rows
```

The total number of tag pairs in each *library* (not tag file) is also computed. Each total includes tag pairs from all linker combinations, and both homo-/hetero-linker columns from the same library are assigned the same total. This is appropriate as the total reflects the sequencing depth, which is constant for all counts derived from the same library.

3.5 Filtering to remove uninteresting features

3.5.1 By distance

Local interactions are frequently observed as pairs of the same or adjacent regions. These are usually uninteresting as they are inevitable consequences of chromatin compaction. The gap distance between the interacting regions can be computed with the `getDistance` function. This refers to the distance between the interacting regions on the same chromosome.

```
> gapped <- getDistance(freqs)
> summary(gapped)
```

```
     Min.   1st Qu.    Median      Mean   3rd Qu.      Max.      NA's 
    -5000     -5000         0     35380      5000 132000000       124 
```

Local interactions can then be discarded by putting a minimum threshold on the gap. This avoids confounding the results with many genuine yet uninteresting local interactions. For the binning strategy, setting a minimum gap of 1 or more will select for pairs of bins that are non-adjacent. Any inter-chromosomal pairs are marked as `NA` and must also be included.

```
> keep <- gapped > 1 | is.na(gapped)
> freqs <- freqs[keep,]
> sum(keep)
```

```
[1] 7896
```

3.5.2 By abundance

Low-abundance interactions are those with low tag pair counts. These do not provide enough evidence for rejection of the null hypothesis during later testing. They are also more likely to represent weak and uninteresting non-specific ligation events. Removal of these interactions can improve the sensitivity of the analysis by reducing the total number of tests and mitigating the severity of the multiple testing correction. Filtering on the average count (as computed by the aveLogCPM function in the edgeR package [McCarthy et al., 2012]) is recommended for statistical analyses using a negative binomial (NB) model.

```
> require(edgeR)
> ab <- aveLogCPM(asDGEList(freqs))
> summary(ab)

   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
 -1.699  -1.621  -1.476  -1.300  -1.168   3.530
```

The average count is useful as it is approximately independent of the p-value in edgeR [Lun and Smyth, 2014]. This ensures that the filtering procedure will not affect the validity of the ensuing statistical analysis. Of course, some caution is required lest all the interesting interactions be removed. In this case, a relatively mild threshold is used given that the bulk of filtering has already been performed with `filter` during count loading.

```
> keep <- ab > -1
> freqs <- freqs[keep,]
> summary(keep)

   Mode   FALSE    TRUE    NA's
logical    6491    1405       0
```

A more educated choice of filter threshold can be obtained by identifying genuine interactions as those that have higher average abundances than a non-specific interaction. If one assumes that most of the interaction space is filled with non-specific contacts, it is simple to calculate the threshold based on the abundance of those contacts. Note that larger bins are required to count non-specific contacts due to the sparsity of the interaction space.

```
> binsize <- 1e6
> binned <-countPET(curfiles, width=binsize, filter=1)
> rebin <- compressMatrix(binned, hetero=is.het, libname=libname)
```

The abundance of larger bin pairs can be calculated and compared to the values in `ab`. Some adjustments are necessary to account for the differences in the areas of the interaction space that are used for tag pair collection with each bin size. The median of the adjusted values is used as the estimate for the rate of non-specific ligation. Here, each smaller bin pair is only retained if its abundance is 5 times higher than the non-specific estimate.

```
> require(csaw)
> threshold <- median(scaledAverage(asDGEList(rebin), scale=(binsize/countwidth)^2))
> keep <- ab >= threshold + log2(5)
> summary(keep)

   Mode   FALSE    TRUE    NA's
logical    7317     579       0
```

Chapter 4

Normalizing linker combinations

4.1 Motivation

Some normalization is necessary prior to making comparisons between homo- and hetero-linker counts. This accounts for biases introduced by non-equal proportions of the two linkers, as well as any effects of non-random ligation. Failure to normalize will lead to incorrect conclusions. In particular, the nominal expected ratio of homo- to hetero-linker counts is 1:1 in a naïve comparison. The true expected ratio is often higher for a number of reasons (see below), so testing against the nominal ratio will result in many false positives.

4.2 Diagnosing random ligation

4.2.1 Using the Hardy-Weinberg law for normalization

Each DNA fragment can be treated as a "diploid individual", such that each end represents an "allele". Random ligation between ends of DNA fragments is equivalent to random mating between individuals. Now, assume that the proportions of DNA fragments attached to linker A and B are a and $b = 1 - a$, respectively. Under the Hardy-Weinberg law, this means that the the proportion of AA, AB and BB would be a^2, $2ab$ and b^2, respectively.

This result is useful as it allows normalization to remove the effects of non-equal A and B proportions, i.e., $a \neq b$. Consider a case when $a = 0.5b$. The expected homo- to hetero-linker ratio would then be 5:4. If you failed to consider this effect, you would (incorrectly) expect a ratio of 1:1 for a naïve comparison. Any interactions with the true expected ratio would deviate from the incorrect expected ratio, increasing the proportion of false positives.

This problem can be avoided with normalization. The value of a can be empirically determined from the proportion of tags that were attached to linker A. The expected proportions for each linker can then be calculated based on the Hardy-Weinberg law. These proportions are easily transformed into an expected ratio for the homo- and hetero-linker counts. Alternatively, normalization can be performed to coerce the expected ratio to a standard 1:1

setup. In the example above, the expected ratio is 5:4. One could then normalize the data by dividing the homo-linker counts by 1.25 prior to any hypothesis testing.

4.2.2 Checking mitochondrial counts

The accuracy of the random ligation model can be assessed with `extractMito`. This counts the tag pairs mapped between the nuclear and mitochondrial chromosomes for each tag file. The assumption here is that the mitochondrial genome does not interact with nuclear chromosomes. This is fairly reasonable given that they belong to separate organelles. Thus, any counts observed here are attributable to non-specific (and presumably random) ligation.

