Exploration of HiC data

Felix Klein, John Marioni, Wolfgang Huber

May 3, 2012

Abstract

While we often think of the genome as a linear object, in reality chromosomes are folded in a highly complex fashion with regions that are located far apart on the same chromosome often coming in contact with one another. For example, many enhancer or insulator elements are located distal to the genes they target and come in contact via folding. Hence, understanding the conformation of chromatin is criticial for understanding regulatory mechanisms in a cell. With the advent of next-generation sequencing technology, we can now study chromosome conformation genome-wide using a technique called Hi-C. In brief, after cross-linking, digestion, ligation, and sonication, this technique captures interacting regions of DNA sequence by paired-end sequencing. For example, regions i and j might be two non-adjacent regions on the same chromosome in the fragment captured, sequence from region i is at one end and sequence from region j is at the other. Having captured these fragments, paired-end sequencing can then be used to assay a small section of sequence from region i and region j. By mapping these reads back to the reference genome, and by counting the number of reads with one end in region i and the other end in region j one can determine the strength of evidence for an interaction between those two regions.

In this practical, we will analyze one of the first Hi-C datasets generated [2]. We will understand how to obtain the data and read them into R, before processing these data to determine some measure of interaction frequency between different regions on a particular chromosome. Finally, we will combine the conformational information with data about other epigenetic modifications to investigate whether there appear to be any interactions between these different data types.

Contents

1 Introduction to the data

In this lab we will work with the data of the GM06990 cell line obtained from three experiments. Of these, two replicates used the restriction enzyme HindIII and one used NcoI. Altogether, these

experiments produced ca. 8.4 Mio mappable paired-end reads. Here, we only focus on the data of chromosome 14. How the data can be downloaded from the internet and formated into the dataframe `HiC_GM_chr14`, which is provided in the package, is described in Section 4 in detail. We follow the analysis of Lieberman-Aiden and pool the three experiments. This is justified by comparing the interaction matrices for each experiment. See Figures 1 B-D in the paper.

The format of `HiC_GM_chr14` is explained in the file `GSE18199_readme_v4.txt`, which we provide in the `extdata` directory of the package. Briefly, each line corresponds to one paired end alignment. Each line has 9 entries:

```
read name, chromosome1, position1, strand1, restrictionfragment1, chromosome2,
                    position2, strand2, restrictionfragment2
```

Of these, the four with last character "1" correspond to the first paired end, and the four with last character "2" to the second paired end. `position` is position in base pairs where the alignment starts. The alignments are based on the hg18 assembly of the human genome.

2 The adjacency matrix for chromosome 14

As first step, we load the data and have a look at the content of the data.frame. A first visual impression of the interactions is obtained by plotting the position of the first fragment against the position of the second fragment in a scatterplot (Figure 1).

```
> library(LiebermanAidenHiC2009)
> data("HiC_GM_chr14")
> head(HiC_GM_chr14)
```

```
            read name chromosome1 position1 strand1 restrictionfragment1
3112130  1:92:364:1569          14  28933774       0                  3226
3112131 1:20:1690:1830          14  18070705       0                     0
3112132  1:65:479:1658          14  18070922       0                     0
3112137    1:72:633:42          14  18071973       1                     1
3112145  1:97:353:2040          14  18072740       0                     1
3112148 1:14:1294:1281          14  18087996       0                     9
        chromosome2 position2 strand2 restrictionfragment2
3112130          14  18071056       0                    0
3112131          14  18071075       1                    0
3112132          14  18071183       1                    0
3112137          14  18071976       0                    1
3112145          14  18072977       1                    2
3112148          14  18074297       1                    2
```

```
> pos = with(HiC_GM_chr14, cbind(position1, position2))
```

```
> plot(pos, pch='.', col="#77777777")
```

To create a matrix of intrachromosomal locus–locus interaction frequencies, let us smooth the data using the `bkde2D` function of the *KernSmooth* package.

```
> chrlen = max(pos)
> gridsize = ceiling(chrlen/2e5)
> bandwidth = 3e5
> den = bkde2D( pos, bandwidth=c(1,1)*bandwidth, gridsize=c(1,1)*gridsize)
```

