

# GENOVA: explore the Hi-C's

***Robin H. van der Weide<sup>1</sup>, Teun van den Brand<sup>1</sup>, and Elzo de Wit<sup>\*1</sup>***

<sup>1</sup>Division of Gene Regulation, the Netherlands Cancer Institute

<sup>\*</sup>e.d.wit@nki.nl

**12 February 2020**

## Abstract

The increase in interest for Hi-C methods in the chromatin community has led to a need for more user-friendly and powerful analysis methods. The few currently available software packages for Hi-C do not allow a researcher to quickly summarize and visualize their data. An easy to use software package, which can generate a comprehensive set of publication-quality plots, would allow researchers to swiftly go from raw Hi-C data to interpretable results. Here, we present **GENome ORganisation Visual Analytics** (GENOVA): a software suite to perform in-depth analyses on various levels of genome organisation, using Hi-C data. GENOVA facilitates the comparison between multiple datasets and supports the majority of mapping-pipelines.

## Contents

1	Loading data . . . . .	3
1.1	Data structures of input . . . . .	3
1.2	Recommended resolutions . . . . .	3
1.3	construct.experiment . . . . .	4
1.4	Juicebox & cooler . . . . .	6
2	Genome-wide analyses . . . . .	6
2.1	<i>Cis</i> -quantification . . . . .	6
2.2	chromosome plots . . . . .	7
2.3	RCP . . . . .	8
2.4	A- and B-compartments . . . . .	11
3	Interaction plots . . . . .	13
3.1	<i>cis</i> -interactions . . . . .	14
3.2	<i>trans</i> -interactions . . . . .	15
3.3	matrix plots . . . . .	16
4	TADs . . . . .	23
4.1	Insulation . . . . .	23
4.2	Call TADs . . . . .	26
4.3	ATA . . . . .	27
4.4	TAD+N . . . . .	28
5	Loops . . . . .	31
5.1	APA . . . . .	31
6	Far- <i>cis</i> interactions . . . . .	34
6.1	PE-SCAn . . . . .	34
6.2	centromere.telomere.analysis . . . . .	35
7	Session info . . . . .	36
	References . . . . .	37

# 1 Loading data

```
# devtools::install_github("robinweide/GENOVA", ref = 'dev')
library(GENOVA)
```

## 1.1 Data structures of input

GENOVA expects two input files: the signal- and the index-file. Signal-files have three columns (bin1, bin2, contactCount) and index-files have four (chromosome, start, end, bin). These are the default output of the Hi-C mapping pipeline HiC-Pro (Servant et al. 2015), where they are called \*.matrix and \*.bed. The files are expected to be genome-wide and may be corrected with ICE-normalisation.

## 1.2 Recommended resolutions

To ensure computational strain and time is kept to a minimum, we recommend different resolutions for different functions (table 1). More experienced users are free to deviate, while keeping in mind that these datasets are quite memory-heavy (table 2).

**Table 1: Recommended resolutions**

These will provide optimal resource/result tradeoffs.

Function	Resolution
APA	10kb-20kb
ATA	10kb-40kb
cisTotal.perChrom	500kb-1Mb
chromosomeMatrix	500kb-1Mb
RCP	40kb-500kb
intra.inter.TAD.contacts	20kb - 40kb
PE-SCAn	20kb-40kb
hic.matrixplot	<i>width in bp of window</i> 500
centromere.telomere.analysis	40kb
.compartment.plot	100kb

**Table 2: Memory footprints of objects loaded into R**

Experiment	Contacts	10kb	40kb	100kb	1Mb
Hap1 (Haarhuis et al. 2017)	433.5M	2.9Gb	1.7Gb	1.1Gb	0.1Gb
iPSC (Krijger et al. 2016)	427.9M	3.1Gb	1.9Gb	1.0Gb	53.1MB

Several functions rely on centromere-information. You can add this in the form of a BED-like three-column data.frame when constructing the experiment-object.<sup>1</sup> If not present, the centromeres will be empirically identified by searching for the largest stretch of no coverage on a chromosome.

<sup>1</sup>Please make sure that the chromosome-names match.

## GENOVA: explore the Hi-C's

```
centromeres = read.delim('../data/hg19_cytobandAcen.bed',
                        sep = '\t',
                        h = F,
                        stringsAsFactors = F)
head(centromeres)
#>      V1      V2      V3
#> 1 chr1 121500000 128900000
#> 2 chr10 38000000 42300000
#> 3 chr11 51600000 55700000
#> 4 chr12 33300000 38200000
#> 5 chr13 16300000 19500000
#> 6 chr14 16100000 19100000
```

### 1.3 construct.experiment

Every Hi-C experiment will be stored in an experiment-object. This is done by invoking the `construct.experiment` function. Inside, several sanity checks will be performed, data is normalised to the total number of reads and scaled to a billion reads (the default value of the `BPsclaling`-option). For this example, we are going to use the Hi-C maps of WT and  $\Delta$ WAPL Hap1 cells from Haarhuis et al. (2017). Since the genome-wide analyses do not need very high-resolution data, we will construct both 10kb, 40kb and 1Mb resolution experiment-objects.

```
Hap1_WT_10kb <- load_contacts(signal_path = '../data/symlinks/WT_10000_iced.matrix',
                                 indices_path = '../data/symlinks/WT_10000_abs.bed',
                                 sample_name = "WT",
                                 colour = "black")
#> Reading data...

Hap1_WAPL_10kb <- load_contacts(signal_path = '../data/symlinks/WAPL_10000_iced.matrix',
                                   indices_path = '../data/symlinks/WAPL_10000_abs.bed',
                                   sample_name = "WAPL",
                                   colour = "red")
#> Reading data...

Hap1_SCC4_10kb <- load_contacts(signal_path = '../data/symlinks/SCC4_10000_iced.matrix',
                                   indices_path = '../data/symlinks/SCC4_10000_abs.bed',
                                   sample_name = "SCC4",
                                   colour = "green")
#> Reading data...

Hap1_WT_40kb <- load_contacts(signal_path = '../data/symlinks/WT_40000_iced.matrix',
                                 indices_path = '../data/symlinks/WT_40000_abs.bed',
                                 sample_name = "WT",
                                 colour = "black")
#> Reading data...

Hap1_WAPL_40kb <- load_contacts(signal_path = '../data/symlinks/WAPL_40000_iced.matrix',
                                   indices_path = '../data/symlinks/WAPL_40000_abs.bed',
```

## GENOVA: explore the Hi-C's

```
sample_name = "WAPL",
colour = "red")

#> Reading data...

Hapl_WT_1MB <- load_contacts(signal_path = '../data/symlinks/WT_1000000_iced.matrix',
                               indices_path = '../data/symlinks/WT_1000000_abs.bed',
                               sample_name = "WT", centromeres = centromeres,
                               colour = "black")

#> Reading data...

Hapl_WAPL_1MB <- load_contacts(signal_path = '../data/symlinks/WAPL_1000000_iced.matrix',
                                 indices_path = '../data/symlinks/WAPL_1000000_abs.bed',
                                 sample_name = "WAPL",
                                 centromeres = centromeres,
                                 colour = "red")

#> Reading data...
```

The resulting `contacts`-object has several slots. `MAT` and `IDX` are the signal- and index-data.tables. We also have slots for the included chromosomes (`CHRS`) and the given centromers (`CENTROMERES`).

```
#> List of 4
#> $ MAT :Classes 'data.table' and 'data.frame': 179689855
#>   obs. of 3 variables:
#>   ..$ V1: int [1:179689855] NULL ...
#>   ..$ V2: int [1:179689855] NULL ...
#>   ..$ V3: num [1:179689855] NULL ...
#>   ... - attr(*, ".internal.selfref")=<externalptr>
#>   ... - attr(*, "sorted")= chr [1:2] ...
#> $ IDX :Classes 'data.table' and 'data.frame': 309581 obs.
#>   of 4 variables:
#>   ..$ V1: chr [1:309581] ...
#>   ..$ V2: int [1:309581] NULL ...
#>   ..$ V3: int [1:309581] NULL ...
#>   ..$ V4: int [1:309581] NULL ...
#>   ... - attr(*, ".internal.selfref")=<externalptr>
#>   ... - attr(*, "sorted")= chr [1:2] ...
#> $ CHRS : chr [1:24] ...
#> $ CENTROMERES:Classes 'data.table' and 'data.frame': 24
#>   obs. of 3 variables:
#>   ..$ chrom: chr [1:24] ...
#>   ..$ start: int [1:24] NULL ...
#>   ..$ end : int [1:24] NULL ...
#>   ... - attr(*, ".internal.selfref")=<externalptr>
#>   ... - attr(*, "sorted")= chr ...
#>   - attr(*, "class")= chr ...
#>   - attr(*, "znorm")= logi NULL ...
#>   - attr(*, "samplename")= chr ...
#>   - attr(*, "colour")= chr ...
#>   - attr(*, "resolution")= num NULL ...
#>   - attr(*, "rmChrom")= logi NULL ...
```

## GENOVA: explore the Hi-C's

```
#> - attr(*, "balanced")= logi NULL ...
#> - attr(*, "scale_cis")= logi NULL ...
#> - attr(*, "package")= chr ...
```

Finally, the object has a lot of specific attributes, like metadata and given parameters during loading. The amount of contacts in the *ICE* data.table is likely different from the input-data, because it is scaled to a fixed number of reads (which can be set with the `scale_bp`-option in `load_contacts()`).

```
#>      class znorm samplename colour resolution rmChrom balanced scale_cis
#> 1 contacts FALSE          WT  black     10000    TRUE    TRUE    FALSE
#>   package
#> 1 GENOVA
```

## 1.4 Juicebox & cooler

Both `.hic` and `.cooler` files can be loaded from version 1 onwards. The same `load_contacts()` function can be used, which will automatically determine the file-type based on the extention.

```
Hap1_WT_10kb_juicer <- load_contacts(signal_path = '../data/symlinks/WT.hic',
                                         sample_name = "WT",
                                         resolution = 10e3,
                                         balancing = 'KR', # this is the default
                                         colour = "black")

Hap1_WT_10kb_cooler <- load_contacts(signal_path = '../data/symlinks/WT.cooler',
                                         sample_name = "WT",
                                         balancing = T,
                                         colour = "black")
```

## 2 Genome-wide analyses

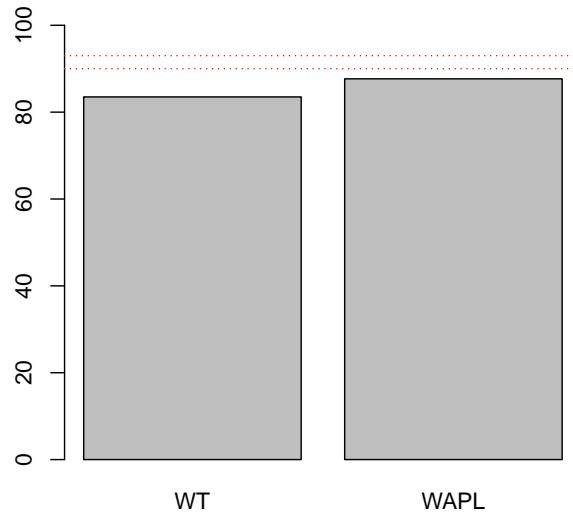
A good place to start your analyses are some functions on a genome-wide level. We can assess the quality of the library, identify translocations and generate contact probability (aka scaling or interaction decay plots).

### 2.1 Cis-quantification

Work by the group of Amos Tanay showed that the expected amount of intra-chromosomal contacts is the range of 90 to 93 percent (Olivares-Chauvet et al. 2016). Assuming that any extra inter-chromosomal contacts are due to debris/noise, the user might aspire to get the *cis*-percentages as close to 90% as possible. To compute the percentage of per-sample *cis*-contacts, we simply provide `cis_trans()` with the exp-object of interest. It will produce a boxplot of the percentages *cis* per sample (figure 1).

## GENOVA: explore the Hi-C's

```
cisChrom_out <- cis_trans( list(Hap1_WT_1MB, Hap1_WAPL_1MB) )
barplot(cisChrom_out$cis, names.arg = cisChrom_out$sample, ylim = c(0, 100) )
abline(h = 90, col = 'red', lty = 3)
abline(h = 93, col = 'red', lty = 3)
```



**Figure 1: Fraction of cis-contacts per sample**

Red dotted line denote the percentage-range from Olivares et al..