```
> mitocounts <- sapply(curfiles, FUN=function(x) { sum(extractMito(x)) })
> mitocounts

SRR372741_AA.h5 SRR372741_AB.h5 SRR372741_BB.h5 SRR372742_AA.h5 SRR372742_AB.h5
           1987             813            2071            2052             872
SRR372742_BB.h5
           1975
```

It is fairly obvious that these counts do not follow the predicted Hardy-Weinberg proportions. This can be confirmed by using Pearson's chi-squared test. So, does this correspond to some fascinating new biology involving inter-organelle interactions? Well, probably not. It just means that non-specific ligation is not necessarily random.

```
> lib.1 <- mitocounts[1:3]
> prop.a <- (lib.1[1]*2 + lib.1[2])/sum(lib.1)/2
> expected <- c(prop.a^2, prop.a*(1-prop.a)*2, (1-prop.a)^2)
> expected

SRR372741_AA.h5 SRR372741_AA.h5 SRR372741_AA.h5
      0.2414519       0.4998513       0.2586968

> chisq.test(lib.1, p=expected)

        Chi-squared test for given probabilities

data:  lib.1
X-squared = 2161.135, df = 2, p-value < 2.2e-16
```

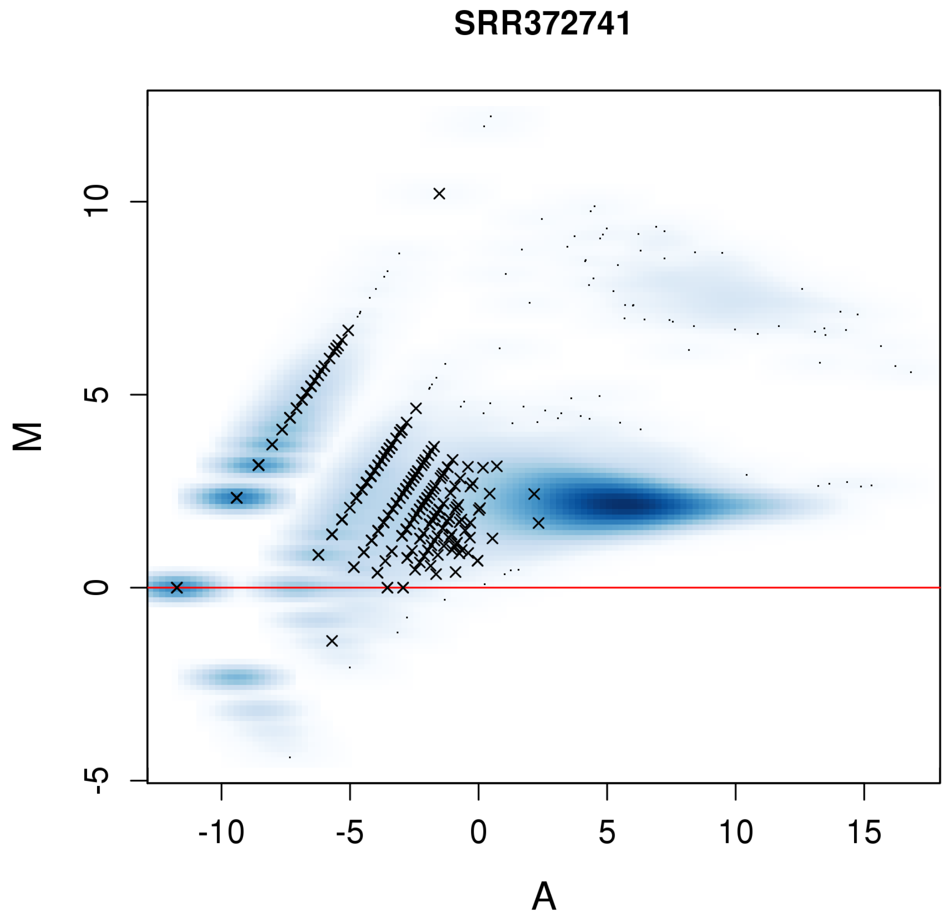
4.2.3 Checking counts across the interaction space

The mitochondrial results can be confirmed with counts from the rest of the dataset. Tag pairs are counted into pairs of 10 Mbp bins as described before. The assumption here is that most inter-chromosomal bin pairs only contain non-specific ligation events. This is generally reasonable as chromosomes are organized into self-contained territories within the nucleus [Bickmore, 2013]. This sterically limits the potential contacts between loci on different chromosomes. Again, large bins are necessary to count these rare non-specific events.

```
> binned <-countPET(curfiles, width=1e7, filter=1)
> rebin <- compressMatrix(binned, hetero=is.het, libname=libname)
> head(counts(rebin))

     SRR372741hom SRR372741het SRR372742hom SRR372742het
[1,]         8746           27         8814           20
[2,]          407           35          373           55
[3,]         6301           22         6445           25
[4,]          256           46          266           40
[5,]          337           43          349           41
[6,]         7964           23         7847           34
```

The M-value for each interaction is defined as the log-ratio of homo- to hetero-linker counts. This is used to construct a MA plot as shown below. If non-specific ligation is random, the expected M-value under the random ligation model should be observed (red line). However, the bulk of interactions have substantially higher M-values. This is consistent with the above-expected homo-linker counts for the nuclear-mitochondrial "interactions". That the inter-chromosomal M-values are generally equal to or less than the nuclear-mitochondrial M-values (crosses) also suggests that most inter-chromosomal contacts are non-specific.

