dacpet: Differential analysis of ChIA-PET data

User's Guide

Aaron Lun

First edition 15 August 2012

Last revised 15 May 2014

Contents

1 Introduction 3

1.1 Scope . . . 3

1.2 How to get help . . . 3

1.3 Quick start . . . 3

2 Preparing tag directories 5

2.1 Splitting tags and aligning reads . . . 5

2.2 Converting BAM files to tag directories . . . 5

2.3 Diagnosing ligation quality . . . 6

2.3.1 Making strand orientation plots . . . 6

2.3.2 Mopping up outward-facing tag pairs . . . 8

3 Counting tag pairs into interactions 10

3.1 Motivation . . . 10

3.2 Using windows or bins . . . 11

3.2.1 Counting wth `countPET` . . . 11

3.2.2 Effects of reverse read extension . . . 11

3.2.3 Choosing an appropriate width . . . 12

3.2.4 Filtering for computational efficiency . . . 12

3.3 Using peaks . . . 12

3.3.1 Peak calling from outward-facing pairs . . . 12

3.3.2 Pooling to a BED file . . . 13

3.3.3 Reading peaks and checking widths . . . 14

3.4 Aggregating homo- and hetero-linker counts . . . 15

3.5 Filtering to remove uninteresting features . . . 16

3.5.1 By distance . . . 16

3.5.2 By abundance . . . 17

4 Normalizing linker combinations 18

4.1 Motivation . . . 18

4.2 Diagnosing random ligation . . . 18

4.2.1 Using the Hardy-Weinberg law for normalization . . . 18

4.2.2 Checking mitochondrial counts 19
4.2.3 Checking counts across the interaction space 19
4.3 Normalizing for empirical effects 21

5 Assessing biological variability 23
5.1 Motivation 23
5.2 The paired-sample design 23
5.3 The one-way design 25

6 Testing for significant interactions 27
6.1 Using the quasi-likelihood F-test 27
6.2 Multiplicity correction and the FDR 28
6.3 Visualization with plaid plots 28

7 References and other stuff 31
7.1 Session Information 31
7.2 References 32

Chapter 1

Introduction

1.1 Scope

This document describes the analysis of CHIA-PET data with the dacpet package. In particular, specific interactions are distinguished from non-specific ligation events by comparing the homo-linker and hetero-linker counts for each interaction. 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 is not necessary for reading.

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 Rprompt. If that doesn't help, reading this guide or contacting one of the authors is probably the best approach. Thoughtful suggestions for improvements are occassionally appreciated.

1.3 Quick start

As a quick demonstration of the analysis is largely impossible, I'll just leave this part empty. Suffice to say that there are several components of the pipeline that shall be discussed:

1. converting BAM files to tag directories

2. removing self-ligation events

3. counting tag pairs for each interaction

4. normalizing for the linker combinations

5. modelling biological variability

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

In the various examples for this guide, we'll be using data from an experiment targeting RNA polymerase II in MCF7 cells [Li et al., 2012].

Chapter 2

Preparing tag directories

2.1 Splitting tags and aligning reads

Raw ChIA-PET data shows up as a bunch of 36 bp read pairs, where each read has a linker and tag component. The linker sequence must be identified as either A or B. It must then be removed prior to alignment of the tag to the reference genome. Some external programs can be used to facilitate this process, including:

- splitLinkers: a C++ program that splits each pair of FastQ files into a set of pairs of files of tag sequences for the AA, AB and BB linker combinations.
- chia_map.sh: a Bash script that performs quality control with fastqc, linker splitting, tag alignment with Bowtie2 [Langmead and Salzberg, 2012] and duplicate marking with Picard's markDuplicates, given a set of SRA or Gzipped FastQ files.

You can probably find these programs, somewhere in cyberspace. The ultimate aim is to produce a name-sorted BAM file for each linker combination in each library, i.e., 3 BAM files for each incoming ChIA-PET library. For the MCF7 dataset, this results in 6 files overall:

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

Note that the three files named SRR372741_* correspond to a single library, as do those named SRR372742_*. The two libraries represent different biological replicates according to the GEO entry, and can be used to assess the reproducibility of the results.

2.2 Converting BAM files to tag directories

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

```
> incoming <- bamfiles[1]
> outcoming <- sub("\\.bam", "", incoming)
> out <- preparePET(incoming, dir=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 reported in the directory. A reasonably stringent value is recommended given that alignment of short tags is likely to be error-prone. The dedup value specifies whether or not marked duplicates should be removed. This is recommended as the chance of obtaining exactly overlapping pairs is low. Any overlaps are likely to be PCR duplicates rather than genuine enrichment. 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 
17741155   987571  9201336  7786131 
```

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 directory. You can also have a look at the number of single tags or those with more than two alignments. 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 as required.

