SimBindProfiles: Similar Binding Profiles, identifies common and unique regions in array genome tiling array data

Bettina Fischer

April 26, 2022

Contents

1 Introduction

SimBindProfiles identifies common and unique binding regions in genome tiling array data. This package does not rely on peak calling, but directly compares binding profiles processed on the same array platform. It implements a simple threshold approach, thus

allowing retrieval of commonly and differentially bound regions between datasets as well as events of compensation and increased binding.

The tool requires each data set in SGR file format, which are tab-delimited files with chromosome, position and signal value columns. We suggest preprocessing two-colour microarray data with *Ringo* [3] and Affymetrix cel files with *Starr* [4], which perform smoothing of probe intensities across replicate arrays for each data set. It is important that data are sorted by chromosomes and ascending positions along each chromosome.

```
> library("SimBindProfiles")
```

2 Reading data and normalisation

We provide three working example data sets in this vignette. These data have been processed on NimbleGen Drosophila melanogaster DM5 Tiling Set HX1 in triplicates and smoothed into window scores using *Ringo*.

In *Drosophila melanogaster*, SoxNeuro (SoxN) and Dichaete (D) belong to the SoxB family of transcription factors. SoxN and Dichaete play essential roles in many aspects of neurogenesis and exhibit some degree of functional redundancy in cells where they are coexpressed. Here, we study the binding patterns of SoxN and Dichaete in wildtype (wt) and SoxN mutant embryos via DamID. In this vignette, we only use the probes located on chromosome X between 1-6000000 bp:
SoxNDam = SoxN binding in wt embryos
SoxN-DDam = Dichaete binding in SoxN mutants
DDam = Dichaete binding in wt embryos

```
> dataPath <- system.file("extdata",package="SimBindProfiles")
> list.files(dataPath, pattern=".txt")
```

```
[1] "DDam_trunc.txt"      "SoxN-DDam_trunc.txt" "SoxNDam_trunc.txt"
```

```
> head(read.delim(paste(dataPath, "SoxNDam_trunc.txt", sep="/"),
+         header=FALSE, nrows=5))
```

```
    V1  V2        V3
1 chrX 116 -1.218227
2 chrX 181 -1.116333
3 chrX 236 -1.081980
4 chrX 276 -1.116333
5 chrX 331 -1.116333
```

To read data into an *ExpressionSet* object each file name is specified omitting the .txt extension. While reading the files, data is quantile normalised as per *limma* [2]. For more details on *ExpressionSet* please refer to "An Introduction to Bioconductor's ExpressionSet Class" [1].

```
> readTestSGR <- readSgrFiles(X=c("SoxNDam_trunc", "SoxN-DDam_trunc",
+         "DDam_trunc"), dataPath)
```

We continue in this vignette using the larger chromosome X probe set data which was previously saved as SGR.Rdata object and can be loaded and viewed.

```
> dataPath <- system.file("data",package="SimBindProfiles")
> load(paste(dataPath, "SGR.RData", sep="/"))
> print(SGR)
```

```
ExpressionSet (storageMode: lockedEnvironment)
assayData: 98715 features, 3 samples
  element names: exprs
protocolData: none
phenoData
  sampleNames: SoxNDam SoxN-DDam DDam
  varLabels: FileName tiling array
  varMetadata: varLabel labelDescription
featureData
  featureNames: 1 2 ... 98715 (98715 total)
  fvarLabels: PROBE_ID CHR START
  fvarMetadata: labelDescription
experimentData: use 'experimentData(object)'
Annotation:
```

A useful plot comparing all the data sets and their correlation after normalisation can be created with the `eSetScatterPlot` function (Figure 1).

```
> eSetScatterPlot(SGR)
```

3 Determining a bound cut-off and a difference cut-off

In order to identify probes or regions, which are bound similarly or differentially bound between the data sets, `bound.cutoff` and `diff.cutoff` (difference cut-off) thresholds have to be chosen.