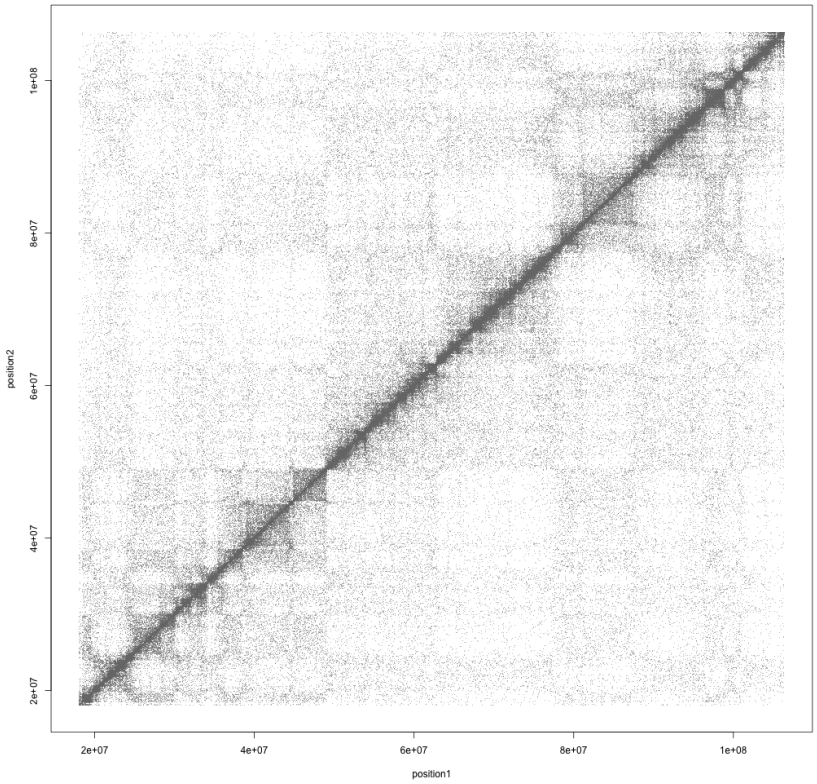

Figure 1: Scatterplot of fragment position 1 against 2 for the HiC read pairs for which both ends map to chromosome 14.

Since the the labeling of the reads within a pair as 1 or 2 is arbitrary, and pairwise interaction is a symmetric concept, we symmetrize the matrix by adding the transpose.

```
> den$fhat <- den$fhat + t(den$fhat)
```

We use the `image` function to visualise the interaction matrix in false color representation.

```
> with(den, image(x=x1, y=x2, z=fhat^0.3,
+       col=colorRampPalette(c("white","blue"))(256), useRaster=TRUE))
```

The result is shown in Figure 2.

- What is the point of the exponentiation ($\wedge 0.3$)? What happens if you do not do it, or use another transformation?

The diagonal is strongly pronounced because the contact frequency is highest for small genomic distances. To normalize for this, and focus on the (possibly interesting) deviations from this general relationship, we calculate the mean number of interactions at a given genomic distance. First we do this for the secondary diagonals and then add the diagonal. The resulting matrix m is plotted in Figure 3.