2.3 Diagnosing ligation quality

2.3.1 Making strand orientation plots

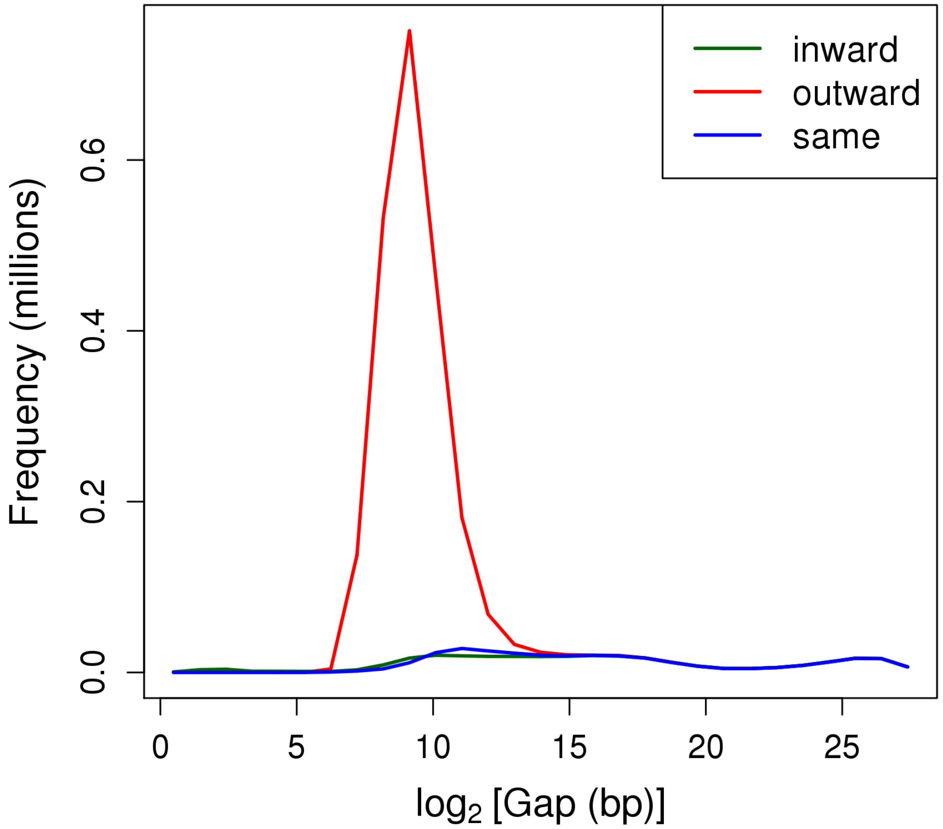
Some diagnostics can be pulled out of each processed directory using the diagnosePET function. This extracts the strand information for inter-chromosomal tag pairs, i.e., those mapped to different chromosomes. Each code refers to the combination of strands to which each pair is mapped, i.e., both forward (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 
1125040 1123944 1123849 1124109 
```

The function will also collect strand and gap information for intra-chromosomal tag pairs. In this case, the “first” tag is that which maps at a lower position. This means that a code of `1` marks an inward-facing tag pair whereas a code of `2` marks an outward-facing tag pair. 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 explains the balance that is observed between the distributions at higher gap sizes.

It's worth pointing out that no skews in strand orientation should exist for the AB library. This is because all AB pairs should be formed by inter-ligation events. The presence of any self-ligation spikes is probably caused by 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. The spike in the plot above is mostly gone at a gap distance of 25 kbp, so all outward-facing PETs with lower gap distances will be removed.

```
> stripfile <- paste0(outcoming, "_strip.tsv")
> stripped <- stripOutwardPET(outcoming, min.gap=25000, discard.to=stripfile)
> stripped
```