The same function can also be ran on specific regions. For this example, we will compute the intra-arm percentages. Plotting this shows us that there are interesting differences between the amounts of intra-arm contacts in *cis*, which is can be attributed to the loss of TADs (Haarhuis et al. 2017) (figure 2).

```
p_arms <- data.frame('chromosome' = centromeres[,1],
                      'start' = 0,
                      'end' = centromeres[,2])
cisChrom_out <- cis_trans( list(Hap1_WT_10kb, Hap1_SCC4_10kb) , bed = p_arms)
barplot(cisChrom_out$cis, names.arg = cisChrom_out$sample )
abline(h = 90, col = 'red', lty = 3)
abline(h = 93, col = 'red', lty = 3)
```

## 2.2 chromosome plots

Hi-C has been shown to be a powerful data-source to detect chromosomal rearrangements (Harewood et al. 2017). To find possible translocations, we can plot the genome-wide enrichment of interactions between all combinations of chromosomes. The values in the matrix are  $\log_{10}(\text{observed}/\text{expected})$ . The Hap1 cell line has two known translocations, which we can easily see in the resulting plot (figure 3). To narrow-in on this location, you could use the `trans.compartment.plot`-function (discussed below).

```
# Lets remove mitochondrial and Y-chromosomal contacts
chromosomeMatrix(Hap1_WT_1MB, remove = c("chrM", "chrY"))
```

## GENOVA: explore the Hi-C's

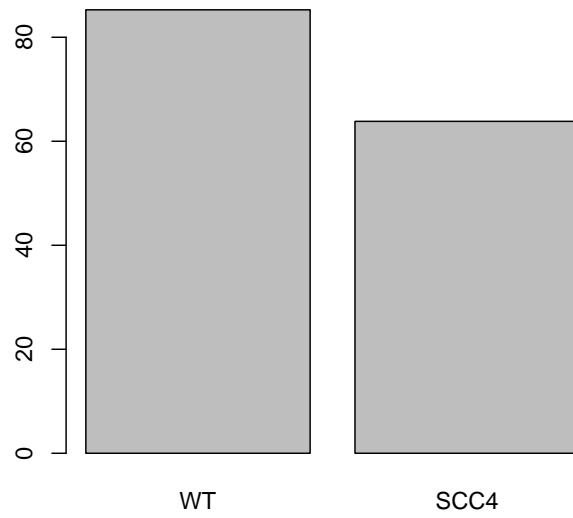


Figure 2: Fraction of cis-contacts per sample at the p-arms

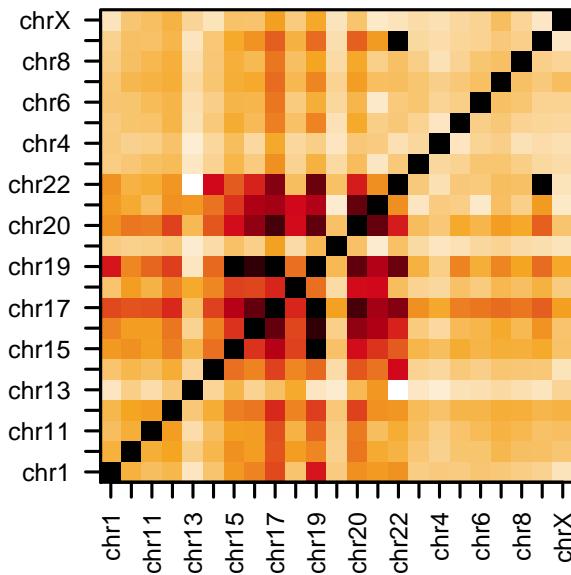


Figure 3: Chromosome matrix

The two known translocations of Hap1 cells are easily identified (15-19 & 9-22).

## 2.3 RCP

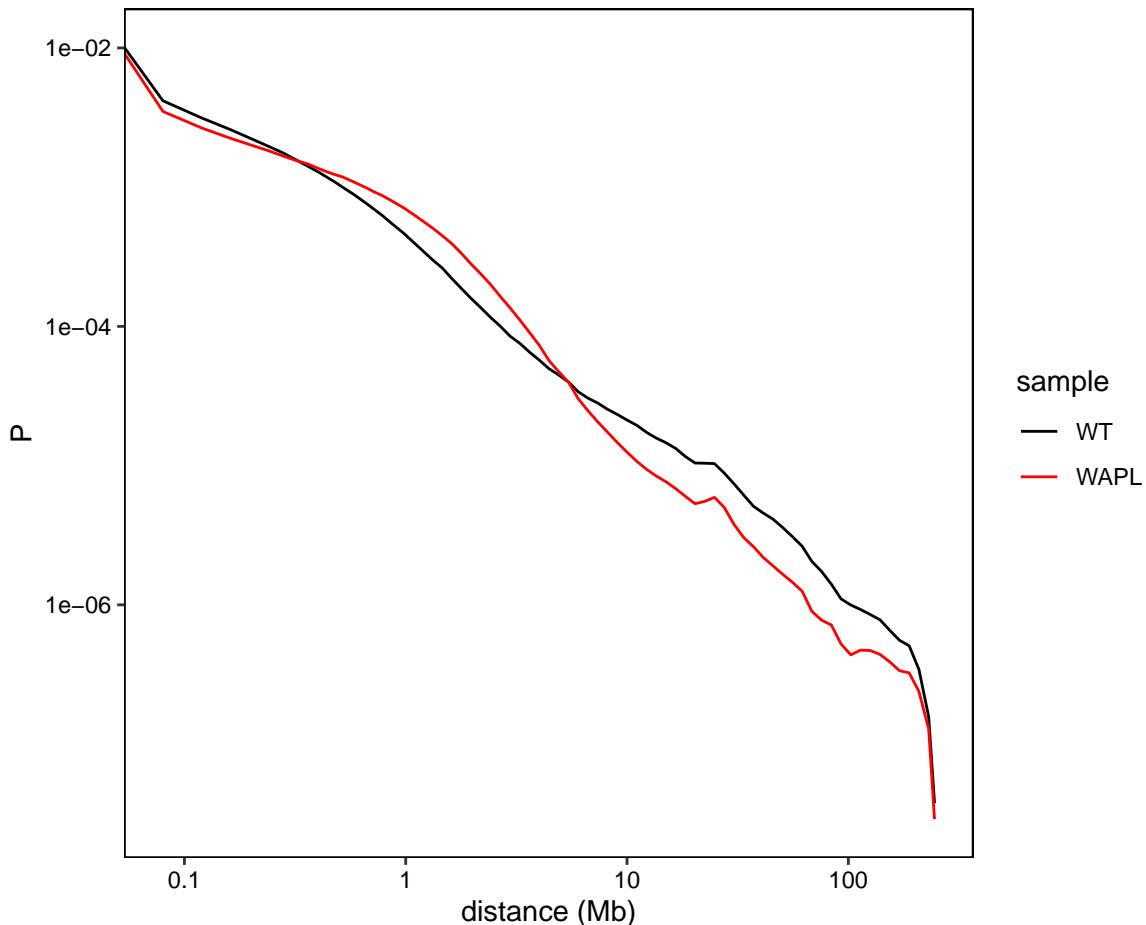
The Relative Contact Probability computes the contact probability as a function of genomic distance, as described in (Lieberman-Aiden and Berkum 2009). This can be computed for a specific set of chromosomes or genome-wide. To be able to ignore centromeric contacts (which have a abberant RCP), centromeric information is need. This is taken from the experiment-object or found emperically by comparing trans-interactions.

```
RCP_out <- RCP(explist = list(Hap1_WT_40kb, Hap1_WAPL_40kb),
                  chromsToUse = 'chr1')
```

## GENOVA: explore the Hi-C's

The user can decide to plot the RCP per chromosome. If the data is sparse, a LOESS-smoothing could be convenient. It takes the color and name from the experiment-objects. If we look at the resulting plot, we can see that the  $\Delta WAPL$  has more interactions in the  $[\pm 800kb, \pm 2Mb]$  range (figure 4). The sizes of TADs are fall into this range, so a next step could be to dive into the TAD-specific analyses (discussed below). Moreover, the  $\Delta WAPL$  has less interactions in the far-*cis* range ( $[10Mb, 100Mb]$ ): A- and B-compartments are often of these sizes, so a next step could be to look more into compartmentalisation with `cis.compartment.plot` or `trans.compartment.plot`, for example.

```
# Plot RCP: per-chromosome
visualise(RCP_out,
           smooth = T)
```



**Figure 4: RCP**

Every facet shows the RCP of one chromosome.

### 2.3.1 differentials

It is also possible to directly compare samples to one (like WT versus WAPL). For this, metric has to be set to `lfc` and contrast to 1 (figure 5). The log-fold change of average probabilities are then plotted.

## GENOVA: explore the Hi-C's

```
# Plot RCP: combined  
visualise(RCP_out, contrast = 1, metric = 'lfc')
```

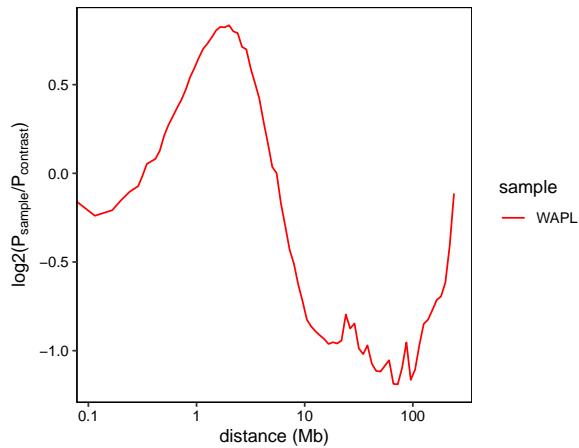


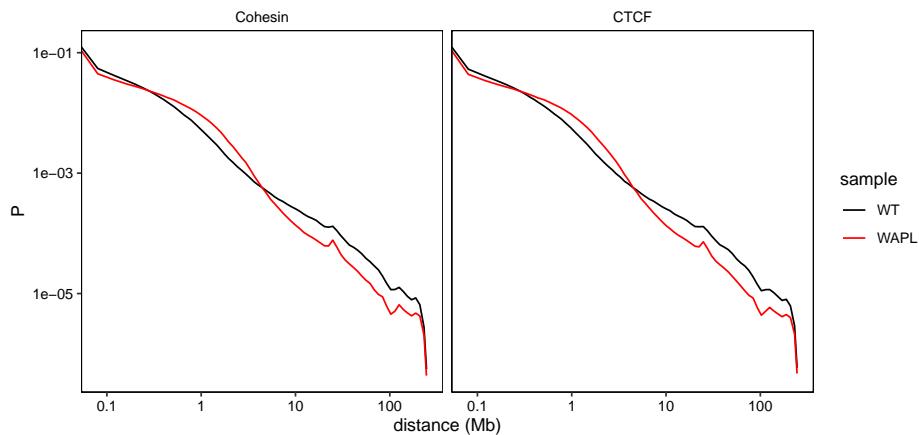
Figure 5: RCP in lfc-mode

### 2.3.2 regions

But what if you want to compare the contact probabilities of specific regions, like Cohesin- or CTCF-bound regions? For this, we added the possibility to add a list of BED-data\_frames to the bedList-argument. Under the hood, we perform a standard per-arm RCP (thus still enabling users to also set the chromsToUse-parameter), whereafter we filter out Hi-C bins that do not have entries in the dataframe(s) of bedList. The same plot-function can be used: different BED-files will have different line-types. The fact that we use linetype for the bedList entries, allows us to still use multiple samples in experimentList, as shown in figure ???. But if you only provide one experiment-object, we will use different line-colours of the different BEDs.

```
CTCF = read.delim('../data/symlinks/CTCF_WT_motifs.bed', h = F)  
SMC1 = read.delim('../data/symlinks/SMC1_WT_peaks.narrowPeak', h = F)  
  
RCP_out = RCP(list(Hap1_WT_40kb, Hap1_WAPL_40kb),  
              bedlist = list("CTCF" = CTCF,  
                            'Cohesin' = SMC1),  
              chromsToUse = 'chr1')  
  
visualise(RCP_out)  
  
visualise(RCP_out, contrast = 1, metric = 'lfc')  
#> Warning in melt(out, id.vars = "distance"): The melt generic in data.table  
#> has been passed a data.frame and will attempt to redirect to the relevant  
#> reshape2 method; please note that reshape2 is deprecated, and this  
#> redirection is now deprecated as well. To continue using melt methods from  
#> reshape2 while both libraries are attached, e.g. melt.list, you can prepend  
#> the namespace like reshape2::melt(out). In the next version, this warning  
#> will become an error.
```

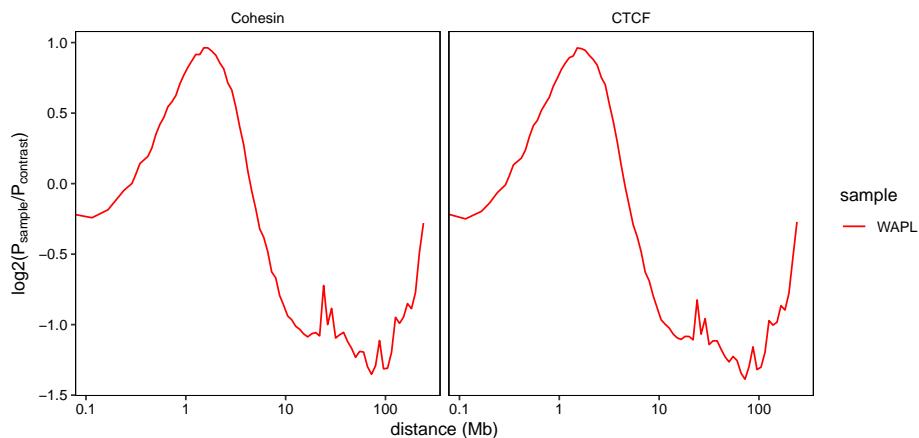
## GENOVA: explore the Hi-C's



**Figure 6: RCP with BEDs**

We can also add BEDs as sites to compute the RCP.

```
#> Warning in melt(out, id.vars = "distance"): The melt generic in data.table  
#> has been passed a data.frame and will attempt to redirect to the relevant  
#> reshape2 method; please note that reshape2 is deprecated, and this  
#> redirection is now deprecated as well. To continue using melt methods from  
#> reshape2 while both libraries are attached, e.g. melt.list, you can prepend  
#> the namespace like reshape2::melt(out). In the next version, this warning  
#> will become an error.
```



**Figure 7: RCP with BEDs**

We can also add BEDs as sites to compute the RCP.