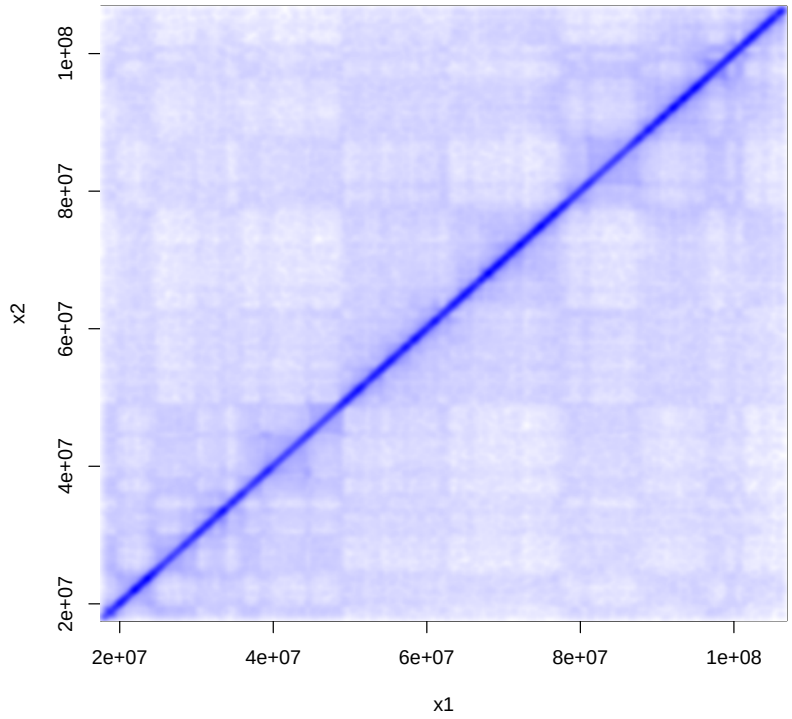

Figure 2: Interaction matrix obtained from smoothing the positions of paired read alignments. Compare this to Figure 3 A in the paper [2]. Can you explain why the diagonal looks more pronounced here than in the paper, and more off diagonal structures can be seen in the paper. Hint: Try to set a threshold for the maximum value of the data.

```
> m = matrix(0, nrow=gridsize, ncol=gridsize)
> for(i in 1:(gridsize-1)) {
+   band = (row(m)==col(m)+i)
+   m[band] = mean(den$fhat[band])
+ }
> m = m + t(m)
> diag(m) = mean(diag(den$fhat))

> image(x=den$x1, y=den$x2, z=m^0.3,
+       col=colorRampPalette(c("white","blue"))(256), useRaster=TRUE)
```

Plotting the avarage number of interactions against the genomic distance is a better way of visualizing the data. It is achieved by plotting the first row (or column as the matrix is symmetric) against genomic distance and shown in Figure 4

```
> genomicDistance = den$x1 - min(den$x1)
> averageInteractions = m[1,]
```

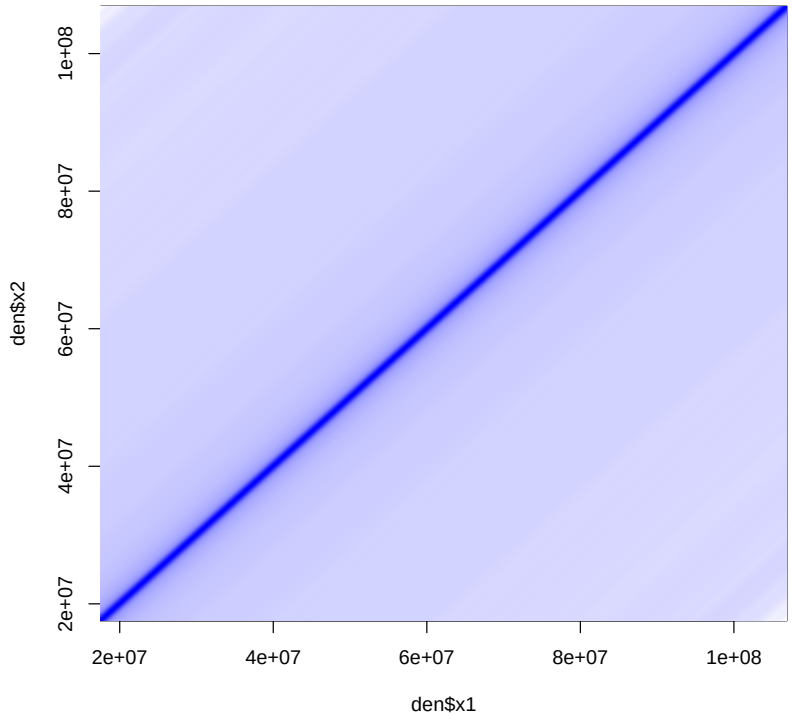

Figure 3: The matrix of avarage interactions `m`. It looks like a ridge that falls of on both sides of the diagonal.

```
> plot(genomicDistance, sqrt(averageInteractions), type ="l")
```

- How can the steep drop be explained?
- What could be an explanation for the bump at the right end of the curve?
- Why might it make sense to plot the square root of average interactions against genomic distance between to loci? Hint: Think about where points lie in three dimensions at a given distance r from a point of origin p.