We implemented two methods to set the bound cut-off, probes above this threshold are considered "bound". The `twoGaussiansNull` method established in the *Ringo* package

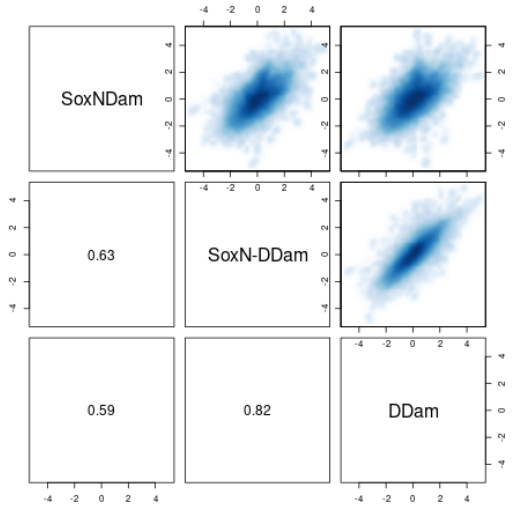

Figure 1: *Smoothed scatterplots of normalised signals and the corresponding correlations.*

[3], in which data is assumed to follow a mixture of two Gaussian distributions. The Gaussian with the lower mean value is assumed to be the null distribution and probe levels are assigned p-values based on this null distribution. Alternatively the user can select the `normalNull` method which assumes the null distribution is normal and symmetrical around the mode or zero. For both methods the user can decide if the resulting p-values are to be adjusted for multiple testing (fdr) or select a p-value threshold.

In our example we use the twoGaussiansNull at 25% FDR, the function also provides a QC plot of the two Gaussians curves and an optional p-value histogram (not shown).

```
> bound.cutoff <- findBoundCutoff(SGR, method="twoGaussiansNull", fdr=0.25)
```

```
Using bound.cutoff = 2.05
```

To show the `bound.cutoff` in relation to the data one can plot a histogram of the data with the bound cutoff (Figure 2).

```
> hist(exprs(SGR)[,1], breaks=1000, freq=FALSE, border="grey",
+         main=sampleNames(SGR)[1], xlab="signal",
+         sub=paste("bound.cutoff =", bound.cutoff, sep=" "))
> abline(v=bound.cutoff, col="red", lty=3, lwd=2)
```

A probe is considered uniquely bound in one data set if it is bound above the `diff.cutoff` threshold to the other set. We propose that the difference cut-off should be smaller than the bound cut-off, but for more stringent analysis criteria it can also be set to the same value as the `bound.cutoff`.

In our example we used the diff.cutoff as 75% of the `bound.cutoff`

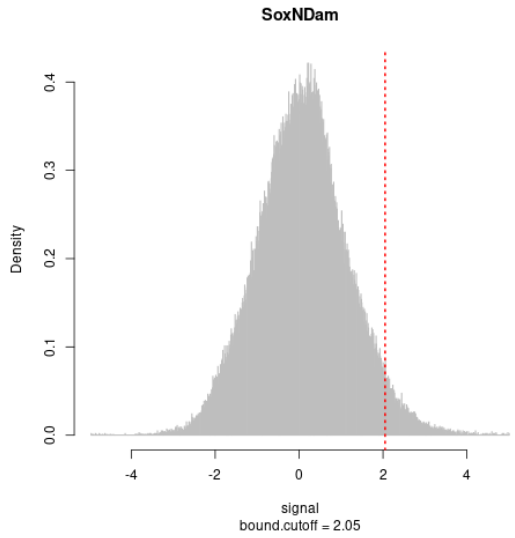

Figure 2: *Histogram of the signal of SoxNDam. Probes with a signal above the bound.cutoff threshold are considered "bound".*

```
> diff.cutoff <- round(bound.cutoff * 0.75,2)
```

We can plot the probe intensities along parts of the chromosome using the `chipAlongChrom` function from *Ringo*, which can be called via the plot command. First we create a *probeAnno* class object which contains the mapping between the probes and their genomic positions and uses the information stored in the *ExpressionSet* object and requires the probe length of the oligo on the array. In Figure 3 SoxNDam (green curve) is uniquely bound at region 2324000 - 2326000 as the intensity is above the bound.cutoff. Whereas SoxN-DDam (orange) is also above the bound.cutoff at region 2334000 - 2336000 but the difference in the intensity of SoxNDam is too small (below diff.cutoff) to call this region uniquely bound.

```
> probeAnno <- probeAnnoFromESet(SGR, probeLength=50)
```

```
> plot(SGR, probeAnno, chrom="X", xlim=c(2323000,2337000), ylim=c(-3,4),
+         samples=c(1,2))
```

4 Setting array specific parameters

The user has to specify the `probes` and `probe.max.spacing` parameters. Both depend on the array platform and how densely the probes are spaced across the genome on the tiling array. The minimum number of probes specifies how many probes have to be in a valid region. The probe.max.spacing is the maximum distance in base pairs allowed between probes before a region is split into separate regions.