```
> adj.counts <- cpm(asDGEList(rebin), log=TRUE)
> m <- adj.counts[,1]-adj.counts[,2]
> a <- adj.counts[,1]+adj.counts[,2]
> smoothScatter(x=a, y=m, xlab="A", ylab="M", main=info(rebin)$libname[1], cex.lab=1.4, cex.axis=1.2)
> abline(h=log2((expected[1]+expected[3])/expected[2]), col="red")
> has.mito <- as.logical(seqnames(anchors(rebin)) == "chrM" | seqnames(targets(rebin)) == "chrM")
> points(a[has.mito], m[has.mito], pch=4, cex=0.8)
```



This result supports the notion that non-specific ligation is not equivalent to random ligation. At the very least, non-randomness means that the random ligation model cannot be used for normalization. At worst, the ligation may be so non-random that there are no hetero-linker counts for any meaningful comparison. Note that different datasets may experience differing violations of the randomness assumption. Making diagnostic plots like the one shown above is recommended for routine quality control.

4.3 Normalizing for empirical effects

Normalization must account for both non-equal linker proportions and non-random ligation. This can be done empirically by assuming, again, that most inter-chromosomal interactions are non-specific. This means that the average M-value for these interactions can be used to define the expected homo-/hetero-linker ratio under the null hypothesis of non-specificity. In the MA plot above, most inter-chromosomal M-values are close to 2. One would then expect a homo-/hetero-linker count ratio of around 4:1 for a non-specific interaction.

Normalization can then be performed according to the expected ratio. Counts for inter-chromosomal bin pairs are passed to the trimmed mean of M-values (TMM) procedure [Robinson and Oshlack, 2010]. This returns a normalization factor for each homo-/hetero-linker column in each library. Counts are then (conceptually) divided by the normalization factor, and a comparison is directly performed between normalized counts. For example, an expected ratio of 4:1 would be normalized by dividing the homo-linker counts by 4.

```
> inters <- rebin[is.na(getDistance(rebin)),]
> normfacs <- normalize(inters)
> normfacs

[1] 2.1191635 0.4723837 2.1132767 0.4726985
```

Note that only inter-chromosomal interactions are used in the TMM procedure. This weakens the assumption of a non-specific majority as genuine interactions between chromosomes are rarer than those within chromosomes. In the MA plot above, all inter-chromosomal interactions lie within the main bulk of M-values at $\sim$2 whereas only intra-chromosomal events are in the cloud at $\sim$7. This is consistent with the greater specificity of the latter.

Chapter 5

Assessing biological variability

5.1 Motivation

The availability of biological replicates means that the biological variability of the system can be modelled. This reduces the significance of detected features when the data is highly variable. For count-based data, this can be achieved using the NB model in edgeR [Robinson et al., 2010]. Estimation of the NB dispersion parameter allows modelling of the variation between biological replicates. Similarly, estimation of the quasi-likelihood (QL) dispersion can be performed to account for heteroskedasticity [Lund et al., 2012].

Dispersion estimation requires fitting of a generalized linear model (GLM) to the counts for each interaction [McCarthy et al., 2012]. To this end, the choice of a design matrix is critical. Multiple designs are available to parameterize the combination of homo-/hetero-linker and library. The effect of these choices will be discussed in the following section. Firstly, it is necessary to assemble a `DGEList` object for entry into edgeR:

```
> y <- asDGEList(freqs, norm.factors=normfacs)
```

5.2 The paired-sample design

The most obvious choice is a paired-samples design, where the homo- and hetero-linker counts from a single library are paired together. The aim is to identify differences between the homo- and hetero-linker counts within each pair. This design accommodates library-specific effects, i.e., differences in the absolute abundance between libraries will not affect the result as long as the homo-/hetero-linker count ratio is constant between libraries. This avoids any inflation of the dispersions due to batch effects between libraries.

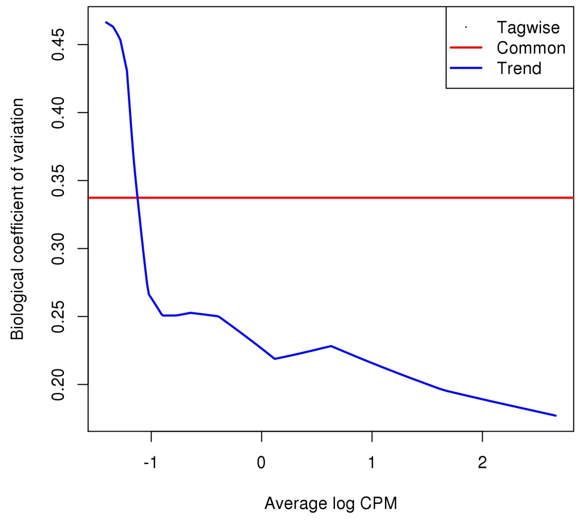
```
> design.paired <- model.matrix(~factor(info(freqs)$libname) + factor(info(freqs)$hetero))
> design.paired
```

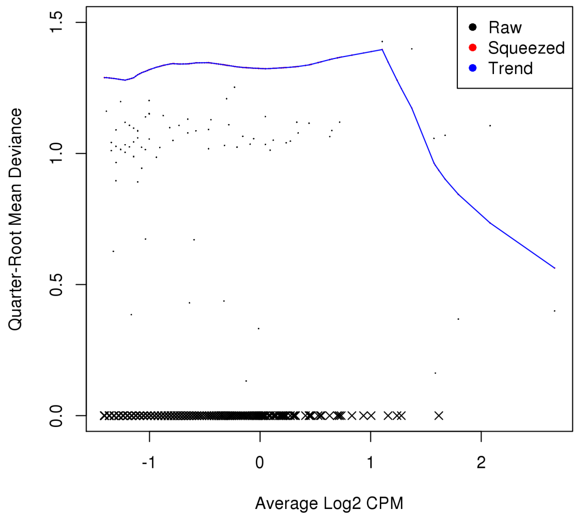
```
  (Intercept) factor(info(freqs)$libname)SRR372742
1           1                                    0
2           1                                    0
3           1                                    1
4           1                                    1
  factor(info(freqs)$hetero)TRUE
1                              0
2                              1
3                              0
4                              1
attr(,"assign")
[1] 0 1 2
attr(,"contrasts")
attr(,"contrasts")$`factor(info(freqs)$libname)`
[1] "contr.treatment"

attr(,"contrasts")$`factor(info(freqs)$hetero)`
[1] "contr.treatment"
```

Estimation of the dispersions can then be performed using methods in the edgeR package. The `estimateDisp` function will estimate the NB dispersion whereas the `glmQLFit` function will estimate the QL dispersion. In both cases, the low number of replicates means that limited information is available to estimate the dispersion for each interaction. Thus, an empirical Bayes strategy is used to stabilise the estimates by sharing information between interactions. This is represented by the estimation of a mean-dispersion trend (for NB) and shrinkage of the per-interaction dispersion estimates towards a trend (for QL).