## 2.4 A- and B-compartments

```
H3K27acPeaks = read.delim('..../data/symlinks/H3K27ac_WT.narrowPeak', h = F)  
  
CS_out = compartment_score(list(Hap1_WT_40kb, Hap1_WAPL_40kb), bed = H3K27acPeaks)
```

## GENOVA: explore the Hi-C's

```
visualise(CS_out, chr = "chr17")
```

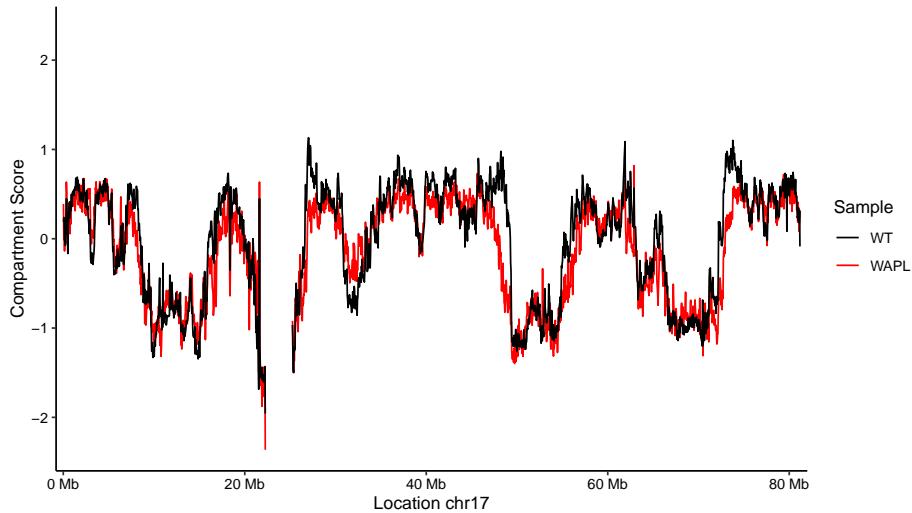


Figure 8: The compartment-scores on chromosome 17

### 2.4.1 Saddle-analyses

```
saddle_out = saddle(list(Hap1_WT_40kb, Hap1_WAPL_40kb),
                     CS_discovery = CS_out,
                     bins = 50)

visualise(saddle_out)
```

### 2.4.2 Compartment-strength

```
CSS <- quantify(saddle_out)

compared <- tidyrr::spread(unique(CSS[,-c(3,4)]), key = 'exp', value = 'strength')

plot(compared$WT, compared$WAPL, xlim = c(0,4), ylim = c(0,4), pch = 20)
abline(a = 0, b = 1)
```

## GENOVA: explore the Hi-C's

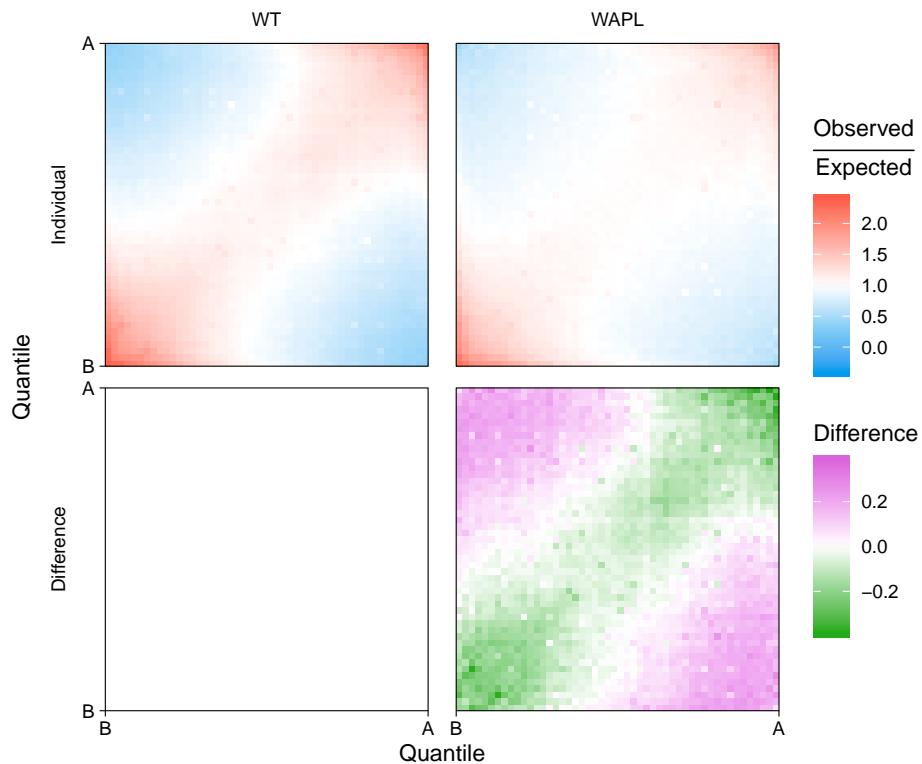


Figure 9: A saddle-plot: WAPL-knockout cells have less intra-compartment enrichment

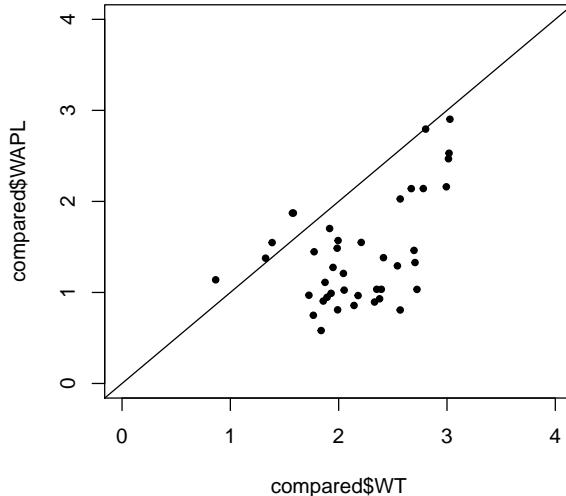


Figure 10: The per-arm compartment strength is at almost every arm lower in the WAPL knockout

## 3 Interaction plots

GENOVA has several plotting-functions for genomic loci. `cis.compartment.plot` and `trans.compartment.plot` provide an easy way to plot whole chromosome arms, including compartmentalisation-score tracks. For more zoomed-in plots `hic.matrixplot` can be used.

## GENOVA: explore the Hi-C's

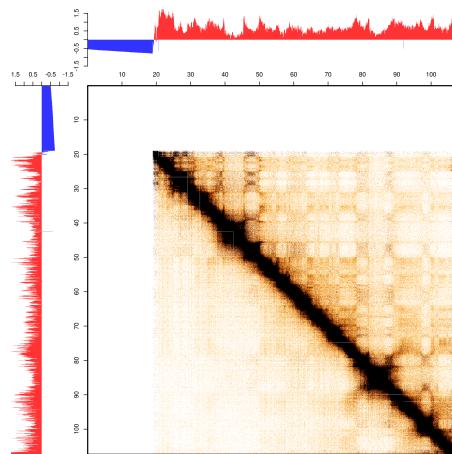
This function also allows rich annotation and between-experiment comparision possibilities. All functions try to guess the most appropriate color-scale limits, but finer control of this can be gotten by setting the `cut.off`-argument.

### 3.1 *cis*-interactions

The compartmentalisation of the chromatin into A and B van already described in the original Hi-C paper (Lieberman-Aiden and Berkum 2009). Several papers have described the loss of compartmentalisation when the Cohesin complex is stabilised (Haarhuis et al. 2017, @Wutz2017, @Gassler2017). To view this interesting level of chromatin organisation, we can use `cis.compartment.plot`. With this, we can plot one arm of a chromosome with the compartment-score plotted above. To infer which compartment is A (viewed as the active state) and which is B, we can add a BED-data frame of ChIP-seq peaks from active histone marks (e.g. H3K27ac, H3K4me1). In figure 11 you can see the resulting plots, where you can see that the checkerboard-pattern in the matrix and the amplitude of the compartment-score are diminished in the WAPL-knockout.

```
H3K27ac_peaks = read.delim('../data/symlinks/H3K27ac_WT.narrowPeak', h = F)

cis.compartment.plot(exp1 = Hap1_WT_40kb,
                     exp2 = Hap1_WAPL_40kb,
                     chrom = 'chr14',
                     arm = 'q',
                     cs.lim = 1.75, # max compartment-score
                     cut.off = 15,
                     chip = H3K27ac_peaks)
```



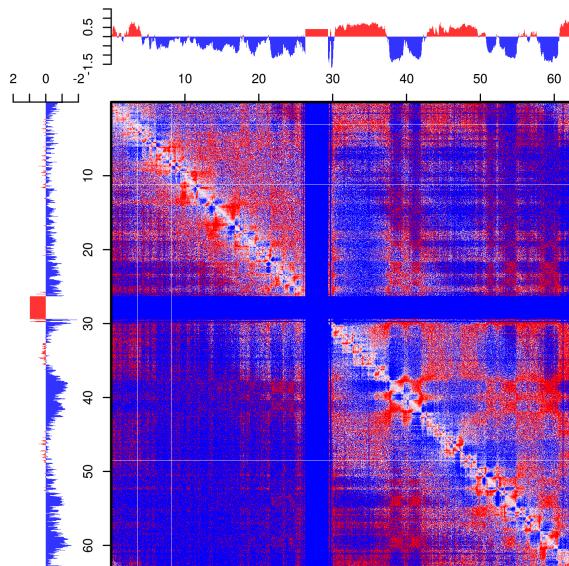
**Figure 11: Cis compartment plot: WT vs WAPL**

Stabilised Cohesin-mediated loops by WAPL-knockout leads to loss of compartments.

The compartment-score is calculated by performing an eigenvector decomposition on the correlation-matrix of the expected over expected matrix. To view this O/E matrix, we can set the `obs.exp`-option to TRUE. This view gives a visually better view of the A- and B-compartments (figure 12).

## GENOVA: explore the Hi-C's

```
cis.compartment.plot(exp1 = Hap1_WT_40kb,
                     exp2 = Hap1_WAPL_40kb,
                     chrom = 'chr20',
                     arm = 'q',
                     cut.off = 1,
                     obs.exp = T,
                     chip = H3K27ac_peaks)
```

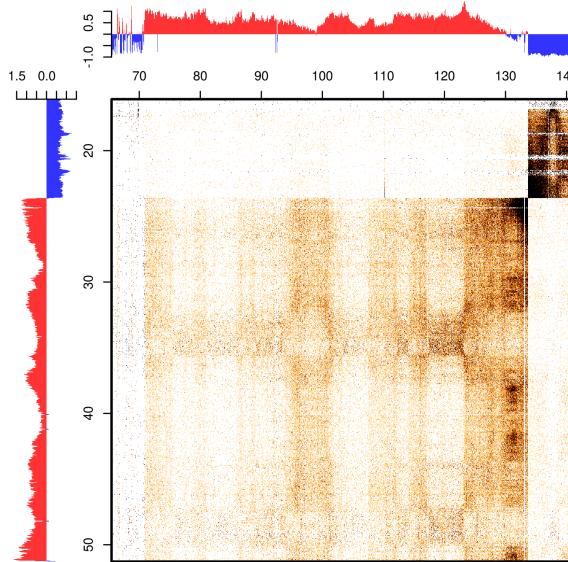


**Figure 12: Cis compartment plot**  
Observed over expected.

## 3.2 *trans*-interactions

As could be seen above, A-compartments interact more with other A-compartments and the same is true for B-compartments. However, is the same true for *trans*? The function `trans.compartment.plot` will allow the user to plot a *trans*-matrix (i.e. a matrix of the arms of two different chromosomes) along with the respective *cis* compartment-scores. This function could also be used to investigate chromosomal translocations: the 9 $q$ ;22 $q$  translocation can be clearly seen if we use this function, as in figure 13.

```
trans.compartment.plot(exp = Hap1_WT_40kb,
                       chrom1 = 'chr9',
                       arm1 = 'q',
                       chrom2 = 'chr22',
                       arm2 = 'q',
                       cut.off = 10,
                       chip = H3K27ac_peaks)
```

**Figure 13: Trans compartment plot**

The t(9q;22q) translocation is easily identified.