```
[1] 2202955
```

This will overwrite the old directory with the stripped results. The return value represents the number of tag pairs discarded in this manner. This is usually a substantial proportion of all pairs in the directory. Alternatively, the stripped results can be diverted to a new directory if further quality control on the original results is desired. 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 analysis is to identify specific interactions. This is achieved by comparing the homo-linker count for an interaction with the hetero-linker count. The aim is to find those interactions where the homo-linker count is significantly greater than the hetero-linker count, given that the latter can only be formed from non-specific ligation between complexes. This requires some manner of counting tag pairs for each interaction.

The various demonstrations here will use processed tag directories for the full MCF7 dataset. Some work is necessary to determine which directories contain the hetero-linker counts, and which directories are generated from the same library. These vectors of identifiers will be used later to collapse counts for each linker combination into homo-/hetero-linker counts for each library.

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

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

3.2 Using windows or bins

3.2.1 Counting wth `countPET`

The tag pairs must be summarized into counts for interactions prior to the statistical analysis. The simplest strategy for doing so is to count tag pairs into pairs of bins. Let the genome be partitioned 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(curdirs, width=countwidth, filter=10)
> head(actual$counts)
```

```
     [,1] [,2] [,3] [,4] [,5] [,6]
[1,]    3    0    3    2    0    5
[2,]    5    0    3    4    0    3
[3,]   10    0    5    7    0    6
[4,]   14    0   16   15    0   11
[5,]    2    0    6    4    0    2
[6,]    8    0   12   14    0    6
```

The function can also be configured 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 span the boundary of a bin. However, it is more laborious and generates overlapping window pairs that are difficult to interpret. The gains in resolution are also minor given the sparsity of tag pairs throughout the genome-by-genome interaction space.

3.2.2 Effects 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 given that the length of the tag is only 20 bp.

The gap between two $3'$ ends in an outward-facing read pairs represents the interval spanned by the immunoprecipitated DNA fragment. This suggests that the location of the peak in the plot of Section 2.3.1 represents the average length of the fragments 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 the average length. Each reverse-extended tag 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 `2*ext` will have the same effect. This does not provide strand-specific counting but that level of detail is probably unnecessary (see discussions on sparsity below).

3.2.3 Choosing an appropriate width

The width of each bin is an important parameter 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. 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 data, 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-aware 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 to ensure that sufficiently large count sizes are obtained.

3.2.4 Filtering for computational efficiency

Only bin pairs with a sum of counts across all directories 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 tag pairs. Otherwise, the downstream analyses will be very computationally intensive for little practical gain. Conversely, the filter value should not be too high lest genuine interactions 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 directory. This is equivalent to identifying clusters 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 directories 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:

> Assume there are no self-ligation events. Any outward-facing tag pairs are generated from inter-ligation events and will be correlated with the other strand orientations. Peak calling on outward-facing homo-linker tag pairs will then select for interactions with large homo-linker counts (based on the other orientations). This compromises the independence between peak definition and statistical testing. In particular, interactions are likely to be selected that have spurious differences between homo- and hetero-linker counts.

In practice, this is not an issue as self-ligation events should dominate the outward-facing tag pairs. This means that only homo-linker directories need to be used for peak calling. Relatively few outward-facing tag pairs should be present for hetero-linker directories anyway.

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 taken from each fragment to mimic single-end data, given that most peak-callers cannot handle paired-end experiments.

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 was called with the command `macs14 -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 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]      *
  ---
  seqlengths:
                 chr1                chr10 ...                 chrX
                   NA                   NA ...                   NA
```

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.

```
> 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)
> peaks

GRanges with 6064 ranges and 0 metadata columns:
         seqnames                 ranges strand
            <Rle>              <IRanges>  <Rle>
     [1]     chr1     [ 851585,  857223]      *
     [2]     chr1     [ 900792,  902791]      *
     [3]     chr1     [ 934166,  936667]      *
     [4]     chr1     [ 954656,  956655]      *
     [5]     chr1     [1003968, 1006188]      *
     ...      ...                   ...     ...
  [6060]     chrX [153743460, 153745459]      *
  [6061]     chrX [153773862, 153776739]      *
  [6062]     chrX [153989883, 153991882]      *
  [6063]     chrX [154443769, 154445768]      *
  [6064]     chrX [154841444, 154843443]      *
  ---
  seqlengths:
                    chr1                chr10 ...                chrX
                      NA                   NA ...                  NA
```

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 binding sites. The counting can be performed using the `recountPET` function with filtering on the count sum as previously described. Reverse read extension can also be done but this is probably unnecessary.