```
> y.paired <- estimateDisp(y, design.paired)
> plotBCV(y.paired)
> fit.paired <- glmQLFit(y.paired, design.paired, robust=TRUE)
> plotQLDisp(fit.paired, ylim=c(0, 1.5))
> is.zero <- fit.paired$df.residual.zeros==0
> points(fit.paired$AveLogCPM[is.zero], fit.paired$deviance[is.zero], pch=4)
```

Note that the biological coefficient of variation (BCV) is defined as the square root of the NB dispersion. This represents the proportion of the counts that is attributable to biological variability. The QL dispersion is estimated from the GLM deviance and is shown as such, with a quarter-root transformation applied for maximum visibility across the range of values. Crosses indicate those bin pairs with no residual degrees of freedom in the fit.





For real datasets, the paired-sample design is not effective due to the frequency of zeros for the hetero-linker counts. This means that no residual degrees of freedom are available for dispersion estimation. Conceptually, you can imagine that the paired-sample design measures the variability of the homo-/hetero-linker fold change across libraries. Zeros for the hetero-linker counts means that the fold-change for each library will be undefined. This manifests as zero values for most of the QL dispersions in the plot above.

It may be possible to overcome this problem by adding a prior value onto the counts. This will avoid zeros and allow passage through the QL framework. However, this is a fairly arbitrary solution as the estimated dispersion will be sensitive to the choice of prior. It will also lead to underestimation of the dispersion, as the hetero-linker count will become a non-zero constant that does not reflect the true variability of the counts.

5.3 The one-way design

Instead, routine ChIA-PET analyses should use a one-way layout for the design matrix. All hetero-linker counts are treated as a single group, and all homo-linker counts are treated as another group. This means that the NB and QL dispersions can be properly estimated from the non-zero homo-linker counts, even when the hetero-linker counts are all zero.

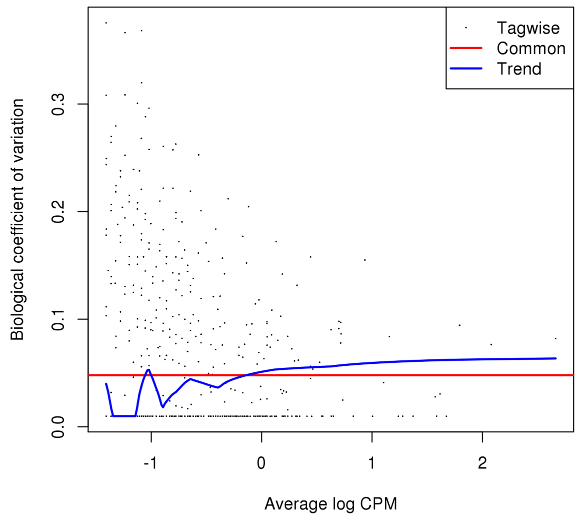
```
> design.group <- model.matrix(~factor(info(freqs)$hetero))
> design.group
```

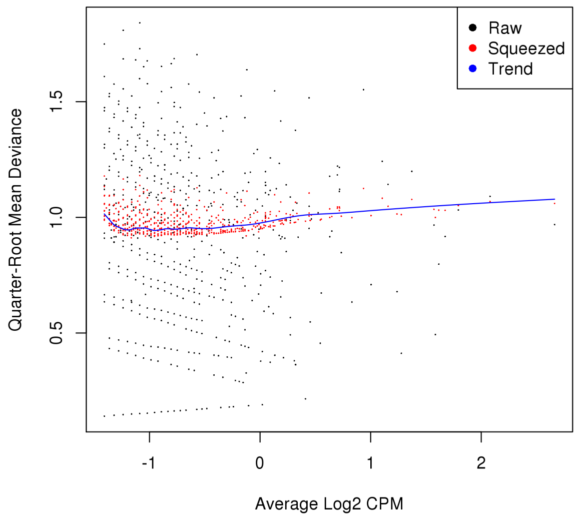
```
  (Intercept) factor(info(freqs)$hetero)TRUE
1           1                              0
2           1                              1
```

```
3          1                              0
4          1                              1
attr(,"assign")
[1] 0 1
attr(,"contrasts")
attr(,"contrasts")$`factor(info(freqs)$hetero)`
[1] "contr.treatment"
```

Estimation can be performed as previously described. This approach avoids the zero-valued QL dispersions observed for the paired-sample design. Of course, it assumes that there are no library-specific effects. For example, if the abundance changed randomly between libraries but the homo-/hetero-linker fold change was constant, the dispersions would be inflated in the one-way design but not in the paired-sample design. The homo-/hetero-linker group means would also be correlated, which could cause problems in downstream testing.

```
> y.group <- estimateDisp(y, design.group)
> plotBCV(y.group)
> fit.group <- glmQLFit(y.group, design.group, robust=TRUE)
> plotQLDisp(fit.group)
```





Based on these results, inflation of the NB dispersion is not a problem here. In fact, the NB dispersion is exceptionally low given that biological replicates are being compared. For other types of sequencing data like RNA-seq or ChIP-seq, dispersion estimates are usually around 0.01 to 0.1. Much lower values are observed here, i.e., < 0.01 after squaring the BCV. This is a bit odd, as ChIA-PET should be at least as variable as ChIP-seq. Oh well.

Chapter 6

Testing for significant interactions

6.1 Using the quasi-likelihood F-test

The `glmQLFTest` function performs a QL F-test for each interaction, to identify those with significant differences between homo- and hetero-linker counts. Users should make sure that they are specifying the contrast correctly (see the edgeR user's guide for more details). For the one-way layout, the coefficient of interest is that corresponding to the homo-/hetero-linker fold change. This should be specified through the `coef` or `contrast` arguments.