### 3.3 matrix plots

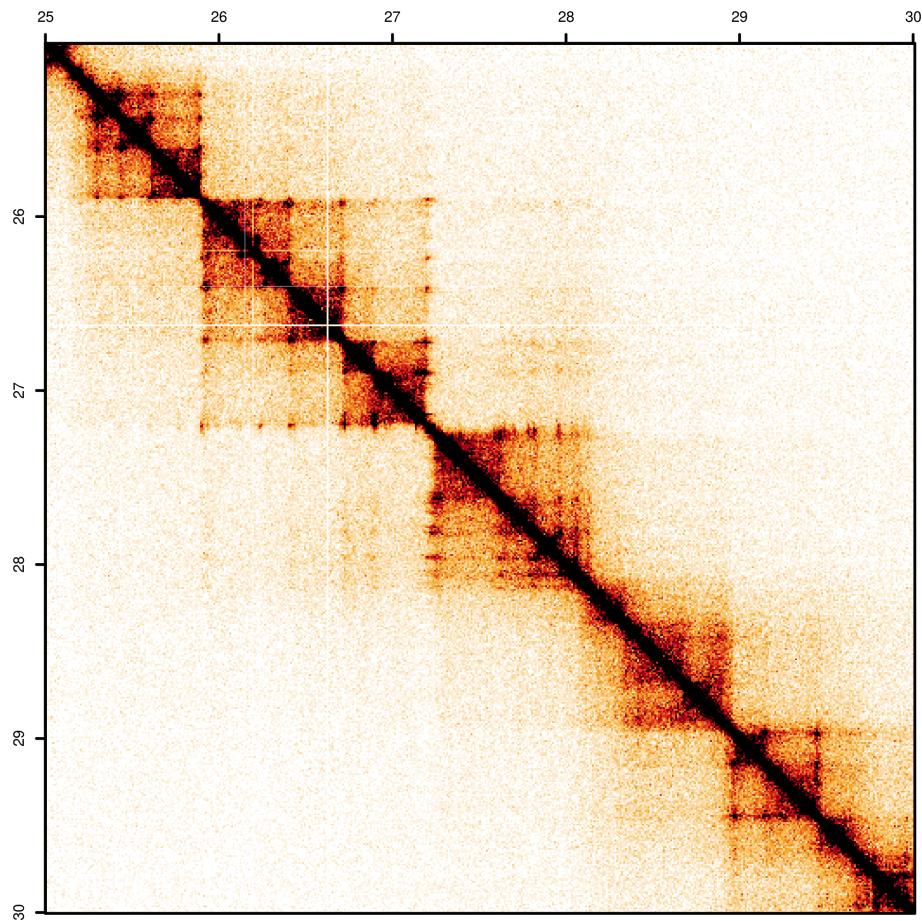
To produce richly annotated zoomed-in (i.e. max 10Mb) plots of specific regions, we use the `hic.matrixplot` function. In this, we can use one or two experiment objects: two can be shown either in diff-mode (the difference between the two) or upper/lower triangle mode. TAD- and loop-annotations can be added, as well as bigwig- and bed-tracks. Moreover, genemodel-files can be added. In this section, we will build up to a final, fully annotated, matrix from a humble one-experiment plot (figure 14).

```
hic.matrixplot(exp1 = Hap1_WT_10kb,
               chrom = 'chr7',
               start = 25e6,
               end=30e6,
               cut.off = 50) # upper limit of contacts
```

#### 3.3.1 two experiments

Adding a second experiment will give us the option of `coplot`, which can be `dual` (default) or `diff`. The first shows `exp1` in the upper triangle and `exp2` in the lower. `Exp1` is subtracted from `exp2` in `diff`-mode: red is therefore more contacts in `exp2` and blue denotes more contacts in `exp1` (figure 15).

```
hic.matrixplot(exp1 = Hap1_WT_10kb,
               exp2 = Hap1_WAPL_10kb,
               chrom = 'chr7',
               start = 25e6,
               end=30e6,
               cut.off = 50) # upper limit of contacts
```



**Figure 14: Hi-C matrixplot**

Simplest example: one experiment, no annotation

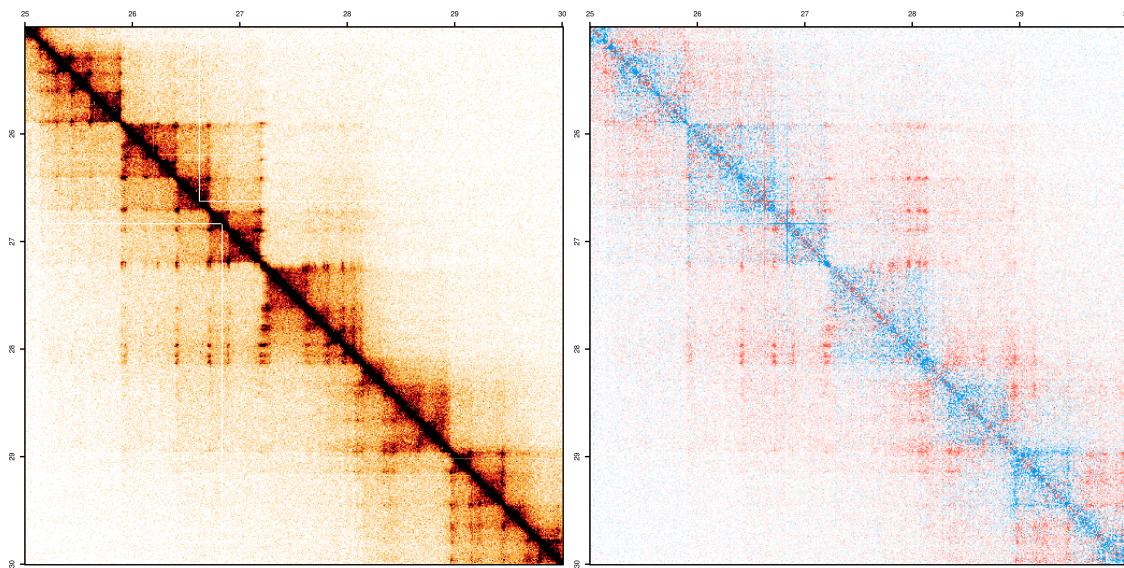
```
hic.matrixplot(exp1 = Hap1_WT_10kb,
               exp2 = Hap1_WAPL_10kb,
               coplot = 'diff',
               chrom = 'chr7',
               start = 25e6,
               end=30e6, # upper limit of contacts
               cut.off = 25)
```

### 3.3.2 TADs and loops

It can be very useful to annotate the matrix with the positions of TADs and loops: take, for example, the situation where these structures are altered in a knockout for example. We are going to use the TAD- and loop-calls of WT Hap1 20-kb matrices from Haarhuis et al. (2017), generated with HiCseg (Lévy-Leduc et al. 2014).

Lets load some TAD- and loop-annotations:

## GENOVA: explore the Hi-C's



**Figure 15: Hi-C matrixplot with two experiments: dual vs diff mode**

The extended loops in the WAPL knockout are easily seen at around 28Mb in the lower triangle in dual-mode (left panel) and as red points in diff-mode (right panel).

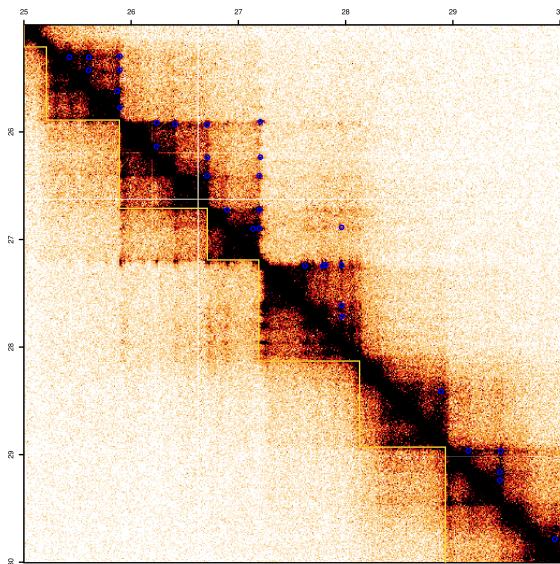
```
WT_TADs = read.delim('../data/symlinks/WT_hicseg_TADs.bed', h = F)
WT_Loops = read.delim('../data/symlinks/WT_HICCUPS.bedpe', h = F, skip = 1)
sanborn2015_Loops = read.delim('../data/symlinks/GSE74072_Hap1_HICCUPS_looplist.txt.gz')
```

Add them to the plot by using the `tad-` and `loops-`arguments. Both can be plotted in one or both of the triangles and colored as wished (figure 16). Since loops are very small in a hic-matrixplot, they will be fully overlapped by the loop-annotations. To overcome this, we expand the annotations with a fixed bp using `loops.resize`. This will lead to a more box-like annotation surrounding the loop.

```
hic.matrixplot(exp1 = Hap1_WT_10kb,
               chrom = 'chr7',
               start = 25e6,
               end=30e6,
               loops = WT_Loops, # see APA
               loops.colour = 'blue', # purple loops
               loops.type = 'upper', # only plot in upper triangle
               loops.radius = 20e3, # expand for visibility
               tads = WT_TADs, # see ATA
               tads.type = 'lower', # only plot in lower triangle
               tads.colour = '#ffd92f', # green TAD-borders
               cut.off = 25) # upper limit of contacts
```

### 3.3.3 BigWigs and BEDs

Manipulation of CTCF-binding sites can result in loss or gain of loops and/or TADs (Wit et al. 2015). If one is interested in the effects of a knockout on the binding of a protein in combination with changes in interaction frequencies, adding ChIP-seq signal or -peaks to



**Figure 16:** Hi-C matrixplot: TAD- and loop-annotations from Haarhuis et al (2017).

**Table 3:** A data.frame holding a standard BED6 format

V1	V2	V3	V4	V5	V6
chr1	237749	237768	GCAGCACCAGGTGGCAGCA	1412	+
chr1	714180	714199	CGGCCACCAGTAGGCAGCG	1428	-
chr1	793458	793477	CCACCAGCAGGTGGCTCC	1160	-

the matrix can be very helpfull. Two tracks above and two tracks to the left can be added. These can be either BED-like data.frames or the paths the .bw files. For example, lets load a BED6-file (chrom, start, end, name, score, and strand<sup>2</sup>) of CTCF-motifs under CTCF-ChIP peaks. The argument type can be set to either *triangle* or *rectangle*: triangle is nice to use if you want to look at the orientation of the BED-entries (figure 17). If you only have a three column BED, then the output will always be *rectangle*.

<sup>2</sup><https://genome.ucsc.edu/FAQ/FAQformat.html>

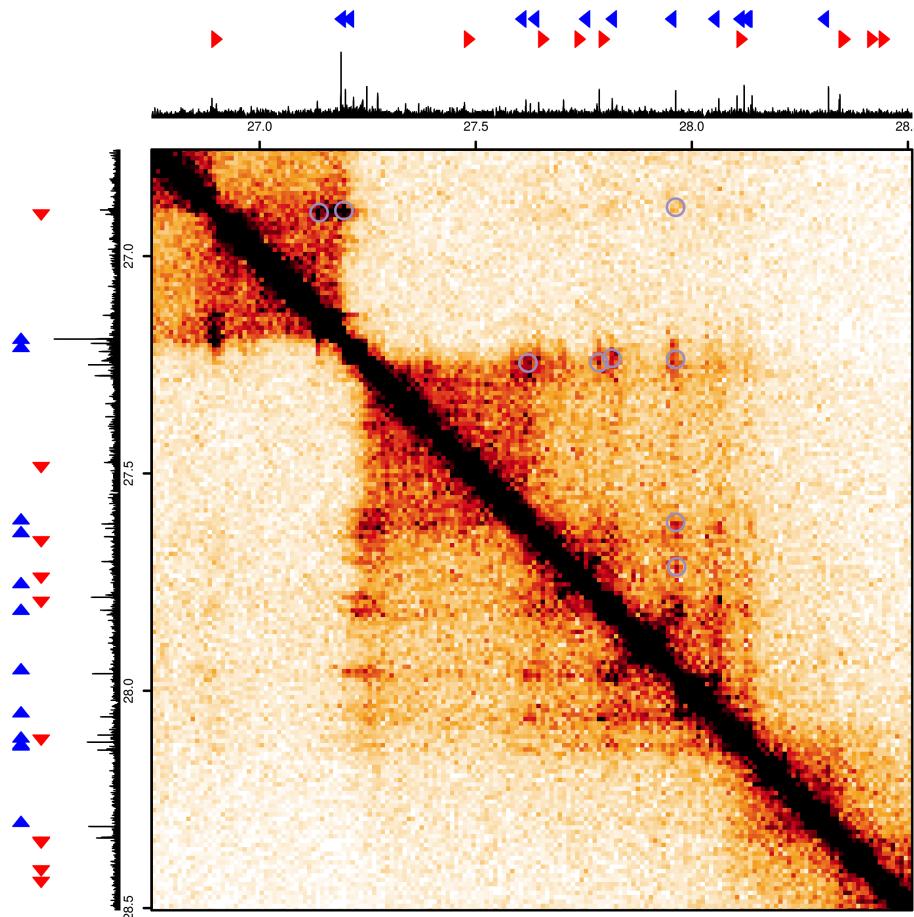
```
CTCF = read.delim('../data/symlinks/CTCF_WT_motifs.bed', h = F)
SMC1 = read.delim('../data/symlinks/SMC1_WT_peaks.narrowPeak', h = F)
```

Moreover, we can use a bigwig (.bw) file to plot a track. For this example, we are using a SMC1 ChIP-seq track from (Haarhuis et al. 2017). We need the `bigwrig` package, which is easily installed from github using `devtools::install_github()`. The `yMax` argument is handy if you want to compare bigwig-tracks: it lets you set the y-axis maximum.

```
hic.matrixplot(exp1 = Hap1_WT_10kb,
               chrom = 'chr7', start = 26.75e6, end=28.5e6,
               loops = WT_Loops, # see APA
               loops.colour = '#998ec3', # purple loops
               loops.type = 'upper', # only plot in upper triangle
               loops.radius = 20e3, # expand for visibility
               type = 'triangle',
               chip = list('../data/symlinks/SMC1_WT.bw', # inner top
                          CTCF),# outer-top
```

## GENOVA: explore the Hi-C's

```
symmAnn = F, # place annotations also on left side
cut.off = 65) # upper limit of contacts
#> chip.yMax not given for a .bw-track: yMax is 2.27817010879517
#> chip.yMax not given for a .bw-track: yMax is 2.27817010879517
```



**Figure 17: Hi-C matrixplot: ChIPseq**

A BED-file of CTCF-sites is plotted at the top and a coverage-track of SMC1 ChIP-seq is plotted beneath this. The symmAnn-option leads to the same tracks being plotted on the left.

### 3.3.4 Genes

(Dixon et al. 2012) showed that housekeeping-genes are enriched in the vicinity of TAD-borders. Another interesting question could be whether differentially expressed genes are also found near TAD-borders or binding sites of specific proteins when studying a knockout. These type of questions can be tackled by adding the appropriate gene-models to `hic.matrixplot`. To do this, we make use of the `data.fame`, where each row is an exon from a gene. There are several ways to get this. One of the easiest is to use biomart to get exon-coordinates. This can be done with the `biomaRt`-package or via the web-based service. For this example, we downloaded data of all exons from the Ensembl biomart and plotted both the BED-file and the genes (figure 18).