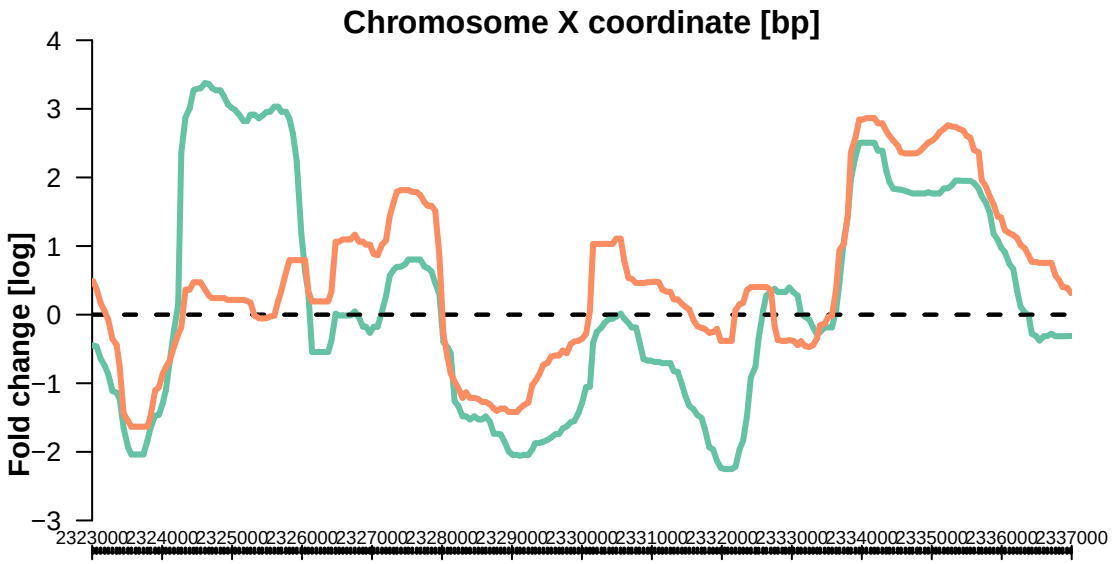

Figure 3: *Probe intensities along chromosome, green = SoxNDam, orange = SoxNDDam.*

```
> probes <- 10
> probe.max.spacing <- 200
```

A frequency plot of number of probes per bound regions can be used to help determine the probes parameter selection

```
> probeLengthPlot(SGR, sgrset=1, chr=NULL, bound.cutoff, probe.max.spacing=200,
+         xlim.max=25)
```

5 Performing pairwise classification

In this section we identify regions that are uniquely or commonly bound in two data sets. First the probes for each data set are flagged as bound or not bound depending on whether the signal is above the `bound.cutoff`. Then the probes are split into three classes:
Class 1: uniquely bound in set 1 (bound in set 1, not bound in set 2, signal 1 minus signal 2 above diff.cutoff).
Class 2: uniquely bound in set 2 (bound in set 2, not bound in set 1, signal 2 minus signal 1 above diff.cutoff)
Class 3: bound in both data sets
Then the classified probes are filtered into regions using the `probes` and `probe.max.spacing` parameters. The regions for each class are exported to bed files, which provide the chromosome, start and end positions, the name and a score for the region. For example the score for class 1 is calculated as follows. First we subtract for each probe within a region