With the calculated matrix of average interactions we can normalize our data by dividing the two matrices and plot the result.

```
> fhatNorm <- den$fhat/m
```

```
> image(x=den$x1, y=den$x2, z=fhatNorm,
+       col=colorRampPalette(c("white","blue"))(256), useRaster=TRUE)
```

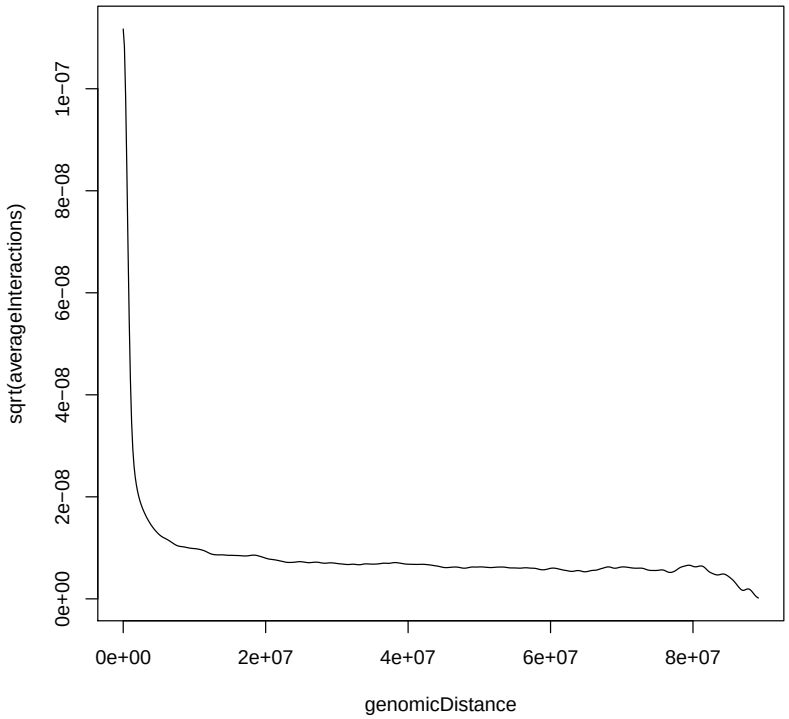

Figure 4: Average number of interactions within chromosome 14 ploted against the genomic distance between the fragments.

By normalizing for one, fairly obvious factor, genomic distance, we have removed uninteressting structure from the raw data, and obtained a possibly more interesting picture in Figure 5. We can see that the plaid pattern is more pronounced than before.

One could think of other factors that might influence the results, like GC content or restriction enzyme biases, and try if a correction for these is worthwhile.

Another view on the data is provided by caluclating the correlation matrix, where the entry (i, j) is the correlation of the of the row i with row j. Plotting the correlation matrix and using a third color in the plot, we can see that the chromosome basically splits into 2 compartments (blue and red) shown in Figure 6.

```
> cm <- cor(fhatNorm)
```

```
> image(x=den$x1, y=den$x2, z=cm,
+       zlim=c(-1,1),
+       col=colorRampPalette(c("red", "white","blue"))(256), useRaster=TRUE)
```

As next step we perform a principal component analysis on the matrix and retrieve the first two eigenvectors.

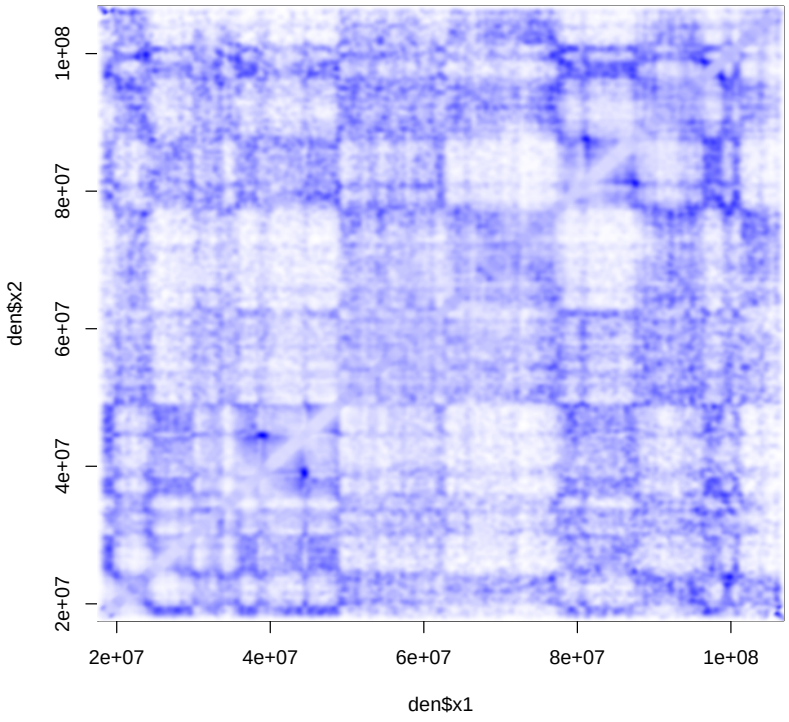

Figure 5: Interaction matrix after devision by the expected number of read pairs.