**Table 4:** A data.frame holding the needed columns for plotting genes

	chrom	txStart	txEnd	exonStart	exonEnd	strand
38182	chr7	1022835	1029276	1022835	1023021	+
38183	chr7	1022835	1029276	1024048	1024210	+
38184	chr7	1022835	1029276	1024586	1024735	+
38185	chr7	1022835	1029276	1024802	1024959	+
38186	chr7	1022835	1029276	1026260	1026433	+

```
# features downloaded:
## Gene stable ID & Transcript stable ID & Chromosome/scaffold name &
## Transcript start (bp) & Transcript end (bp) & Exon region start (bp) &
## Exon region end (bp) & Strand
# martExport = read.delim('../data/mart_export.txt.gz', stringsAsFactors = F)
# colnames(martExport) = c('ENSG','ENST','chrom' , # change column names
#                         'txStart' , 'txEnd' ,
#                         'exonStart' , 'exonEnd' , 'strand')
# martExport$chrom = gsub(martExport$chrom, # add chr-prefix
#                         pattern = '^',
#                         replacement = 'chr')
# martExport$strand = ifelse(martExport$strand == 1, '+', "-") # 1/-1 to +/--
load('../data/martExport.Rdata')

hic.matrixplot(exp1 = Hapl_WT_10kb,
               chrom = 'chr7', start = 26.75e6, end=28.5e6,
               genes = martExport,
               cut.off = 65) # upper limit of contacts
```

### 3.3.5 Everthing together

Finally, we can combine all these options in one. This may be complete overkill, but it could be quite handy. In this example, we can see that most TAD-borders and loop-anchors have clear SMC1- and CTCF-signal (figure 19). Both these are expected to be found at these locations according to the *chromatin extrusion model*. Moreover, we can also see that the CTCF-orientation of the upstream and downstream loop-anchor are forward and reverse, resp. This *convergent rule* is a known feature of loops (de Wit et al. 2015).

```
hic.matrixplot(exp1 = Hapl_WT_10kb,
               chrom = 'chr7',
               start = 25e6,
               end=28.5e6,
               loops = WT_Loops, # see APA
               loops.colour = '#998ec3', # purple loops
               loops.type = 'upper', # only plot in upper triangle
               loops.radius = 20e3, # expand for visibility
               genes = martExport,
               chip.colour = 'black',
               chip = list('../data/symlinks/SMC1_WT.bw', # inner-top
                           SMC1, # outer-top
```

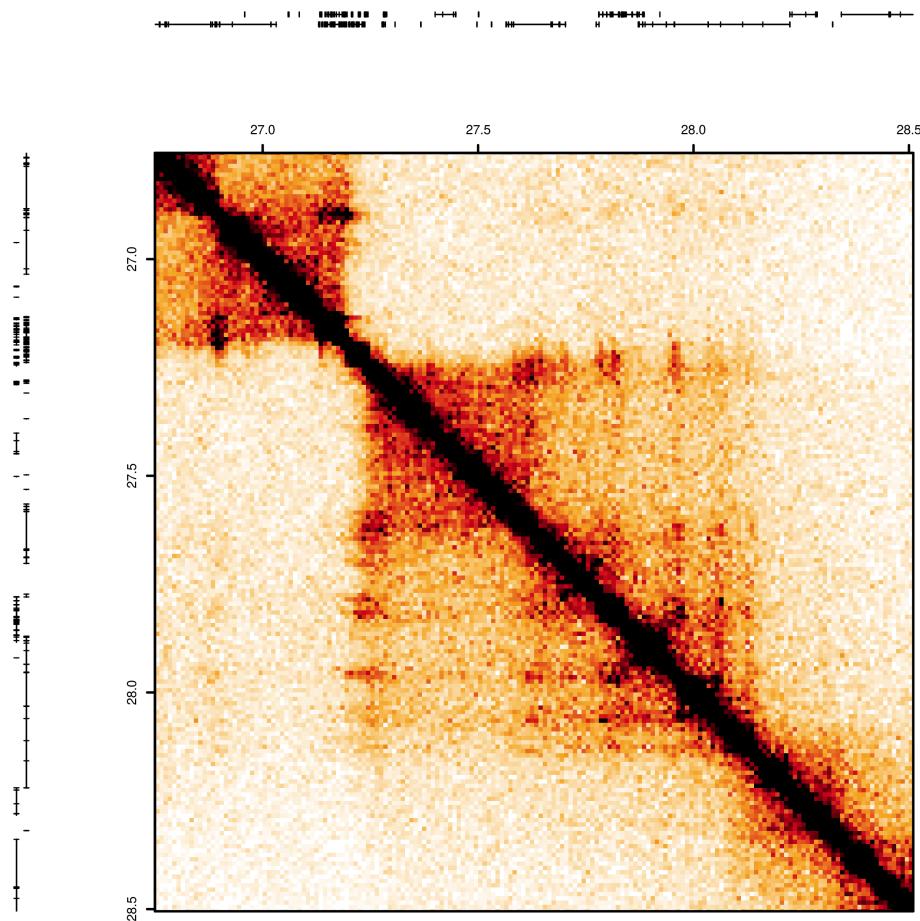
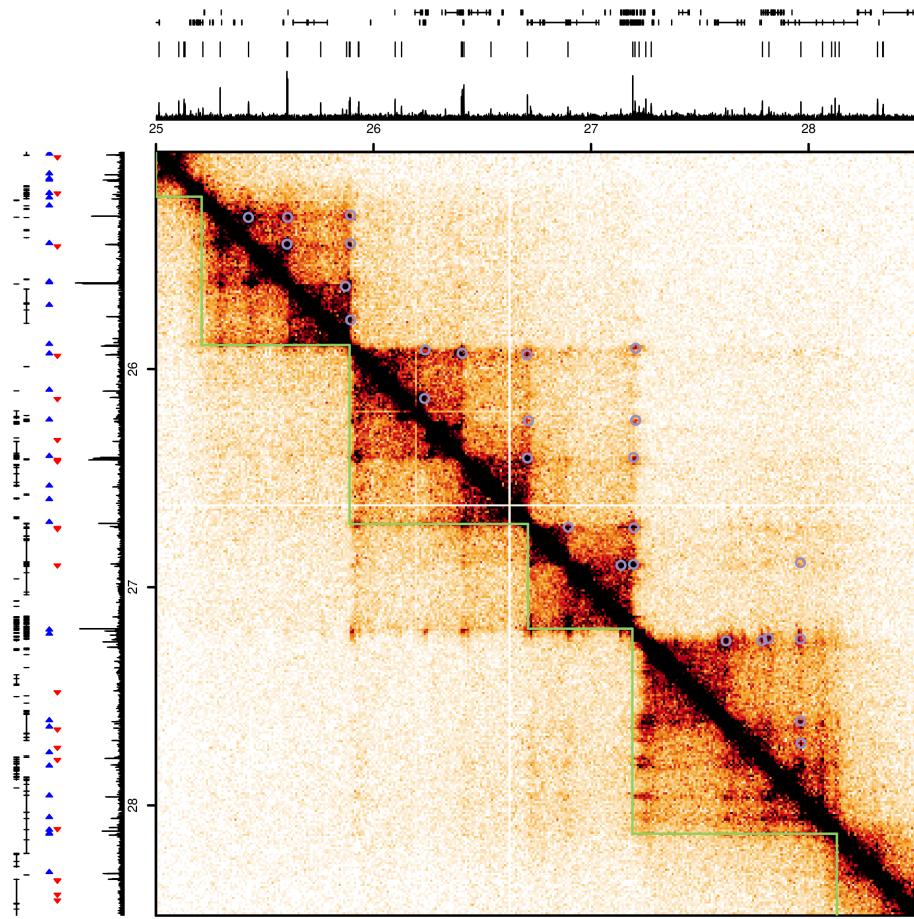


Figure 18: Hi-C matrixplot: genes

```
'../data/symlinks/SMC1_WT.bw', # inner-left
CTCF), # outer-left
tads = WT_TADs, # see ATA
tads.type = 'lower', # only plot in lower triangle
tads.colour = '#91cf60', # green TAD-borders
cut.off = 50) # upper limit of contacts
```

**Figure 19: Hi-C matrixplot: a complex case**

Loops and TADs are annotated within the Hi-C matrix. On the top annotation-bar, we have plotted the ChIP-seq signal and peaks of SMC1. On the left annotation-bar are the ChIP-seq signal and peaks (with orientation) of CTCF. Genes are plotted on both annotation-bars.

## 4 TADs

Topologically Associated Domains (TADs) are  $\pm 0.8 - 2 Mb$  regions, which are seen as triangles in the matrix: regions that have more interactions within than outside. GENOVA has a repertoire of functions to generate and analyse TADs. First, we will use the insulation score to call TADs and compare the strength of TAD-borders between samples. Next, we will explore ATA to analyse aggregates of TADs. Finally, we will investigate whether TADs interact with their neighbouring TADs.

### 4.1 Insulation

To estimate the strength of TAD-borders, we can look at the insulation-score (Crane et al. 2015). At a TAD-border, this score reaches a local minimum: the lower the score, the stronger the insulation. We can generate this for a specific sliding-window size with `insulation_score`. The choice of window-size is quite tricky, since smaller will be sensitive to very local effect

## GENOVA: explore the Hi-C's

(i.e. mapping-errors, loops), while too big windows will lead to an underrepresentation. Luckily, we can generate a domainogram of a range of window-sizes for a specific genomic region with `insulation.domainogram`.

### 4.1.1 Domainogram

To make a domainogram, we simply choose our experiment and our region of interest. The window-size is directly proportional to the amount of Hi-C bins.

```
ID <- insulation_domainogram(Hap1_WT_10kb,
                                'chr7',
                                25.5e6,
                                30e6,
                                window_range = c(1, 101),
                                step = 2)
```

A nice feature of `hic.matrixplot` is that if you use it without plotting anything on the sides (i.e. genes and/or ChIP-tracks), you can insert other plots. This allows us to plot the domainogram directly under the matrix, making it very easy to compare the insulation with the actual data (figure ??).

```
hic.matrixplot(exp1 = Hap1_WT_10kb,
               chrom = 'chr7',
               start = 25e6,
               end=29e6,
               tads = WT_TADs, # see ATA
               tads.type = 'upper', # only plot in lower triangle
               tads.colour = '#91cf60', # green TAD-borders
               cut.off = 25, # upper limit of contacts
               skipAnn = T) # skip the outside annotation

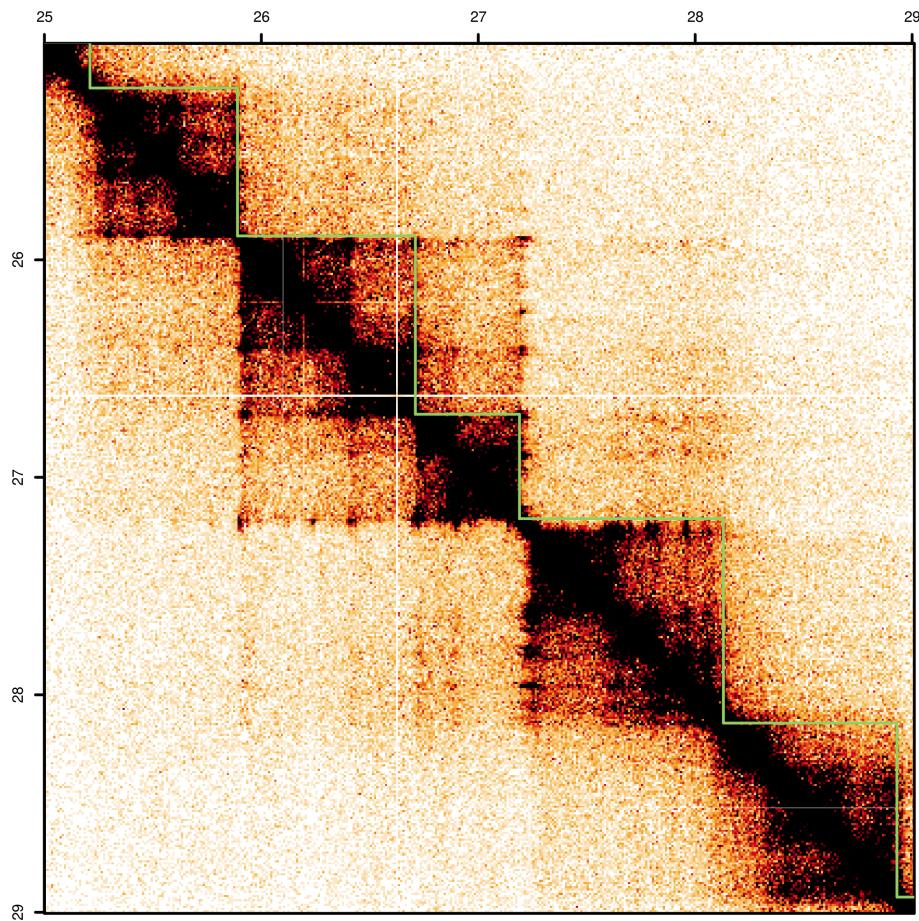
plot(ID)
```

### 4.1.2 Computing the insulation score

To get the genome-wide insulation score in .bedgraph-format,<sup>3</sup> we provide the `genome.wide.insulation` with an experiment-object and the window-size of choice. As can be seen in the domainogram above, at  $W = 25$  we will catch the majority of the hotspots, while limiting the amount of noise. The `visualise()`-function can show both the insulation-scores and the difference between them.