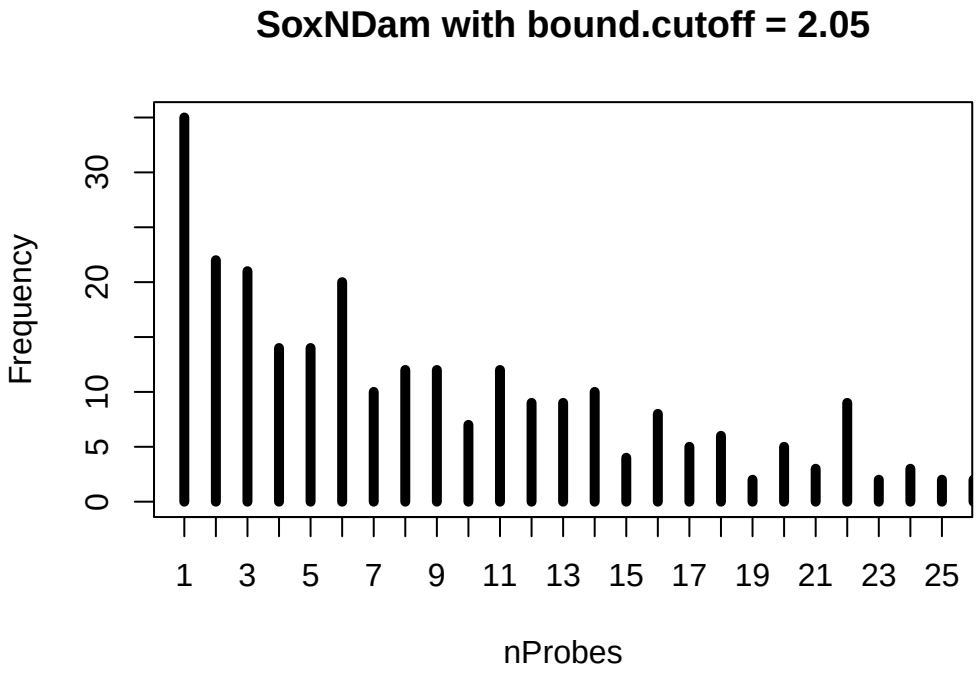

Figure 4: *Frequency plot of number of probes per bound region.*

signal set 1 minus signal set 2, and then we calculate the mean over the region. The tool uses the names of each data set and the selected parameters as the resulting file names (b = bound.cutoff, d = diff.cutoff, v = probes, g = probe.max.spacing).

In our example we query SoxNDam vs. DDam, which correspond to data set 1 and 3 in the ExpessionSet object.

```
> pairwiseR <- pairwiseRegions(SGR, sgrset=c(1,3), bound.cutoff,
+         diff.cutoff, probes, probe.max.spacing)

Pairwise comparison of SoxNDam vs DDam

Filter data into regions...
Writing SoxNDam.vs.DDam.unique_b2.05d1.54v10g200.bed ,regions = 67 ...
Writing DDam.vs.SoxNDam.unique_b2.05d1.54v10g200.bed ,regions = 51 ...
Writing SoxNDam.DDam.common_b2.05d1.54v10g200.bed ,regions = 43 ...

> head(pairwiseR)

            name class.group  chr  start    end    score nProbes
1 SoxNDam.unique           1 chrX 424668 426418 3.142840      33
2 SoxNDam.unique           1 chrX 436708 437593 2.278542      17
```

```
3 SoxNDam.unique            1 chrX 761106 762136 2.255283       20
4 SoxNDam.unique            1 chrX 789430 792565 3.177630       58
5 SoxNDam.unique            1 chrX 900082 900792 1.770974       14
6 SoxNDam.unique            1 chrX 901237 901727 1.896931       10
```

We also provide a ploting method to show the bound probes (before filtering into regions) in colour.

```
> plotBoundProbes(SGR, sgrset=c(1,2), method="pairwise", bound.cutoff,
+         diff.cutoff)
```

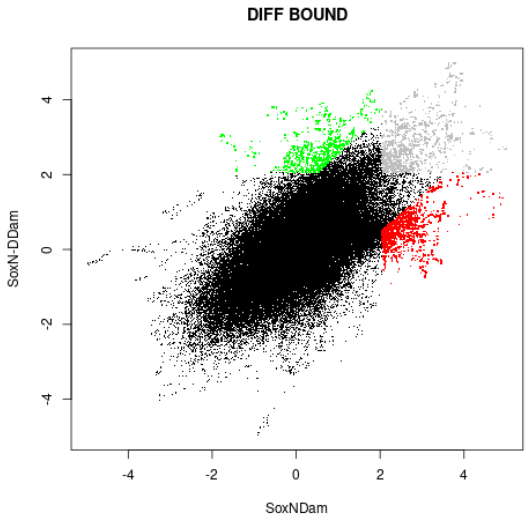

Figure 5: *Scatterplot of pairwise classification of SoxNDam vs. DDam. Red highlights the unique SoxNDam probes, green unique in DDam and grey are probes common to both.*

6 Performing three-way classification

This tool allows identification of regions that are unique or common in three data sets. The approach is the same as for the pairwise classification. The probes are segregated into seven classes:
Class = 1: unique probes in set 1
Class = 2: unique probes in set 2
Class = 3: unique probes in set 3
Class = 4: common probes in set 1+2
Class = 5: common probes in set 2+3
Class = 6: common probes in set 1+3
Class = 7: common probes in set 1+2+3
Then the probes are again filtered into regions using the `probes` and `probe.max.spacing` parameters. The regions for each class are exported to bed files. The score is calculated similar to the pair-wise classification.