```
> result.group <- glmQLFTest(fit.group, coef=2)
> topTags(result.group)

Coefficient:  factor(info(freqs)$hetero)TRUE
        logFC   logCPM         F       PValue          FDR
623 -5.177966 2.663500 151.19105 1.366824e-07 0.0001920387
928 -6.447362 2.080046 108.92669 6.831143e-07 0.0004798878
536 -4.924534 1.585051  89.22179 1.450609e-06 0.0005129070
629 -5.979711 1.673456  92.32164 1.514744e-06 0.0005129070
588 -5.166984 1.793080  86.95368 2.015201e-06 0.0005129070
601 -5.862771 1.572693  83.76649 2.405672e-06 0.0005129070
582 -9.506975 1.616495  97.28198 2.555409e-06 0.0005129070
658 -9.113536 1.277334  81.44872 4.497506e-06 0.0007898746
576 -9.068187 1.238391  78.32396 6.555742e-06 0.0010234242
213 -8.793938 1.004389  67.35305 1.248657e-05 0.0017543628
```

Recall that the null hypothesis is that the (normalized) hetero-linker counts are the same as the homo-linker counts. The computed p-value represents the evidence against this null for each interaction. The log-fold change represents that of the hetero-linker count over the homo-linker count. In this case, they are negative for all interactions. This corresponds to more homo-linker counts and is consistent with the presence of specific interactions.

```
> summary(result.group$table$logFC)
```

```
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max. 
-9.5070 -6.5540 -6.1190 -6.1640 -5.9000 -0.9854 
```

Discerning users may also be wondering why we are not using the likelihood ratio test (LRT). Indeed, the LRT is the more obvious test for hypothesis testing in the GLM framework. However, the QL F-test is preferred here as it accounts for the variability and uncertainty of the QL dispersion estimates [Lund et al., 2012]. This means that it can maintain type I error control in the presence of heteroskedasticity whereas the LRT does not.

6.2 Multiplicity correction and the FDR

Correction for multiple tests is performed by controlling the false discovery rate (FDR), using the method described by Benjamini and Hochberg [1995]. In this case, the FDR refers to the proportion of detected interactions that are false positives. Control of the FDR is often more appropriate than control of the family-wise error rate (i.e., probability of one or more false rejections across all interactions). This is because the former provides a more appropriate compromise between specificity and power in genome-wide analyses.

```
> adj.p <- p.adjust(result.group$table$PValue, method="BH")
> sum(adj.p <= 0.05)
```

```
[1] 1394
```

Significantly specific interactions are defined as those that are detected at an FDR of 5%. These can be saved to file as necessary. In practice, all interactions should be sorted by the *p*-value and saved to file. This means that results can be queried in a flexible manner without needing to re-run the entire analysis, e.g., if the FDR threshold is changed.

```
> ax <- anchors(freqs)
> tx <- targets(freqs)
> final <- data.frame(anchor.chr=seqnames(ax), anchor.start=start(ax), anchor.end=end(ax),
+     target.chr=seqnames(tx), target.start=start(tx), target.end=end(tx),
+     result.group$table, FDR=adj.p)
> o <- order(final$PValue)
> write.table(final[o,], file="results.tsv", sep="\t", quote=FALSE, row.names=FALSE)
```

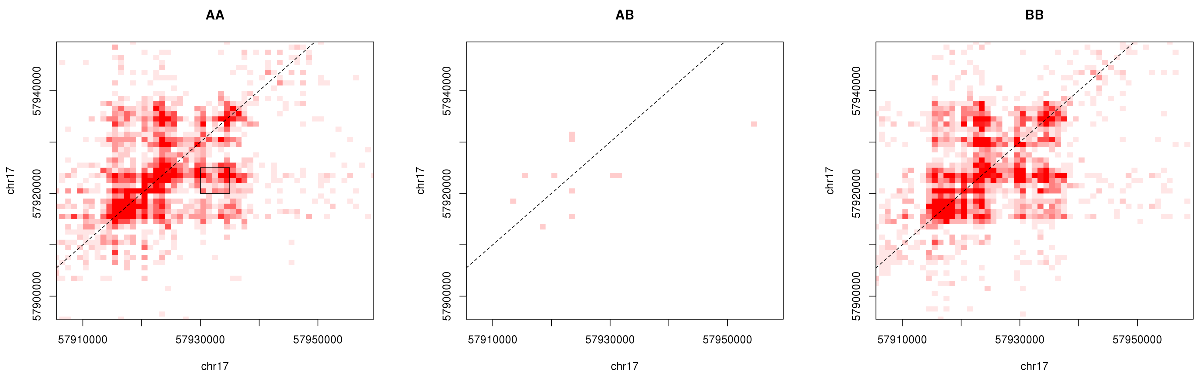
Note that the interactions of interest are those where the homo-linker count is greater than the hetero-linker count. Power can be improved by filtering out all bin pairs with an inappropriate sign for the log-fold change, prior to applying the BH method. This removes uninteresting tests and reduces the severity of the correction. That said, filtering has no effect in this particular dataset as there are no bin pairs with a positive log-fold change.

6.3 Visualization with plaid plots

There comes a time that some visualization of interesting results is desirable. This can be performed using the `plotChIA` function, which constructs a plaid plot of the interaction space [Lieberman-Aiden et al., 2009]. Briefly, each axis is a chromosome segment. The box represents an interaction between the corresponding points on each axis. The colour of the box is proportional to the number of tag pairs mapped between the interacting loci.

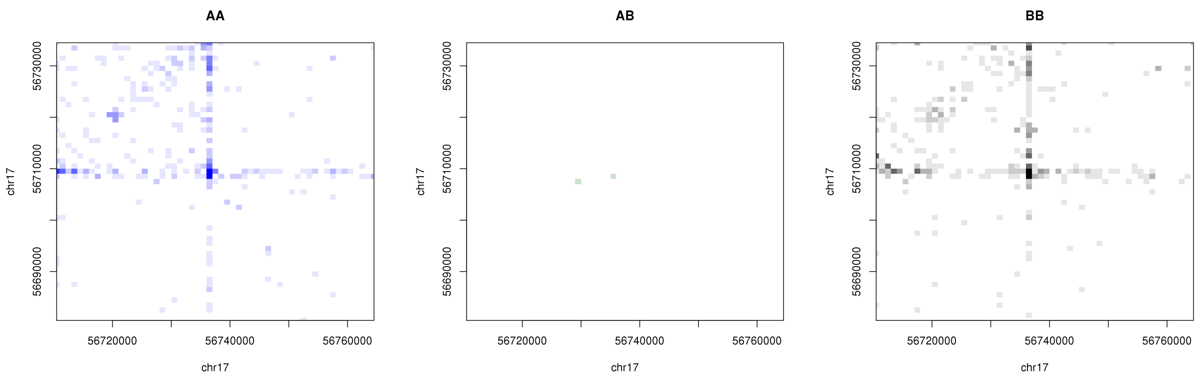
```
> expanded.a <- resize(ax[o[1]], fix="center", width=50000)
> expanded.t <- resize(tx[o[1]], fix="center", width=50000)
> plotChIA(curfiles[1], anchor=expanded.a, target=expanded.t, main="AA", cap=10)
> rect(start(ax[o[1]]), start(tx[o[1]]), end(ax[o[1]]), end(tx[o[1]]))
> abline(0, 1, lty=2)
> plotChIA(curfiles[2], anchor=expanded.a, target=expanded.t, main="AB", cap=5)
> abline(0, 1, lty=2)
> plotChIA(curfiles[3], anchor=expanded.a, target=expanded.t, main="BB", cap=10)
> abline(0, 1, lty=2)
```

Expansion of the plot boundaries is recommended. This shows the context of the interaction by examining the features in the surrounding interaction space. It is also possible to tune the width of the boxes through a parameter that is, rather unsurprisingly, named `width`. The dotted line represents the diagonal around which the plot is symmetric. The actual interaction occurs at the center of the plot and is marked by a rectangle in the first plot.



The `cap` value controls the relative scale of the colours. A smaller `cap` is necessary for the AB track to normalize the intensity, much like that discussed in Section 4.3. In this case, a homo-/hetero-linker ratio of 4:1 is expected under the null, so the cap is (10+10):5. This means that the intensity of colours in the AB plot can be directly compared to those of AA and BB. Upon doing so, you'll find that the intensity of the homo-linkers is much greater than that of the hetero-linkers for this part of the interaction space. This suggests that the interaction is specific and genuine. Here's another example in a different colour:

```
> expanded.a <- resize(ax[o[3]], fix="center", width=50000)
> expanded.t <- resize(tx[o[3]], fix="center", width=50000)
> plotChIA(curfiles[1], anchor=expanded.a, target=expanded.t, main="AA", cap=10, col="blue")
> plotChIA(curfiles[2], anchor=expanded.a, target=expanded.t, main="AB", cap=5, col="darkgreen")
> plotChIA(curfiles[3], anchor=expanded.a, target=expanded.t, main="BB", cap=10, col="black")
```



Tying this in with gene annotation, ChIP-seq peaks and/or RNA-seq data can then suggest some functions for these interactions, e.g., enhancer loops, gene loops, gene-gene interactions. Of course, users should always keep the identify of the target protein in mind. This particular experiment targets RNA polymerase II so most activity will occur at genes and have a transcriptional focus. Other protein targets are likely to have different profiles.

Chapter 7

Repeating with differential analyses

7.1 Loading the homo-linker counts

The previous analysis detects significantly specific interactions by comparing the homo- and hetero-linker counts within a single condition. True differential analyses involve comparison of counts between libraries for different conditions. This can be more useful as differential interactions are more likely to be relevant to the biological difference of interest. To demonstrate, we'll use data from a ChIA-PET experiment targeting RNA polymerase II in MCF7 and K562 cells [Li et al., 2012]. Homo-linker counts are first counted into bin pairs:

```
> curfiles <- c("SRR372741_AA.h5", "SRR372741_BB.h5",
+               "SRR372742_AA.h5", "SRR372742_BB.h5",
+               "SRR372747_AA.h5", "SRR372747_BB.h5",
+               "SRR372748_AA.h5", "SRR372748_BB.h5")
> actual <-countPET(curfiles, width=countwidth, filter=10)
> comp.freq <- compressMatrix(actual)
> head(counts(comp.freq))

     SRR372741hom SRR372742hom SRR372747hom SRR372748hom
[1,]            0            0            7            3
[2,]            2            1           10           10
[3,]            0            0            5            5
[4,]            0            0            3            7
[5,]            0            0            5            6
[6,]            0            0            4            6
```

The aim is to identify interactions where the homo-linker counts for MCF7 and K562 cells are significantly different. This is an effective strategy as homo-linker counts are large enough for stable inference in most ChIA-PET datasets. In theory, hetero-linker counts could also be used to account for library-specific differences in ligation specificity. However, these counts will not be included here as they are too small to be informative. This is analogous to the discussion about paired-sample and one-way designs in Section 5.2.

7.2 Filtering on abundance and distance

Filtering is performed to remove low-abundance interactions for which detection power will always be low. Local interactions are also removed as they provide no information with respect to chromatin organization. This is equivalent to the methods in Section 3.5.2.

```
> min.gap <- getDistance(comp.freq)
> keep <- aveLogCPM(asDGEList(comp.freq)) > 0 & (min.gap > 1 | is.na(min.gap))
> comp.freq <- comp.freq[keep,]
> summary(keep)
```

```
   Mode   FALSE    TRUE    NA's
logical  343003    2599       0
```

7.3 Normalizing differences between libraries

Some normalization is required to correct for systematic biases prior to hypothesis testing. One approach is to assume that most interactions are constant between MCF7 and K562 cells. This means that the TMM method can be applied directly to the counts for the interactions. Prior filtering is critical to ensure that the counts are large enough for sensible normalization on the M-values (i.e., transformed fold changes). Of course, this is not applicable if one expects overall changes to the interaction structure between cell types.

```
> y.diff <- asDGEList(comp.freq)
> y.diff <- calcNormFactors(y.diff)
> y.diff$samples
```

```
             group lib.size norm.factors
SRR372741hom     1 12210649    0.5420361
SRR372742hom     1 12345251    0.5403574
SRR372747hom     1 12177055    1.8627789
SRR372748hom     1 12222496    1.8328605
```

Alternatively, an approach analogous to that in Section 4.3 can be used. No differences are expected between groups for non-specific interactions. Any differences in these counts must be driven by composition biases and other technical effects. To remove such biases, the majority of inter-chromosomal contacts are assumed to be non-specific. Reads are re-counted using large bins to provide sufficiently large counts for these weak contacts. The TMM method can then be applied to compute appropriate normalization factors.