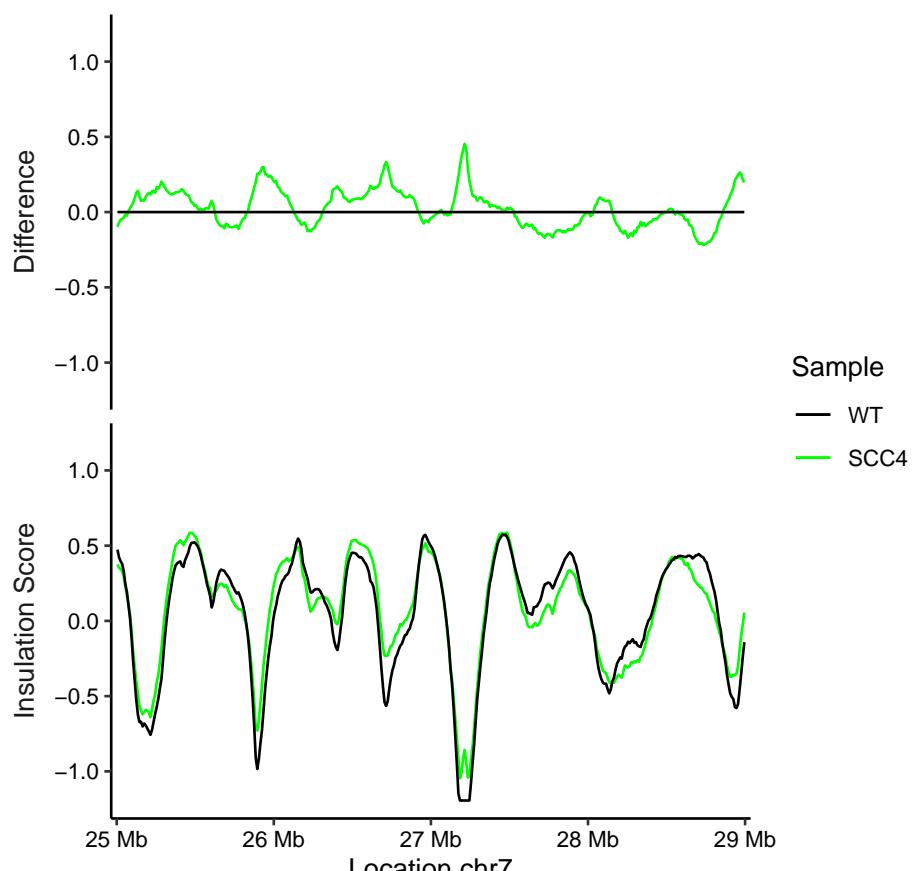
<sup>3</sup>BED3 + signal column

```
Hap1_10kb_insulation = insulation_score(list(Hap1_WT_10kb, Hap1_SCC4_10kb),
                                         window = 25)
visualise(Hap1_10kb_insulation, chr = 'chr7', start = 25e6, end=29e6, contrast = 1)
```

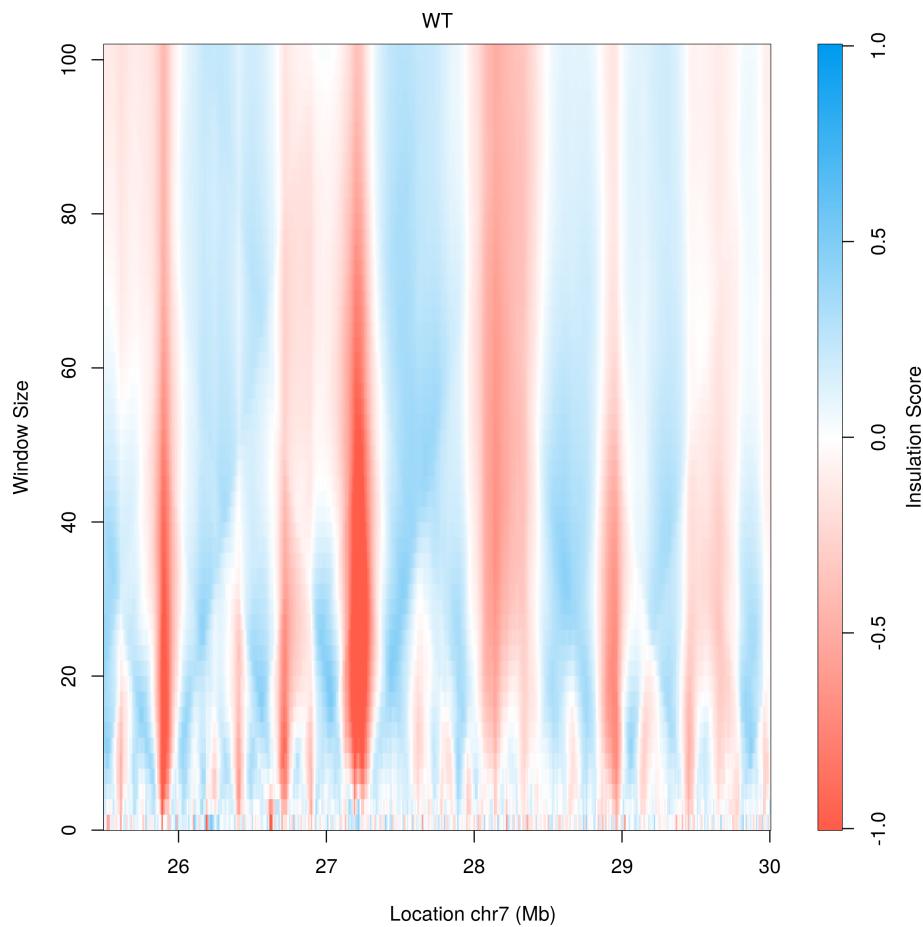


**Figure 20: Insulation domainogram with Hi-C matrix**

The insulation-hotspots are the sites where HiC-seg has called a TAD-border.



## GENOVA: explore the Hi-C's



**Figure 21: Insulation domainogram with Hi-C matrix**

The insulation-hotspots are the sites where HiC-seg has called a TAD-border.

### 4.1.3 Insulation-heatmap

We can align the border-strength of TADs in multiple samples to a specific BED-file, to compare “*borderness*” of feature. For example, let’s use the TAD-borders from (Haarhuis et al. 2017). In figure 22 we can see that the average signal drops at the border (which is to be expected) and that this is a genome-wide feature, as we see in the heatmap.

```
heatmap_insulation(Hap1_10kb_insulation, bed = CTCF, bed_pos = 'center')
```

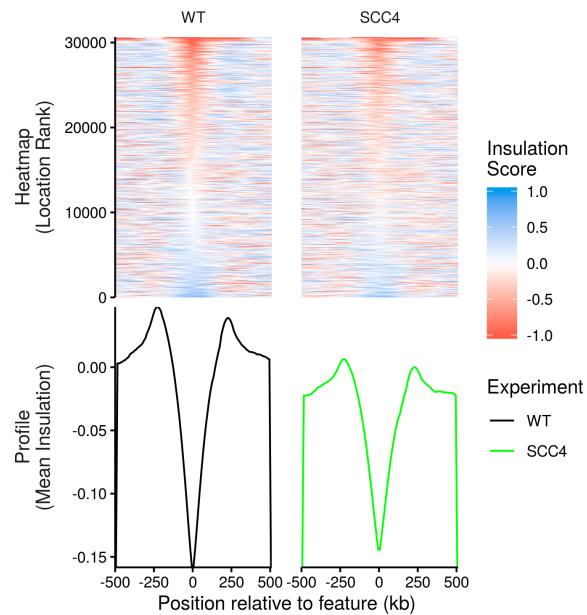
## 4.2 Call TADs

The `insulation-discovery` object can also be used to call TADs.

```
TADcalls <- call_TAD_insulation(Hap1_10kb_insulation)
```

```
hic.matrixplot(exp1 = Hap1_WT_10kb,
               exp2 = Hap1_SCC4_10kb,
```

## GENOVA: explore the Hi-C's



**Figure 22: Insulation heatmap**

The upper panel shows the average score. Each row is a TAD-border in the lower panel.

```
chrom = 'chr7',
start = 24e6,
end=27e6,
tads = list(TADcalls$SCC4, TADcalls$WT), # see ATA
tads.type = list('lower', 'upper'), # only plot in lower triangle
tads.colour = c('green', 'grey'), # green TAD-borders
cut.off = 25) # upper limit of contacts
```

## 4.3 ATA

```
ATA_Hap1_WTcalls <- ATA(list("WT" = Hap1_WT_10kb,
                                'WAPL' = Hap1_WAPL_10kb),
                           bed = TADcalls$WT)
```

We can use `visualise` to plot the ATA-results.

```
visualise(ATA_Hap1_WTcalls,
          colour_lim = c(0,50),
          colour_lim_contrast = c(-5,5),
          focus = 1) # which entry to use as comparison
```

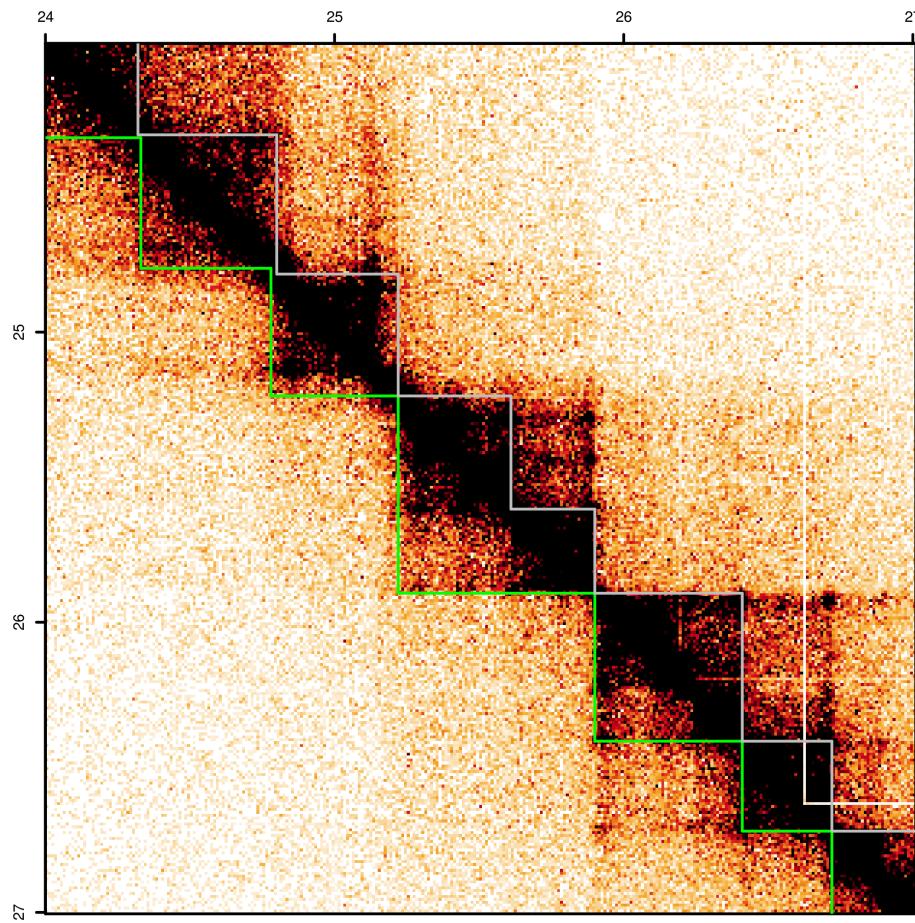


Figure 23: TADs called within GENOVA

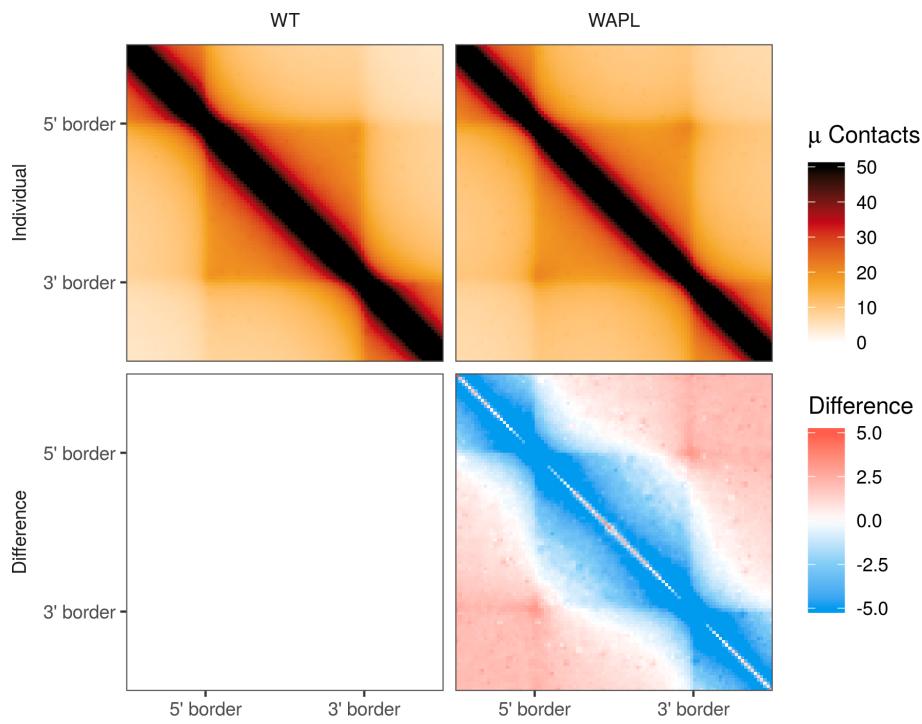
#### 4.4 TAD+N

```
TAD_N_WT <- intra_inter_TAD(list("WT" = Hap1_WT_10kb,
                                    'WAPL' = Hap1_WAPL_10kb),
                                    tad_bed = TADcalls$WT,
                                    max_neighbour = 10)
```

We can compute the enrichment of contacts between TADs with the `visualise()`-function.