Because we want to compare these vectors to ChIP-seq experiments of histone modifications and DNAse1 sensitivity, we use run-length encoding (*Rle*) to store the vectors. The *Rle* class of the IRanges package provides an efficient way of storing vectors that have long constant runs.

```
> princp = princomp(cm)
```

```
> plot(den$x1, princp$loadings[,1], type="l")
```

See Figure 7.

```
> pc1Vec = Rle(values  = princp$loadings[,1],
+              lengths = c(den$x1[1], diff(den$x1)))
> pc2Vec = Rle(values  = princp$loadings[,2],
+              lengths = c(den$x1[1], diff(den$x1)))
```

3 Comparison with chromatin state data

We would like to explore the pattern we found in the previous section in the context of chromatin state data. For this we use data that were produced in the Encode project [1]. How these data were obtainted and formated is described in Section 4.

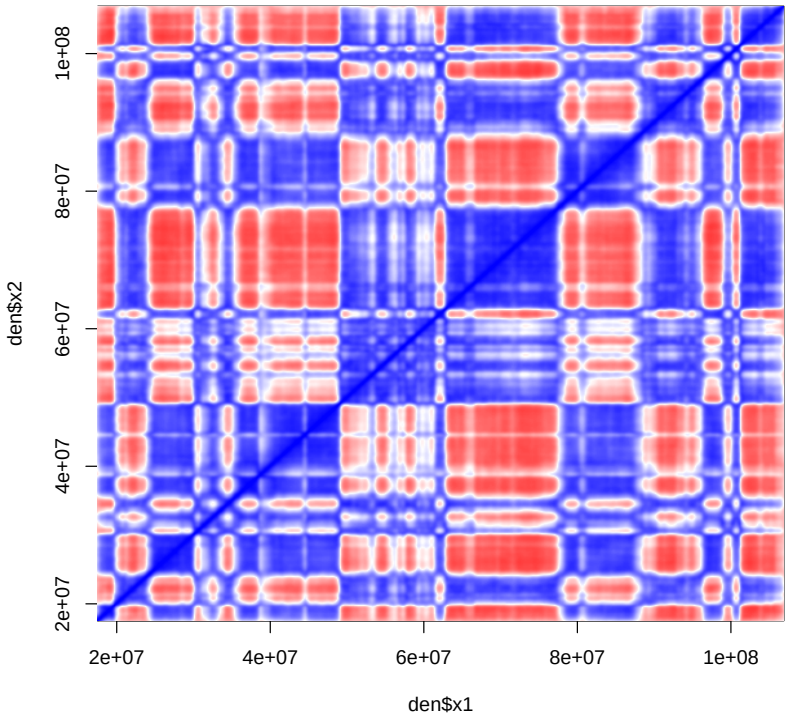

Figure 6: Correlation matrix calculated from the normalized interaction matrix.

The data files contain the the following information about ChIP-seq peaks found for chromosome 14:

`chromosome, start position, end position, name, score, strand, signal value, pValue, qValue`

For each peak, we want the start position, end position and the signal value in this range. To create a Rle vector from the data, we use the following function.

```
> createRleVector = function(tab){
+   RleVec = Rle(0, max(tab$end))
+   for(i in 1:nrow(tab)){
+     RleVec = RleVec +
+       Rle(values  = c(0,              tab$signalValue[i],         0),
+           lengths = c(tab$start[i]-1,
+                       tab$end[i]-tab$start[i]+1,
+                       length(RleVec)-tab$end[i]))
+   }
+   RleVec
+ }
```