```
> peaked <- recountPET(curdirs, peaks, filter=20L)
> head(peaked$counts)

     [,1] [,2] [,3] [,4] [,5] [,6]
[1,]   20    0   24   30    0   14
[2,]    7    0    9    6    0   10
[3,]    5    0    4   10    1    4
[4,]    5    0    5    8    0    3
[5,]   10    0   16   18    0   21
[6,]    9    0    7   11    0   16
```

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 binning or peak-based counts.

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

     SRR372741hom SRR372741het SRR372742hom SRR372742het
[1,]            6            0            7            0
```

```
[2,]            8            0            7            0
[3,]           15            0           13            0
[4,]           30            0           26            0
[5,]            8            0            6            0
[6,]           20            0           20            0
```

The total number of tag pairs in each library is also computed. Note that this refers to each library rather than each directory, such that each total includes tag pairs from all linker combinations. All homo-/hetero-linker counts from the same library are assigned the same total. This is appropriate as sequencing depth is the same for all counts sourced from the same library. Differences in homo-/hetero-linker counts within a library should not be attributed to differences in sequencing depth.

```
> freqs$totals

[1] 13177267 13177267 13330363 13330363
```

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     36490      5000 132000000       101 
```

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$counts <- freqs$counts[keep,]
> freqs$pairs <- freqs$pairs[keep,]
> sum(keep)

[1] 6329
```

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. 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-per-million (CPM) is generally recommended under the NB model.

```
> require(edgeR)
> a.cpm <- aveLogCPM(freqs$counts, lib.size=freqs$totals)
> summary(a.cpm)
```

```
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max. 
 -1.559  -1.480  -1.336  -1.160  -1.028   3.533 
```

The average count is useful as it is approximately independent of the p-value for any downstream tests [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 as the bulk of filtering has already been performed with `filter` during count loading.

```
> keep <- a.cpm > -1
> freqs$counts <- freqs$counts[keep,]
> freqs$pairs <- freqs$pairs[keep,]
> sum(keep)
```

```
[1] 1548
```

A more educated choice of filter threshold can be obtained by identifying putative genuine interactions as those that have say, a `X`-fold higher average abundance 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.

```
> X <- 50000
> binsize <- 1e6
> binned <-countPET(curdirs, width=binsize, filter=1)
> rebin <- compressMatrix(binned, hetero=is.het, libname=libname)
> threshold <- median(aveLogCPM(rebin$counts, lib.size=rebin$totals))
> size.adjust <- 2*log2(binsize/countwidth)
> threshold + log2(X) - size.adjust
```

```
[1] -1.821258
```

Note that large bins are required to count non-specific contacts due to the sparsity in the interaction space. This means that some adjustments for size are necessary to compare the average non-specific abundance to the average abundance of each bin pair of interest.

Chapter 4

Normalizing linker combinations

4.1 Motivation

Some normalization is necessary prior to comparisons between homo- and hetero-linker counts. This accounts for biases introduced by non-equal proportions of each linker as well as non-random ligation. Failure to normalize for these effects will result in 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, 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

Assume that ligation between DNA fragments is random. Each DNA fragment can be treated as a diploid individual where each end represents an allele. Random ligation 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 ratio, increasing the proportion of false positives.

This problem can be avoided with normalization. The value of a can be 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. 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 comparison.

4.2.2 Checking mitochondrial counts

The accuracy of the random ligation model can be assessed for each dataset by counting tag pairs mapped between the nuclear and mitochondrial chromosomes. The assumption here is that the mitochondrial genome does not interact with the 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(curdirs, FUN=function(x) { sum(extractMito(x)) })
> mitocounts

SRR372741_AA SRR372741_AB SRR372741_BB SRR372742_AA SRR372742_AB SRR372742_BB
        1754          735         1888         1848          783         1789
```

It is fairly obvious that these counts do not follow the predicted Hardy-Weinberg proportions. You can also go through the motions with a Pearson's chi-squared test to confirm this result. 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 SRR372741_AA SRR372741_AA
   0.2349270    0.4995314    0.2655416

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

        Chi-squared test for given probabilities

data:  lib.1
X-squared = 1928.863, df = 2, p-value < 2.2e-16
```