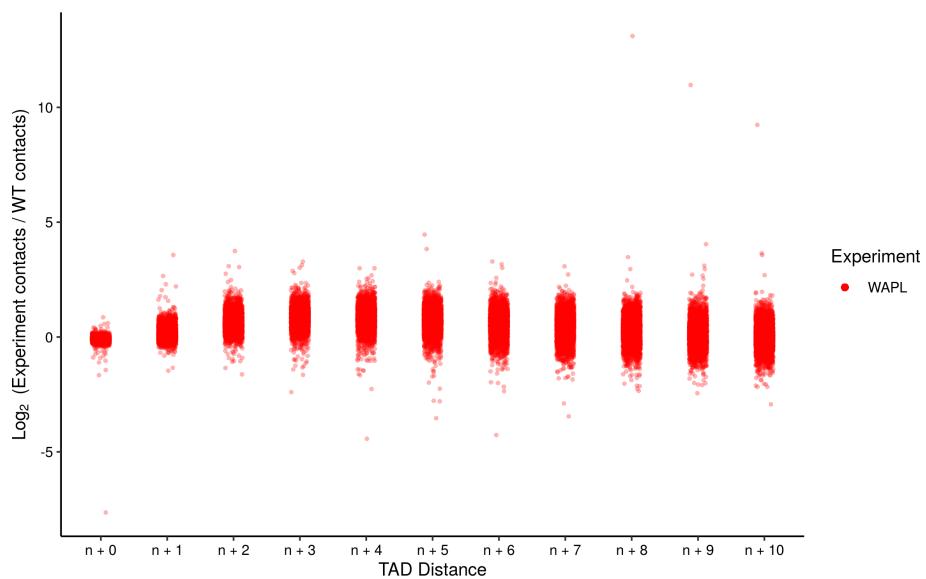
```
visualise(TAD_N_WT, geom = 'jitter')
```

```
visualise(TAD_N_WT, geom = 'violin')
```



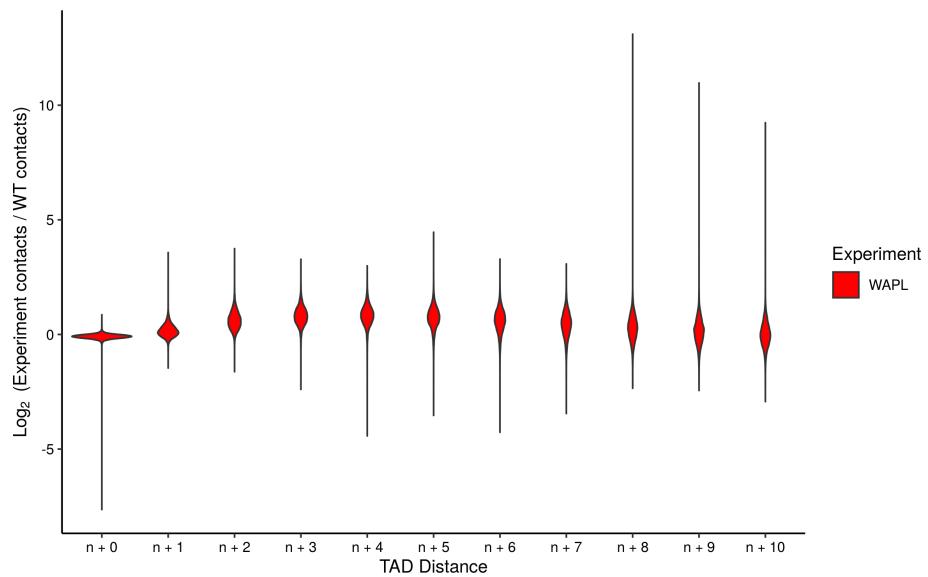
**Figure 24: ATA**

In the WAPL-knockout, we see a decrease of contacts within the TAD, but an increase at the corner.



**Figure 25: Differential TAD-analysis**

Experiment 2 (WAPL) has more interactions between neighbouring TADs compared to wild type.



**Figure 26: Differential TAD-analysis**

Experiment 2 (WAPL) has more interactions between neighbouring TADs compared to wild type.

## 5 Loops

For this section, we are using both the called and the extended loops from Haarhuis et al. (2017). These are the anchor-combinations of the merged loop-calls of WT Hap1 5-, 10- and 20-kb matrices, generated with HICCUPS (Rao et al. 2014).

```
WT_Loops_extended = anchors_extendedloops(Hap1_WT_10kb$IDX,
                                         res = 10e3,
                                         bedpe = WT_Loops)
```

```
\begin{table}[t]
\caption{Anchor-combinations of anchors_extendedLoops}

\begin{array}{|c|c|} \hline
109 & 146 \\ \hline
109 & 168 \\ \hline
109 & 170 \\ \hline
109 & 186 \\ \hline
109 & 234 \\ \hline
\end{array}

\end{table}
```

### 5.1 APA

Explain smallthreshold

```
APA_Hap1_WT <- APA(list("WT" = Hap1_WT_10kb,
                           'WAPL' = Hap1_WAPL_10kb),
                           bedpe = WT_Loops)

APA_Hap1_WT_extended <- APA(list("WT" = Hap1_WT_10kb,
                                    'WAPL' = Hap1_WAPL_10kb),
                                    bedpe = WT_Loops,
                                    anchors = WT_Loops_extended)
```

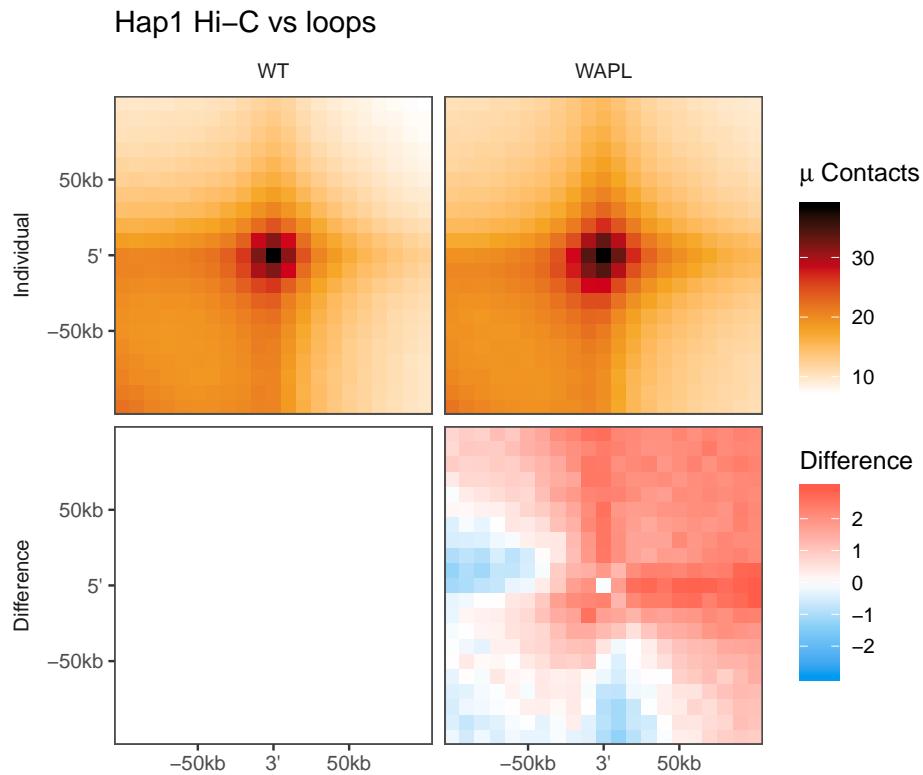
We can use `visualise.APA.ggplot` to combine the APA-results.

```
visualise(APA_Hap1_WT,
          title = "Hap1 Hi-C vs loops",
          zTop = c(0,9.5), # set the zlims of the upper row
          zBottom = c(-5,5),
          focus = 1) # which item in APAlist to use as comparison
```

```
visualise(APA_Hap1_WT_extended,
          title = "Hap1 Hi-C vs extended loops",
          zTop = c(0,9.5), # set the zlims of the upper row
          zBottom = c(-5,5), focus = 1) # which item in APAlist to use as comparison
```

To get some basic statistics on the output(s) of APA-run(s), we use `quantify()`. This function averages the region surrounding the center of each loop, where a region is defined as a square of `pixWidth × pixWith`.

## GENOVA: explore the Hi-C's



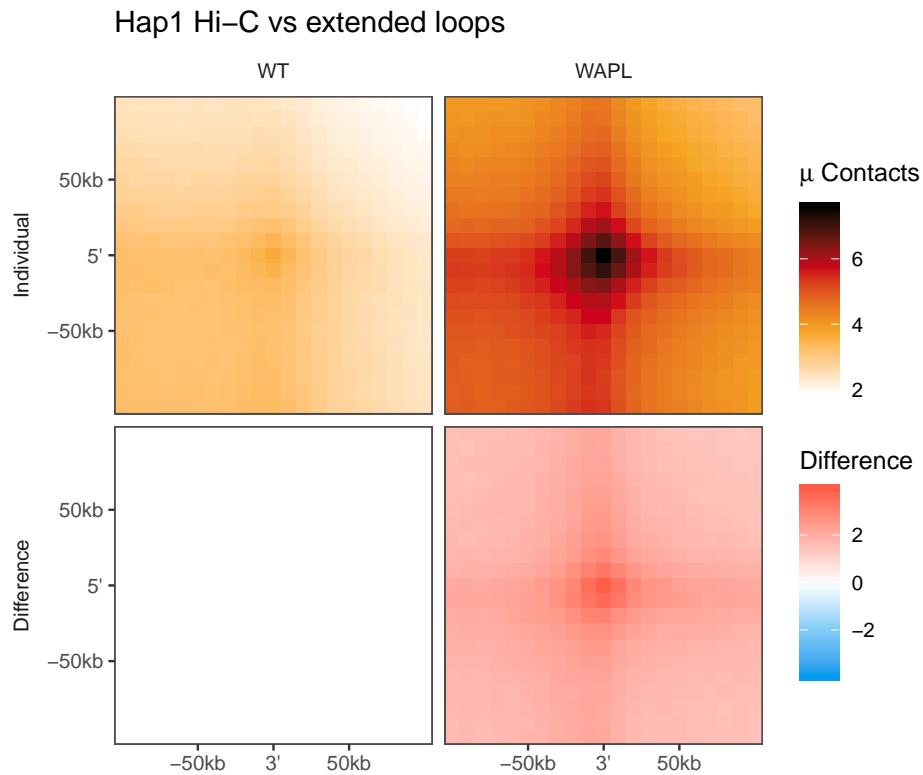
**Figure 27: APA**

In the WAPL-knockout, we see an increase of contacts at the loop.

```
quantifyAPA_out <- quantify(APA_Hap1_WT)
quantifyAPA_out_extended <- quantify(APA_Hap1_WT_extended)

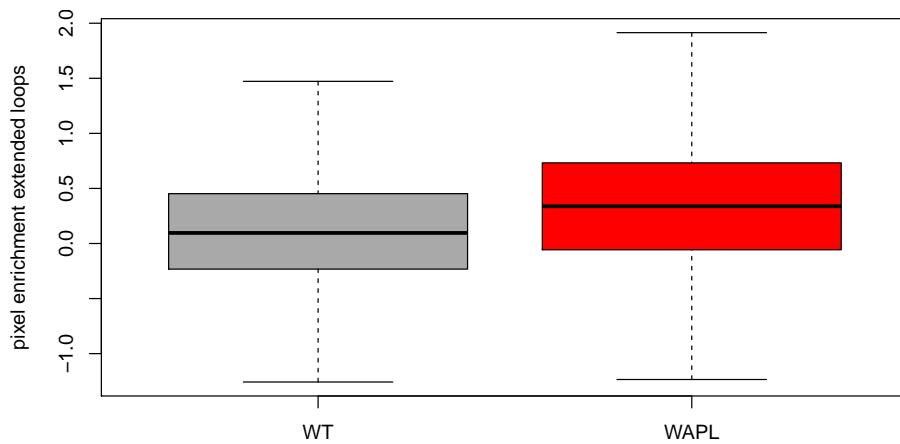
# barplot(quantifyAPA_out$per_sample$contrast_lfc, ylim = c(0, 2))
# barplot(quantifyAPA_out_extended$per_sample$contrast_lfc, ylim = c(0, 2))

# pot boxplot with base-R (ggplot2 would be also easy)
boxplot(split(quantifyAPA_out_extended$per_loop$lfc, f = quantifyAPA_out_extended$per_loop$sample_name),
       col = c('darkgrey', 'red'), outline = F, ylab = 'pixel enrichment extended loops')
```



**Figure 28: APA**

In the WAPL-knockout, we see an increase of contacts at the loop.