```
> binned.diff <-countPET(curfiles, width=1e7, filter=1)
> rebin.diff <- compressMatrix(binned.diff)
> inter.counts <- asDGEList(rebin.diff[is.na(getDistance(rebin.diff)),])
> inter.counts <- calcNormFactors(inter.counts)
> normfacs <- inter.counts$samples$norm.factors
> normfacs
```

```
[1] 1.6107452 1.6076347 0.6224925 0.6203712
```

The ideal approach depends on whether systematic changes in interaction intensity are assumed to be genuine. If so, the second approach should be used as it only removes differences among the non-specific contacts. This preserves any overall differences between conditions for the specific interactions. Otherwise, if the differences are not expected, they are probably technical artifacts and should be removed with the first approach.

For this particular dataset, the second approach is chosen. Large-scale changes are to be expected when comparing two different immortalized cell lines. Note that the normalization factors computed with large bins can still be applied to the counts for the smaller bins. This is because the same composition biases will be present throughout the dataset.

```
> y.diff$samples$norm.factors <- normfacs
```

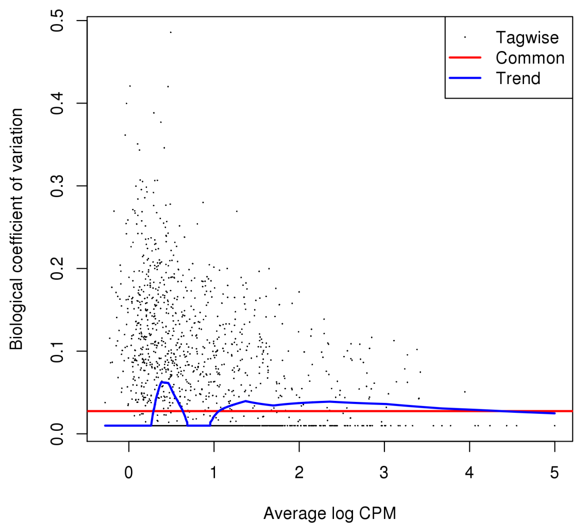
7.4 Running through the statistical analysis

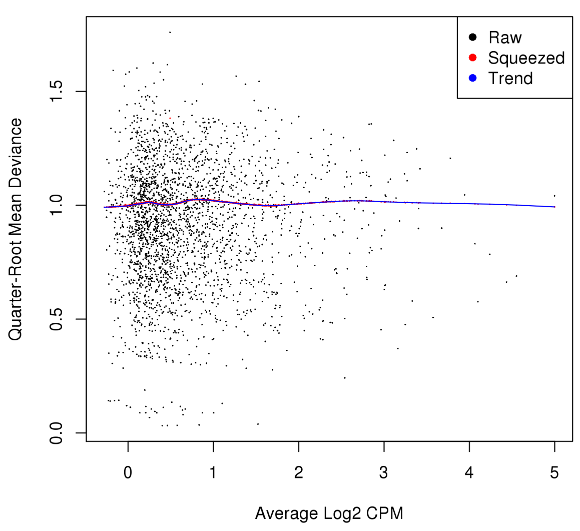
As previously mentioned, users should perform the statistical analysis with a one-way layout. Counts for each cell type are treated as replicates of a group. The ensuing analysis will compare counts between the cell types for each interaction. The null hypothesis is that the second coefficient is zero, i.e., the normalized mean counts for each group are identical.

```
> design.diff <- model.matrix(~factor(c("mcf7", "mcf7", "k562", "k562")))
> design.diff

  (Intercept) factor(c("mcf7", "mcf7", "k562", "k562"))mcf7
1           1                                             1
2           1                                             1
3           1                                             0
4           1                                             0
attr(,"assign")
[1] 0 1
attr(,"contrasts")
attr(,"contrasts")$`factor(c("mcf7", "mcf7", "k562", "k562"))`
[1] "contr.treatment"
```

Dispersion estimation and hypothesis testing can be performed as previously described, using the QL framework in the edgeR package. For the one-way layout, the contrast does not need to be specified as the last coefficient will be tested in `glmQLFTest`. Analyses with multiple groups should always provide an argument to `coef` or `contrasts` to avoid confusion.

```
> y.diff <- estimateDisp(y.diff, design.diff)
> plotBCV(y.diff)
> fit.diff <- glmQLFit(y.diff, design.diff, robust=TRUE)
> plotQLDisp(fit.diff)
> result.diff <- glmQLFTest(fit.diff)
```





7.5 Examination of the results

A summary of the results can then be obtained. Non-zero codes indicate the number of interactions that are significantly different at a FDR threshold of 5%. A negative code indicates that the interaction is stronger in the K562 cells, whereas a positive code indicates that it is strong in the MCF7 cells. Of course, one must be cautious when interpreting results from immortalized lines. Any changes may be driven by genomic rearrangements.

```
> summary(decideTestsDGE(result.diff))
```

```
   [,1]
-1 2273
0   124
1   202
```

It is a trivial matter to save these results for further perusal. The process of annotation and visualization will be skipped here for brevity, as it is the same as previously described.

```
> ax <- anchors(comp.freq)
> tx <- targets(comp.freq)
> final <- data.frame(anchor.chr=seqnames(ax), anchor.start=start(ax), anchor.end=end(ax),
+     target.chr=seqnames(tx), target.start=start(tx), target.end=end(tx),
+     result.diff$table, FDR=p.adjust(result.diff$table$PValue, method="BH"))
> o <- order(final$PValue)
> write.table(final[o,], file="dinter.tsv", sep="\t", quote=FALSE, row.names=FALSE)
```

Chapter 8

Epilogue

8.1 Session Information

```
> sessionInfo()