4.2.3 Checking counts across the interaction space

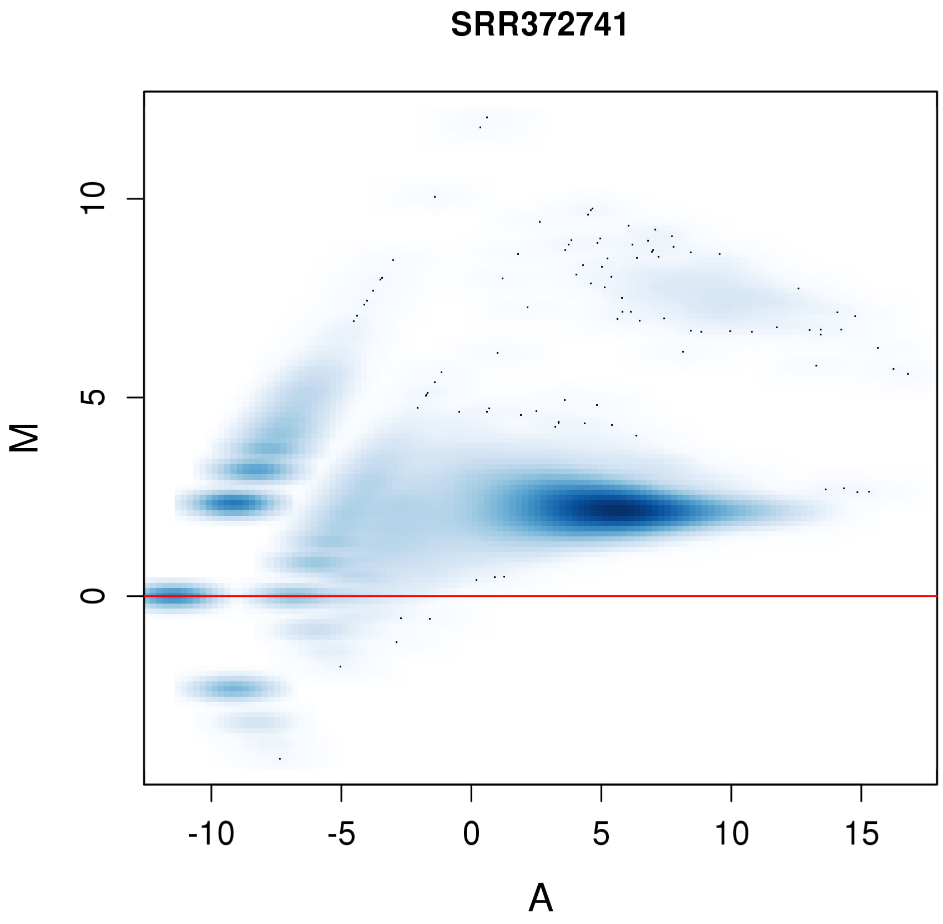
The mitochondrial results can be confirmed with counts from the rest of the dataset. Tag pairs are counted into pairs of 10 Mbp bins as described before. The assumption here is that most bin pairs contain only non-specific ligation events, i.e., most contacts in the interaction space are not specific. This is generally reasonable as contacts would not be expected between all pairs of binding sites. Again, large bins are necessary to count rare non-specific events.

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

     SRR372741hom SRR372741het SRR372742hom SRR372742het
[1,]         7979           25         8113           19
[2,]          364           31          345           49
[3,]         5677           20         5907           23
[4,]          234           45          244           39
[5,]          315           40          321           39
[6,]         7179           21         7161           32
```

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 a substantially higher M-value. This is consistent with the above-expected homo-linker counts for the nuclear-mitochondrial "interactions".

```
> adj.counts<-cpm(rebin$counts, lib.size=rebin$totals, 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=rebin$libname[1], cex.lab=1.4, cex.axis=1.2)
> abline(h=log2((expected[1]+expected[3])/expected[2]), col="red")
```



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. Different datasets may experience differing violations of the randomness assumption, so some diagnostics are recommended for quality control.

4.3 Normalizing for empirical effects

Normalization must account for non-equal linker proportions and non-random ligation. This can be done empirically by assuming that most interactions are non-specific. This means that the average M-value for most interactions can be used to define the expected homo-/hetero-linker ratio under the non-specific hypothesis. For example, if most interactions have an M-value close to 2, one would expect a homo-/hetero-linker ratio of around 4:1.

Normalization can then be performed to scale the counts according to the expected ratio. Specifically, the binned homo-/hetero-linker counts are passed into the trimmed mean of M-values (TMM) procedure [Robinson and Oshlack, 2010]. This returns a normalization factor for each set of counts in each library. Counts are then (conceptually) divided by the normalization factor, such that a direct comparison can be performed between the normalized counts. For example, an expected 4:1 ratio would involve dividing the homo-linker counts by 4 prior to comparison.