**Figure 29:** With `quantifyAPA` In the WAPL-knockout, we see an increase of contacts at the loop

**Table 5:** A data.frame holding the output of homer's findPeaks -style super

V1	V2	V3	V4	V5	V6
chr16-182	chr16	73074453	73092750	+	2572.8
chr12-14931	chr12	122219417	122249906	+	2532.3
chr2-1474	chr2	133025386	133026123	+	2523.7
chr11-4061	chr11	797422	842970	+	2227.4
chr15-2899	chr15	89158155	89165379	+	2087.3

## 6 Far-cis interactions

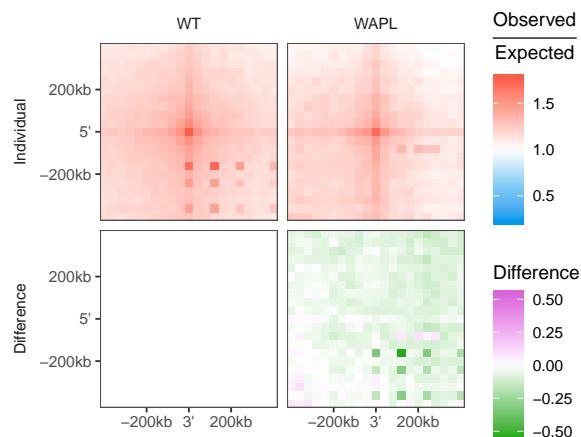
### 6.1 PE-SCAn

Some regulatory features, like super-enhancers come together in 3D-space. To test this, we implemented PE-SCAn. Here, the enrichment of interaction-frequency of all pairwise combinations of given regions is computed. The background is generated by shifting all regions by a fixed distance (1Mb: can be changed with the shift-argument).

```
superEnhancers = read.delim('..../data/symlinks/superEnhancers.txt',
                            h = F,
                            comment.char = "#")
```

The basic visualisation is comparable to ATA and APA: the first row shows the enrichment of all included samples, while the bottom row shows the difference.

```
WT_PE_OUT = PEScan(list(Hap1_WT_40kb, Hap1_WAPL_40kb), bed = superEnhancers[,2:4])
visualise(WT_PE_OUT)
```

**Figure 30: PE-SCAn**

There is a pairwise enrichment of contacts between Superenhancers, compared to shifted regions in the WT.

Another way of looking at the PE-SCAn results, is to make a perspective plot. Here, the enrichment is encoded as the z-axis.

```
RES = 40e3 # resolution of the Hi-C
```

## GENOVA: explore the Hi-C's

```
persp(list(x = seq(-1*(RES*10),(RES*10), length.out = 21)/1e6, # x-ticks (MB)
            y = seq(-1*(RES*10),(RES*10), length.out = 21)/1e6, # y-ticks (MB)
            z = WT_PE_OUT$obsexp[,,'WAPL']), # PE-SCAn out
            phi = 25, # colatitude
            theta = 40, # rotation
            col="#92c5de", # color of the surface
            shade=0.4, # how much shading
            xlab="",
            ylab="",
            zlab="",
            cex.axis = .6,
            ticktype="detailed",
            border=NA,
            zlim = c(min(c(WT_PE_OUT$obsexp)),
                     max(c(WT_PE_OUT$obsexp))))
```

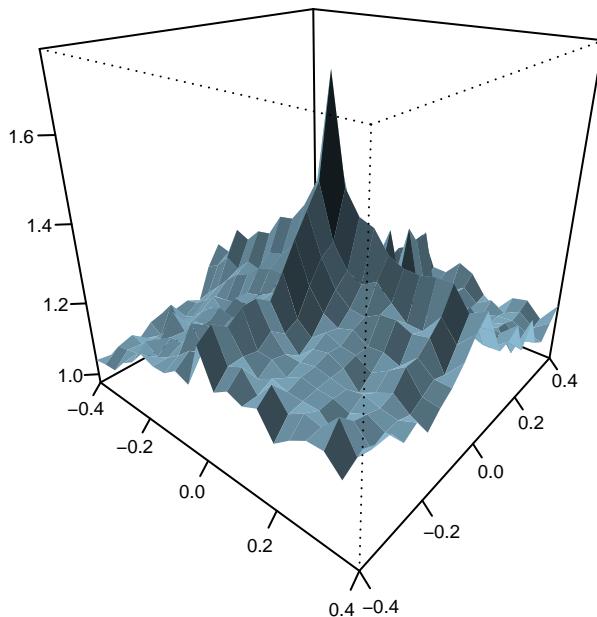


Figure 31: PE-SCAn perspective plot

## 6.2 centromere.telomere.analysis

```
centromere.telomere.analysis  
draw.centromere.telomere We saw a enriched signal between chromosomes 15 and 19. We  
can wh  
out1519 = centromere.telomere.analysis(Hap1_WT_40kb, chrom.vec = c('chr15', 'chr19'))  
draw.centromere.telomere(out1519)
```

Please post questions, comments and rants on [our github issue tracker](#).

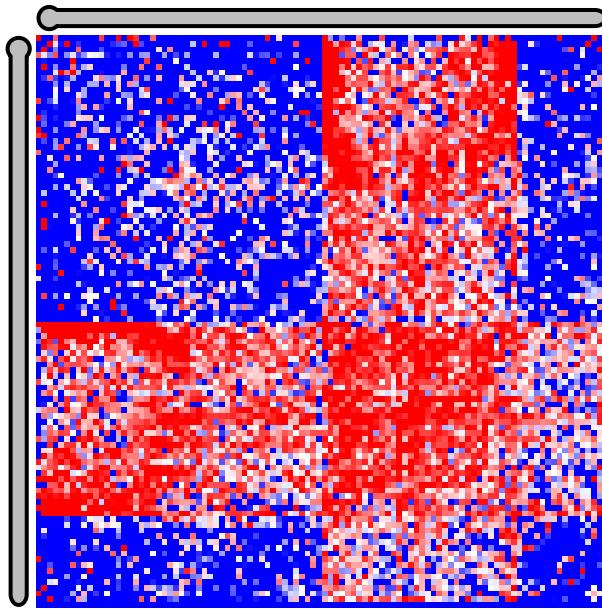


Figure 32: Centromere-telomere plot of chromosomes 15 and 19

## 7 Session info

```
#> R version 3.6.2 (2019-12-12)
#> Platform: x86_64-pc-linux-gnu (64-bit)
#> Running under: Ubuntu 16.04.6 LTS
#>
#> Matrix products: default
#> BLAS: /usr/lib/openblas-base/libblas.so.3
#> LAPACK: /usr/lib/libopenblas-p-r0.2.18.so
#>
#> locale:
#> [1] LC_CTYPE=en_US.UTF-8      LC_NUMERIC=C           LC_TIME=C
#> [4] LC_COLLATE=en_US.UTF-8   LC_MONETARY=C          LC_MESSAGES=en_US.UTF-8
#> [7] LC_PAPER=C               LC_NAME=C            LC_ADDRESS=C
#> [10] LC_TELEPHONE=C         LC_MEASUREMENT=C    LC_IDENTIFICATION=C
#>
#> attached base packages:
#> [1] stats      graphics   grDevices  utils      datasets   methods    base
#>
#> other attached packages:
#> [1] GENOVA_1.0.0      data.table_1.12.8 BiocStyle_2.12.0  rmarkdown_1.15
#>
#> loaded via a namespace (and not attached):
#> [1] Rcpp_1.0.3        plyr_1.8.5       compiler_3.6.2
#> [4] pillar_1.4.3     BiocManager_1.30.4 tools_3.6.2
#> [7] digest_0.6.23    evaluate_0.14    lifecycle_0.1.0
#> [10] tibble_2.1.3    gtable_0.3.0     pkgconfig_2.0.3
#> [13] rlang_0.4.4      RhpcBLASctl_0.18-205 yaml_2.2.0
```

```
#> [16] xfun_0.9           withr_2.1.2       stringr_1.4.0
#> [19] dplyr_0.8.3        knitr_1.24       grid_3.6.2
#> [22] tidyselect_0.2.5    glue_1.3.1       R6_2.4.1
#> [25] bookdown_0.13      tidyverse_0.8.3  ggplot2_3.2.1
#> [28] purrr_0.3.3         farver_2.0.3     reshape2_1.4.3
#> [31] magrittr_1.5         scales_1.1.0     codetools_0.2-16
#> [34] htmltools_0.4.0      assertthat_0.2.1 colorspace_1.4-1
#> [37] bigrwig_0.1.0        labeling_0.3     stringi_1.4.5
#> [40] lazyeval_0.2.2       munsell_0.5.0    crayon_1.3.4
```

## References

- Crane, Emily, Qian Bian, Rachel Patton McCord, Bryan R. Lajoie, Bayly S. Wheeler, Edward J. Ralston, Satoru Uzawa, Job Dekker, and Barbara J. Meyer. 2015. "Condensin-driven remodelling of X chromosome topology during dosage compensation." *Nature* 523 (7559):240–44. <https://doi.org/10.1038/nature14450>.
- de Wit, Elzo, Erica S M Vos, Sjoerd J B Holwerda, Christian Valdes-Quezada, Marjon J A M Verstegen, Hans Teunissen, Erik Splinter, Patrick J. Wijchers, Peter H L Krijger, and Wouter de Laat. 2015. "CTCF Binding Polarity Determines Chromatin Looping." *Molecular Cell* 60 (4):676–84. <https://doi.org/10.1016/j.molcel.2015.09.023>.
- Dixon, Jesse R., Siddarth Selvaraj, Feng Yue, Audrey Kim, Yan Li, Yin Shen, Ming Hu, Jun S. Liu, and Bing Ren. 2012. "Topological domains in mammalian genomes identified by analysis of chromatin interactions." *Nature* 485 (7398). Nature Publishing Group:376–80. <https://doi.org/10.1038/nature11082>.
- Gassler, Johanna, Hugo B Brandão, Maxim Imakaev, Ilya M Flyamer, Sabrina Ladstätter, Wendy A Bickmore, Jan-Michael Peters, Leonid A Mirny, and Kikuë Tachibana. 2017. "A mechanism of cohesin-dependent loop extrusion organizes zygotic genome architecture." *The EMBO Journal*, e201798083. <https://doi.org/10.15252/embj.201798083>.
- Haarhuis, Judith H.I., Robin H. van der Weide, Vincent A Blomen, J Omar Yáñez-Cuna, Mario Amendola, Marjon S. van Ruiten, Peter H.L. Krijger, et al. 2017. "The Cohesin Release Factor WAPL Restricts Chromatin Loop Extension." *Cell* 169 (4):693–707.e14. <https://doi.org/10.1016/j.cell.2017.04.013>.
- Harewood, Louise, Kamal Kishore, Matthew D. Eldridge, Steven Wingett, Danita Pearson, Stefan Schoenfelder, V. Peter Collins, and Peter Fraser. 2017. "Hi-C as a tool for precise detection and characterisation of chromosomal rearrangements and copy number variation in human tumours." *Genome Biology* 18 (1):125. <https://doi.org/10.1186/s13059-017-1253-8>.
- Krijger, Peter H.L., Bruno Di Stefano, Elzo De Wit, Francesco Limone, Chris Van Oevelen, Wouter De Laat, and Thomas Graf. 2016. "Cell-of-origin-specific 3D genome structure acquired during somatic cell reprogramming." *Cell Stem Cell* 18 (5):597–610. <https://doi.org/10.1016/j.stem.2016.01.007>.
- Lévy-Leduc, Celine, M. Delattre, T. Mary-Huard, and S. Robin. 2014. "Two-dimensional segmentation for analyzing Hi-C data." In *Bioinformatics*. Vol. 30. 17. <https://doi.org/10.1093/bioinformatics/btu443>.

## GENOVA: explore the Hi-C's

Lieberman-Aiden, E, and NI Van Berkum. 2009. "Comprehensive mapping of long range interactions reveals folding principles of the human genome." *Science* 326 (5950):289–93.

<https://doi.org/10.1126/science.1181369.Comprehensive>.

Olivares-Chauvet, Pedro, Zohar Mukamel, Aviezer Lifshitz, Omer Schwartzman, Noa Oded Elkayam, Yaniv Lubling, Gintaras Deikus, Robert P. Sebra, and Amos Tanay. 2016.

"Capturing pairwise and multi-way chromosomal conformations using chromosomal walks." *Nature* 540 (7632):296–300. <https://doi.org/10.1038/nature20158>.

Rao, Suhas S P, Miriam H Huntley, Neva C Durand, and Elena K Stamenova. 2014. "A 3D Map of the Human Genome at Kilobase Resolution Reveals Principles of Chromatin Looping." *Cell* 159 (7). Elsevier Inc.:1665–80. <https://doi.org/10.1016/j.cell.2014.11.021>.

Servant, Nicolas, Nelle Varoquaux, Bryan R. Lajoie, Eric Viara, Chong-Jian Chen, Jean-Philippe Vert, Edith Heard, Job Dekker, and Emmanuel Barillot. 2015. "HiC-Pro: an optimized and flexible pipeline for Hi-C data processing." *Genome Biology* 16 (1):259. <https://doi.org/10.1186/s13059-015-0831-x>.

Wit, Elzo de, Erica S M Vos, Sjoerd J B Holwerda, Christian Valdes-Quezada, Marjon J A M Versteegen, Hans Teunissen, Erik Splinter, Patrick J. Wijchers, Peter H L Krijger, and Wouter de Laat. 2015. "CTCF Binding Polarity Determines Chromatin Looping." *Molecular Cell* 60 (4):676–84. <https://doi.org/10.1016/j.molcel.2015.09.023>.

Wutz, Gordana, Csilla Várnai, Kota Nagasaka, David A Cisneros, Roman R Stocsits, Wen Tang, Stefan Schoenfelder, et al. 2017. "Topologically associating domains and chromatin loops depend on cohesin and are regulated by CTCF, WAPL, and PDS5 proteins." *The EMBO Journal*, e201798004. <https://doi.org/10.15252/embj.201798004>.