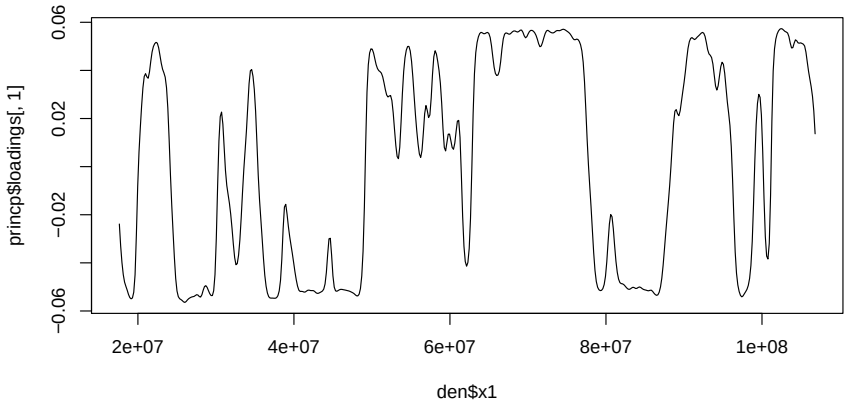

Figure 7: `princp$loadings[,1]`.

The Chip-seq data can be loaded from the package and the function can be applied directly.

```
> data("ChipSeqData")
> H3K27me3 = createRleVector(H3K27me3.df)
> H3K36me3 = createRleVector(H3K36me3.df)
> DNAse1   = createRleVector(DNAse1.df)
> DNAse2   = createRleVector(DNAse2.df)
```

To combine the two DNAse1 replicates we have to make sure that the vectors have the same length.

```
> length(DNAse1)

[1] 106358944

> length(DNAse2)

[1] 106359236

> DNAse = DNAse1 + DNAse2[seq(along=DNAse1)]
```

For plotting the first eigenvector against the ChIP-seq data, we have to make sure that we use the same scale for all plots. Therefore we define a plotting function that covers the same genomic range by setting a fixed x-axis range `xlim`. We also want to plot them on top of each other in one plot and do this by setting `par` accordingly. The result is shown in Figure 8

```
> plotRle = function(RleVector, ...){
+   plot(end(RleVector), runValue(RleVector)+1, type="h", log="y",
+   xlim = c(1.5e+7, 107000000), xlab="", ylab=deparse(substitute(RleVector)),
+   ...)
+ }
>
```