```
> inter.counts <- rebin$counts[is.na(getDistance(rebin)),]
> normfacs <- calcNormFactors(DGEList(inter.counts, lib.size=rebin$totals))$samples$norm.factors
> normfacs

[1] 2.1268066 0.4717666 2.1152203 0.4711825
```

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 ∼2 whereas only intra-chromosomal interactions make up the cloud at M-values of ∼7.

Chapter 5

Assessing biological variability

5.1 Motivation

Datasets with biological replicates account for biological variability when performing hypothesis testing. This reduces the significance of detected features when the data is highly variable. For count-based data, this can be achieved using the negative binomial (NB) model in the edgeR package [Robinson et al., 2010]. Estimation of the NB dispersion parameter allows you to model the variation between biological replicates. Similarly, estimation of the quasi-likelihood (QL) dispersion can be performed to account for heteroskedasticity [Lund et al., 2012].

Both types of dispersion estimation can be performed using methods in the edgeR package. This requires fitting of a generalized linear model (GLM) to the counts for each interaction [McCarthy et al., 2012]. To this end, the choice of 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 this paragraph. Firstly, it is necessary to assemble a `DGEList` object for entry into edgeR:

```
> y <- DGEList(freqs$counts, lib.size=freqs$totals, norm.factors=normfacs)
```

5.2 The paired-sample design

The most obvious choice is a paired-sample 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.

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

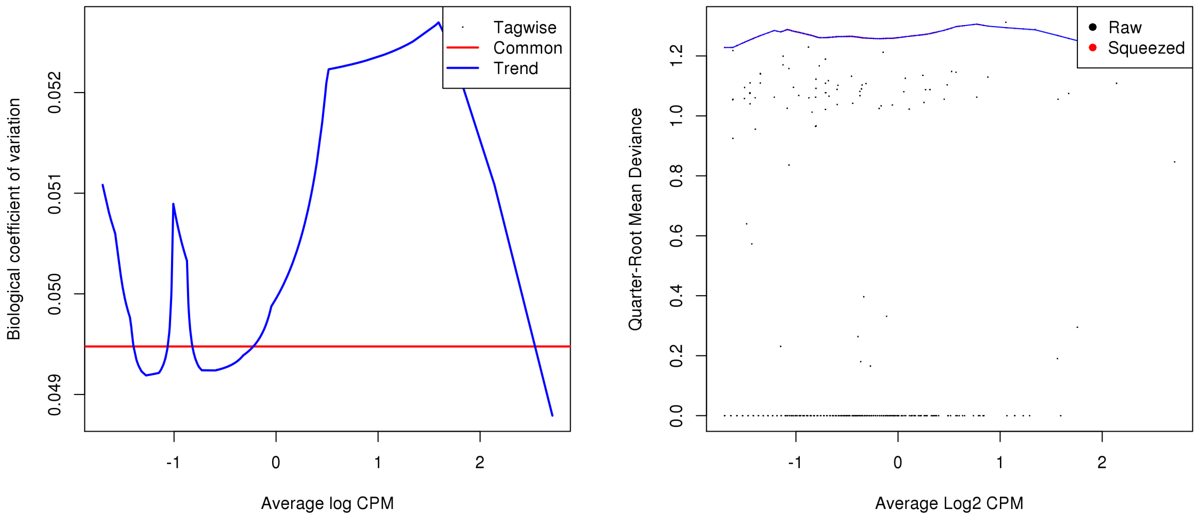
```
  (Intercept) factor(freqs$libname)SRR372742 factor(freqs$hetero)TRUE
1           1                               0                        0
2           1                               0                        1
3           1                               1                        0
4           1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