R version 3.1.1 (2014-07-10)
Platform: x86_64-unknown-linux-gnu (64-bit)

locale:
 [1] LC_CTYPE=en_US.UTF-8       LC_NUMERIC=C
 [3] LC_TIME=en_US.UTF-8        LC_COLLATE=en_US.UTF-8
 [5] LC_MONETARY=en_US.UTF-8    LC_MESSAGES=en_US.UTF-8
 [7] LC_PAPER=en_US.UTF-8       LC_NAME=C
 [9] LC_ADDRESS=C               LC_TELEPHONE=C
[11] LC_MEASUREMENT=en_US.UTF-8 LC_IDENTIFICATION=C

attached base packages:
 [1] splines   stats4    parallel  stats     graphics  grDevices utils
 [8] datasets  methods   base

other attached packages:
 [1] csaw_1.1.12          statmod_1.4.20       locfit_1.5-9.1
 [4] edgeR_3.9.5          limma_3.22.1         dacpet_0.0.6
 [7] GenomicRanges_1.18.1 GenomeInfoDb_1.2.3   IRanges_2.0.0
[10] S4Vectors_0.4.0      BiocGenerics_0.12.1

loaded via a namespace (and not attached):
 [1] AnnotationDbi_1.28.1    base64enc_0.1-2         BatchJobs_1.5
 [4] BBmisc_1.8              Biobase_2.26.0          BiocParallel_1.0.0
 [7] biomaRt_2.22.0          Biostrings_2.34.0       bitops_1.0-6
[10] brew_1.0-6              checkmate_1.5.0         codetools_0.2-9
[13] DBI_0.3.1               digest_0.6.4            fail_1.2
[16] foreach_1.4.2           GenomicAlignments_1.2.1 GenomicFeatures_1.18.2
[19] grid_3.1.1              iterators_1.0.7         KernSmooth_2.23-13
[22] lattice_0.20-29         RCurl_1.95-4.3          rhdf5_2.10.0
```

```
[25] Rsamtools_1.18.2        RSQLite_1.0.0           rtracklayer_1.26.2
[28] sendmailR_1.2-1         stringr_0.6.2           tools_3.1.1
[31] XML_3.98-1.1            XVector_0.6.0           zlibbioc_1.12.0
```

8.2 References

Y. Benjamini and Y. Hochberg. Controlling the false discovery rate: a practical and powerful approach to multiple testing. *J. Roy. Statist. Soc. B*, pages 289–300, 1995.

W. A. Bickmore. The spatial organization of the human genome. *Annu. Rev. Genomics Hum. Genet.*, 14:67–84, 2013.

M. J. Fullwood, Y. Han, C. L. Wei, X. Ruan, and Y. Ruan. Chromatin interaction analysis using paired-end tag sequencing. *Curr. Protoc. Mol. Biol.*, Chapter 21:1–25, Jan 2010.

F. Jin, Y. Li, J. R. Dixon, S. Selvaraj, Z. Ye, A. Y. Lee, C. A. Yen, A. D. Schmitt, C. A. Espinoza, and B. Ren. A high-resolution map of the three-dimensional chromatin interactome in human cells. *Nature*, 503(7475):290–294, Nov 2013.

B. Langmead and S. L. Salzberg. Fast gapped-read alignment with Bowtie 2. *Nat. Methods*, 9(4):357–359, Apr 2012.

G. Li, M. J. Fullwood, H. Xu, F. H. Mulawadi, S. Velkov, V. Vega, P. N. Ariyaratne, Y. B. Mohamed, H. S. Ooi, C. Tennakoon, C. L. Wei, Y. Ruan, and W. K. Sung. ChIA-PET tool for comprehensive chromatin interaction analysis with paired-end tag sequencing. *Genome Biol.*, 11(2):R22, 2010.

G. Li, X. Ruan, R. K. Auerbach, K. S. Sandhu, M. Zheng, P. Wang, H. M. Poh, Y. Goh, J. Lim, J. Zhang, H. S. Sim, S. Q. Peh, F. H. Mulawadi, C. T. Ong, Y. L. Orlov, S. Hong, Z. Zhang, S. Landt, D. Raha, G. Euskirchen, C. L. Wei, W. Ge, H. Wang, C. Davis, K. I. Fisher-Aylor, A. Mortazavi, M. Gerstein, T. Gingeras, B. Wold, Y. Sun, M. J. Fullwood, E. Cheung, E. Liu, W. K. Sung, M. Snyder, and Y. Ruan. Extensive promoter-centered chromatin interactions provide a topological basis for transcription regulation. *Cell*, 148 (1-2):84–98, Jan 2012.

E. Lieberman-Aiden, N. L. van Berkum, L. Williams, M. Imakaev, T. Ragoczy, A. Telling, I. Amit, B. R. Lajoie, P. J. Sabo, M. O. Dorschner, R. Sandstrom, B. Bernstein, M. A. Bender, M. Groudine, A. Gnirke, J. Stamatoyannopoulos, L. A. Mirny, E. S. Lander, and J. Dekker. Comprehensive mapping of long-range interactions reveals folding principles of the human genome. *Science*, 326(5950):289–293, Oct 2009.

A. T. Lun and G. K. Smyth. De novo detection of differentially bound regions for ChIP-seq data using peaks and windows: controlling error rates correctly. *Nucleic Acids Res.*, May 2014.

S. P. Lund, D. Nettleton, D. J. McCarthy, and G. K. Smyth. Detecting differential expression in RNA-sequence data using quasi-likelihood with shrunken dispersion estimates. *Stat. Appl. Genet. Mol. Biol.*, 11(5), 2012.

D. J. McCarthy, Y. Chen, and G. K. Smyth. Differential expression analysis of multifactor RNA-Seq experiments with respect to biological variation. *Nucleic Acids Res.*, 40(10): 4288–4297, May 2012.

M. D. Robinson and A. Oshlack. A scaling normalization method for differential expression analysis of RNA-seq data. *Genome Biol.*, 11(3):R25, 2010.

M. D. Robinson, D. J. McCarthy, and G. K. Smyth. edgeR: a Bioconductor package for differential expression analysis of digital gene expression data. *Bioinformatics*, 26(1):139–140, Jan 2010.

Y. Zhang, T. Liu, C. A. Meyer, J. Eeckhoute, D. S. Johnson, B. E. Bernstein, C. Nusbaum, R. M. Myers, M. Brown, W. Li, and X. S. Liu. Model-based analysis of ChIP-Seq (MACS). *Genome Biol.*, 9(9):R137, 2008.