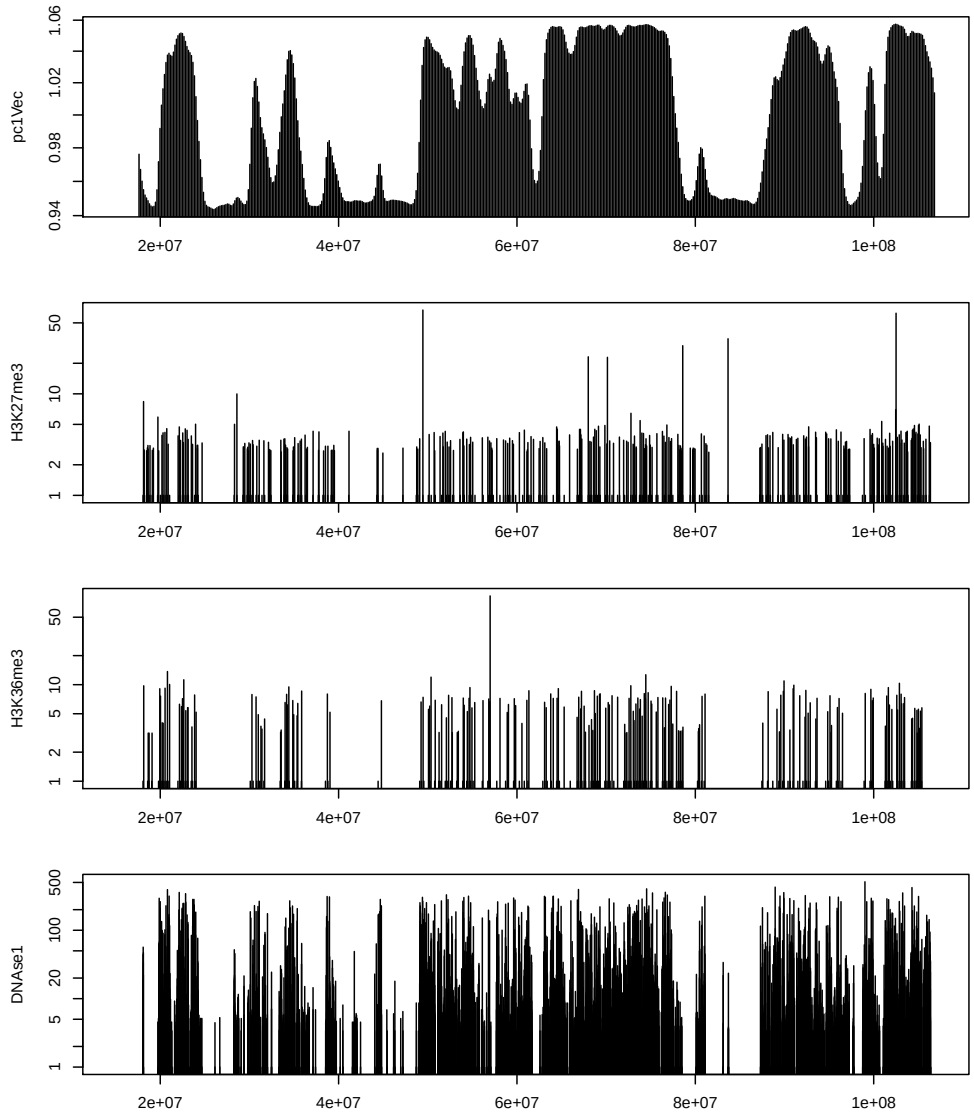

Figure 8: Plots of `pc1Vec`, `H3K27me3`, `H3K36me3`, `DNAse1`.

```
> par(mfrow=c(4,1), mai=c(0.5,0.7,0.1,0.1))
> plotRle(pc1Vec)
> plotRle(H3K27me3)
> plotRle(H3K36me3)
> plotRle(DNAse1)
```

From looking at the plots the data seem to correlate quite well. To assess this more quantitatively we calculate the correlation of the vectors. To do this we need to make sure that they all are of same length.

```
> c(length(H3K27me3),
+   length(H3K36me3),
+   length(DNAse),
+   length(pc1Vec))

[1] 106360575 105401700 106358944 106810060

> x = seq(along=H3K36me3)
> cor(H3K27me3[x], pc1Vec[x])

[1] 0.3783589

> cor(H3K36me3, pc1Vec[x])

[1] 0.4767175

> cor(DNAse[x], pc1Vec[x])

[1] 0.08286722
```

As you see, the correlation coefficents are less convincing than the visual impression, especially for DNAse1.

- Discuss this topic. Repeat the last steps for the second eigenvector of the principal component analysis.
- Which other analyses or annotations could be done now to put the spacial proximity into a functional context?

4 Obtaining the data

The data associated with the article [2] is available from NCBI Gene Expression Omnibus under the accession GSE18199. From this record, we downloaded the supplementary file GSE18199_RAW.tar and extracted its content.

```
$ wget ftp://ftp.ncbi.nih.gov/pub/geo/DATA/supplementary/series/
GSE18199/GSE18199_RAW.tar
$ tar -xvf GSE18199_RAW.tar
```

From the resulting set of files, we selected six files, whose names are given in the following code chunk. They contain HiC data for the GM06990 cell line from three experiments (two replicates with the restriction enzyme HindIII and one with NcoI). The sample information is summarized in the following table.

```
> sampleAnnotation =
+   data.frame(
+     file = c("GSM455133_30E0LAAXX.1.maq.hic.summary.binned.txt.gz",
+              "GSM455134_30E0LAAXX.2.maq.hic.summary.binned.txt.gz",
+              "GSM455135_30U85AAXX.2.maq.hic.summary.binned.txt.gz",
+              "GSM455136_30U85AAXX.3.maq.hic.summary.binned.txt.gz",
+              "GSM455137_30305AAXX.1.maq.hic.summary.binned.txt.gz",
+              "GSM455138_30305AAXX.2.maq.hic.summary.binned.txt.gz"),
```

```
+     experiment = c(1, 1, 2, 2, 3, 3),
+     lane = c(1,2,1,2,1,2),
+     restrictionenzyme = c("HindIII", "HindIII", "HindIII", "HindIII",
+                           "NcoI", "NcoI"),
+     stringsAsFactors = FALSE)
```

The format of these files is explained in the file `GSE18199_readme_v4.txt`, which we provide in the `extdata` directory of the package or can be downloaded from NCBI Gene Expression Omnibus.