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

Estimation can then proceed using the methods in the edgeR package. The `estimateDisp` function will estimate the NB dispersion whereas the `glmQLFTest` 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)
> result.paired <- glmQLFTest(y.paired, design.paired, robust=TRUE, plot=TRUE)
```

It is worth pointing out that the biological coefficient of variation 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.



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 depend on the chosen 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

The preferred design for routine ChIA-PET analyses uses a one-way layout. 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.

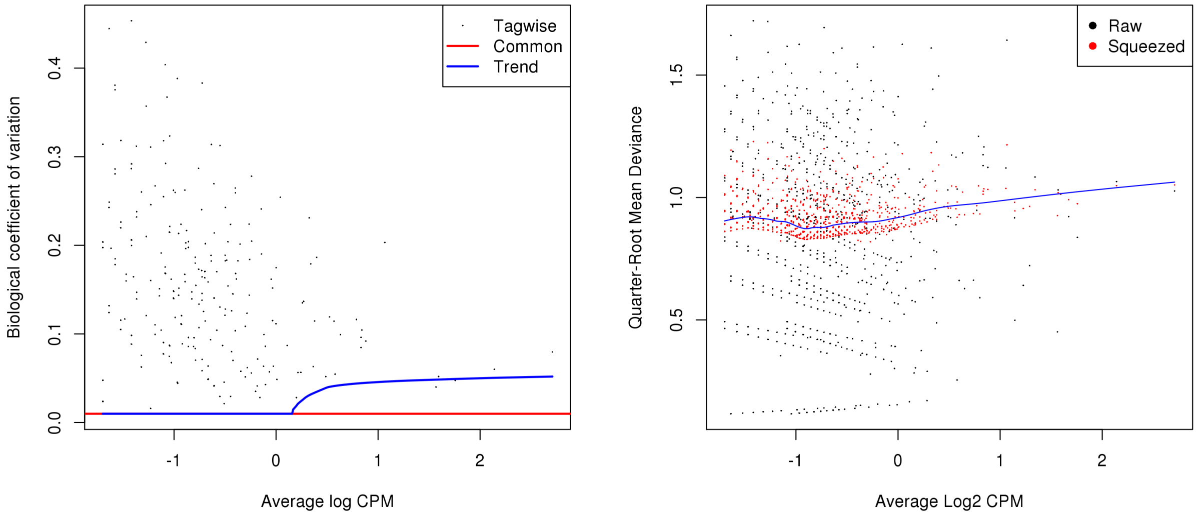
```
> design.group <- model.matrix(~factor(freqs$hetero))
> design.group
```

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

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

Estimation can then be performed as previously described. This approach avoids the zero-valued QL dispersions observed for the paired-sample design. Of course, the one-way layout assumes that there are no library-specific effects. For example, if the absolute count size 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.

```
> y.group <- estimateDisp(y, design.group)
> plotBCV(y.group)
> result.group <- glmQLFTest(y.group, design.group, robust=TRUE, plot=TRUE)
```



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 most other types of sequencing data (e.g., RNA-seq, ChIP-seq), NB dispersion estimates are usually around 0.01 to 0.1. Much lower values are observed here. This is a bit puzzling given that ChIA-PET should reflect the variability of ChIP-seq (and then some).

Chapter 6

Testing for significant interactions

6.1 Using the quasi-likelihood F-test

The glmQLFTest function also performs the F-test for each interaction so a new function call is not required. You should, however, make sure that you are specifying the contrast matrix correctly. 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(y.group, design.group, robust=TRUE, coef=2)
> topTags(result.group)

Coefficient:  factor(freqs$hetero)TRUE
         logFC   logCPM         F       PValue        FDR
701  -5.439309 2.709063 154.87391 1.328015e-05 0.01470323
606  -4.757355 1.561099 110.23357 3.167219e-05 0.01470323
1023 -6.295309 2.139953 110.49143 3.613280e-05 0.01470323
662  -4.982752 1.756849 106.61139 4.013834e-05 0.01470323
708  -5.836333 1.670704 100.67182 4.749105e-05 0.01470323
678  -5.718408 1.567376  82.51292 8.479853e-05 0.01790734
656  -9.347011 1.593222  95.68706 1.568422e-04 0.01790734
736  -8.999734 1.288642  93.77596 1.648607e-04 0.01790734
650  -8.929149 1.226528  92.69234 1.696615e-04 0.01790734
700  -4.503871 1.342437  58.91319 2.226420e-04 0.01790734
```

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 the null for each interaction. The log-fold change represents the relative increase of the hetero-linker count over the homo-linker count. In this case, they are negative for all interactions. This corresponds to an increase in the homo-linker count and is consistent with a specific interaction.

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

```
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max. 
 -9.347  -6.352  -5.928  -5.972  -5.710  -1.118 
```

You may also be wondering why we are not using the likelihood ratio test (LRT). Indeed, the LRT is the more obvious test for inferences with GLMs. However, the QL F-test is preferred 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 Benjamini-Hochberg method [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 positives in the entire dataset), as it provides a more appropriate compromise between specificity and power.

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

```
[1] 1504
```

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.

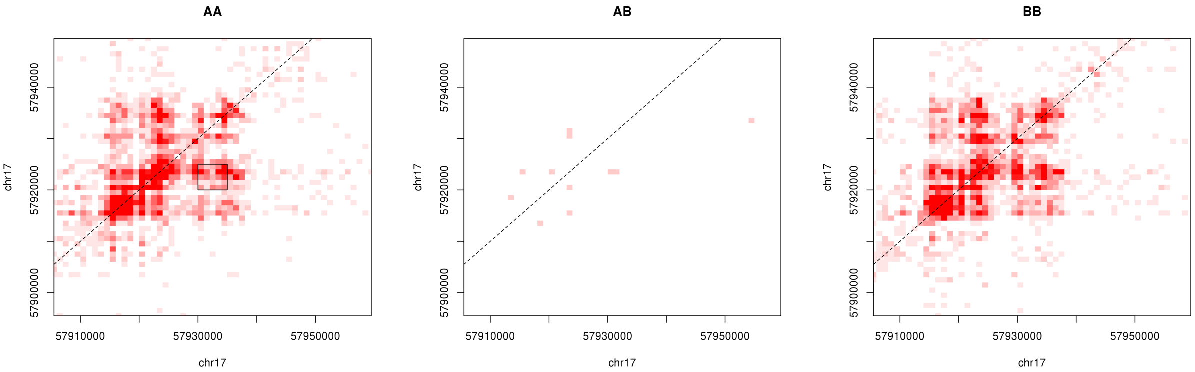
```
> ax <- freqs$region[freqs$pairs$anchor]
> tx <- freqs$region[freqs$pairs$target]
> out <- data.frame(anchor.chr=seqnames(ax), anchor.start=start(ax), anchor.end=end(ax),
+     target.chr=seqnames(tx), target.start=start(tx), target.end=end(tx),
+     result.group$table, FDR=adj.p)
> o <- order(out$PValue)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

6.3 Visualization with plaid plots

It comes a time that some visualization of interesting results is necessary. 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.

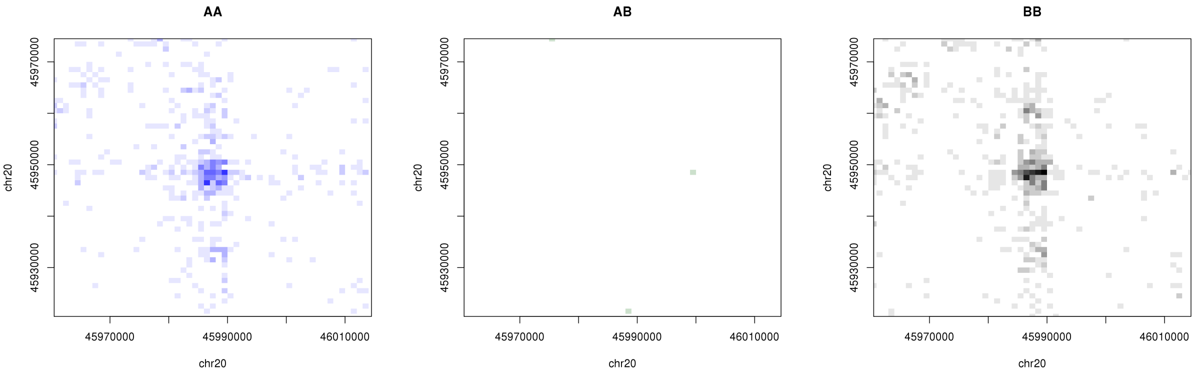
```
> expanded.a <- resize(ax[o[1]], fix="center", width=50000)
> expanded.t <- resize(tx[o[1]], fix="center", width=50000)
> plotChIA(curdirs[1], anchor=expanded.a, target=expanded.t, main="AA", cap=10)
> rect(start(ax[o[1]]), start(tx[o[1]]), end(ax[o[1]]), end(tx[o[1]]))
> abline(0, 1, lty=2)
> plotChIA(curdirs[2],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

Some expansion of the plot boundaries is often desirable. This ensures that the context of the interaction can be determined 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 AB colours 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(curdirs[1], anchor=expanded.a, target=expanded.t, main="AA", cap=10, col="blue")
> plotChIA(curdirs[2], anchor=expanded.a, target=expanded.t, main="AB", cap=5, col="darkgreen")
> plotChIA(curdirs[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, you should always keep in mind the protein that is being targeted. 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

References and other stuff

7.1 Session Information

```
> sessionInfo()

R version 3.1.0 (2014-04-10)
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   parallel  stats     graphics  grDevices utils     datasets
[8] methods   base

other attached packages:
 [1] statmod_1.4.20       locfit_1.5-9.1       edgeR_3.6.2
 [4] limma_3.20.6         dacpet_0.0.1         Rsamtools_1.16.1
 [7] Biostrings_2.32.0    XVector_0.4.0        GenomicRanges_1.16.3
[10] GenomeInfoDb_1.0.2   IRanges_1.22.9       BiocGenerics_0.10.0

loaded via a namespace (and not attached):
[1] bitops_1.0-6       grid_3.1.0         KernSmooth_2.23-12 lattice_0.20-29
[5] stats4_3.1.0       tools_3.1.0        zlibbioc_1.10.0
```

7.2 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.

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.