To create the dataframe `HiC_GM_chr14` for this practical we used the following code and follow Lieberman-Aiden in pooling the three experiments of the GM06990 cell line. This was justified by comparing the interaction matrices of individual experiment with each other. See Figures 1 B-D in the paper.

```
> inputDir = "WHERE YOU SAVED THE FILES"
> outputDir = "WHERE YOU WANT TO SAVE THE FILES"
> df = vector(mode="list", length=nrow(sampleAnnotation))
> for(i in seq(along=df)) {
+   cat("Reading file", i, "\n")
+   r = read.table(gzfile(file.path(inputDir, sampleAnnotation$file[i])),
+            header=FALSE, sep="\t", comment.char = "", stringsAsFactors=FALSE)
+   colnames(r) = c("read name",
+      "chromosome1", "position1", "strand1", "restrictionfragment1",
+      "chromosome2", "position2", "strand2", "restrictionfragment2")
+   ## filter chromosome 14
+   df[[i]] = subset(r, (chromosome1==14L) & (chromosome2==14L))
+ }
> HiC_GM_chr14 = do.call(rbind, df)
> save(HiC_GM_chr14,
+      file=file.path(outputDir, "HiC_GM_chr14.RData"))
```

For comparison with the cromatin state we use Chip-seq data produced within the ENCODE project [1]. It was downloaded from these URLs.

```
$ wget http://hgdownload.cse.ucsc.edu/goldenPath/
hg18/encodeDCC/wgEncodeBroadHistone/
wgEncodeBroadHistoneGm12878H3k27me3StdPk.broadPeak.gz
$ wget http://hgdownload.cse.ucsc.edu/goldenPath/
hg18/encodeDCC/wgEncodeBroadHistone/
wgEncodeBroadHistoneGm12878H3k36me3StdPk.broadPeak.gz
$ wget http://hgdownload.cse.ucsc.edu/goldenPath/
hg18/encodeDCC/wgEncodeUwDnaseSeq/
wgEncodeUwDnaseSeqHotspotsRep1Gm06990.broadPeak.gz
$ wget http://hgdownload.cse.ucsc.edu/goldenPath/
hg18/encodeDCC/wgEncodeUwDnaseSeq/
wgEncodeUwDnaseSeqHotspotsRep2Gm06990.broadPeak.gz
```

These files are read into data.frames and saved.

```
> files = c("wgEncodeBroadChipSeqPeaksGm12878H3k27me3.broadPeak.gz",
+     "wgEncodeBroadChipSeqPeaksGm12878H3k36me3.broadPeak.gz",
+     "wgEncodeUwDnaseSeqHotspotsRep1Gm06990.broadPeak.gz",
+     "wgEncodeUwDnaseSeqHotspotsRep2Gm06990.broadPeak.gz")
```

```
> inputDir = "WHERE YOU SAVED THE FILES"
> outputDir = "WHERE YOU WANT TO SAVE THE FILES"
> cs = vector(mode="list", length=length(files))
> for(i in seq(along=files)) {
+   cat("Reading file", i, "\n")
+   tab = read.table(gzfile(file.path(inputDir, files[i])),   header=FALSE, sep="\t", comment.cha
+   colnames(tab) <- c("chr", "start", "end", "name", "score", "strand",
+                      "signalValue", "pValue", "qValue")
+   cs[[i]] = subset(tab, chr=="chr14")
+ }
> H3K27me3.df = cs[[1]]
> H3K36me3.df = cs[[2]]
> DNAse1.df = cs[[3]]
> DNAse2.df = cs[[4]]
> save(list=c("H3K27me3.df", "H3K36me3.df", "DNAse1.df", "DNAse2.df"), file=file.path(outputDir,
```

5 SessionInfo

```
> toLatex(sessionInfo())
```

- R version 2.15.0 (2012-03-30), `i386-apple-darwin9.8.0`
- Locale: `C`
- Base packages: base, datasets, grDevices, graphics, methods, stats, utils
- Other packages: BiocGenerics 0.2.0, IRanges 1.14.2, KernSmooth 2.23-7, LiebermanAidenHiC2009 0.1.12
- Loaded via a namespace (and not attached): stats4 2.15.0, tools 2.15.0

The output of `sessionInfo` on the build system after running this vignette.

References

[1] The ENCODE Project Consortium. The ENCODE (ENCylopedia Of DNA Elements) Project. *Science*, 306:636–640, 2004.

[2] 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*, 326:289–293, 2009.