In our example we query SoxNDam vs. SoxN-DDam vs. DDam (results not shown).

```
> threewayR <- threewayRegions(SGR, sgrset=c(1,2,3), bound.cutoff,
+         diff.cutoff, probes, probe.max.spacing)
```

7 Performing increased binding classification

It might be of interest to identify regions showing increased binding, which are more bound in one dataset compared to the other. In our example, Dichaete can bind at a higher level in the SoxN mutant embryos (SoxN-DDam) compared to the wt embryos (DDam) (Figure 6). If the signal of a bound probe in set 1 is higher than the diff.cutoff, then these probes are filtered into regions using the `probes` and `probe.max.spacing` parameters and reported as bed file, which reports the chromosome, start, end, name and score. The score is calculated similarly to the pairwise classification.

Compare set SoxN-DDam vs. DDam

```
> increasedR <- increasedBindingRegions(SGR, sgrset=c(2,3), bound.cutoff, diff.cutoff,
+         probes, probe.max.spacing)
> head(increasedR)
```

```
                        name class.group  chr   start     end    score nProbes
1 SoxN-DDam.increasedBinding           1 chrX 2217665 2218265 2.352285      12
2 SoxN-DDam.increasedBinding           1 chrX 4590918 4591908 1.671484      18
```

Visualise the second increased binding region in genomic context.

```
> plot(SGR, probeAnno, chrom="X", xlim=c(4589000,4593200), ylim=c(-0.5,5),
+         samples=c(2,3))
```

To plot the probes showing increased binding (before filtering into regions) in colour (Figure 7).

```
> plotBoundProbes(SGR, sgrset=c(2,3), method="increasedBinding", bound.cutoff,
+         diff.cutoff, pcex=4)
```

8 Performing compensation classification

This is another special case in which we want to identify regions which are bound in two sets but not in the third. In our example, in wt embryos Dichaete is not bound (DDam)

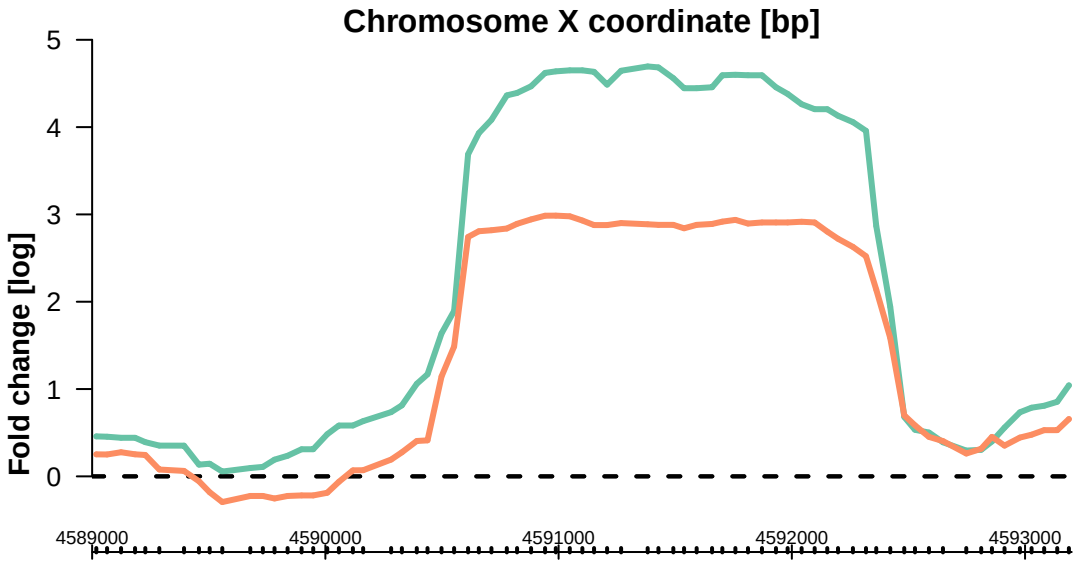

Figure 6: *Probe intensities along chromosome at increased binding region, green = SoxN-DDam, orange = DDam.*

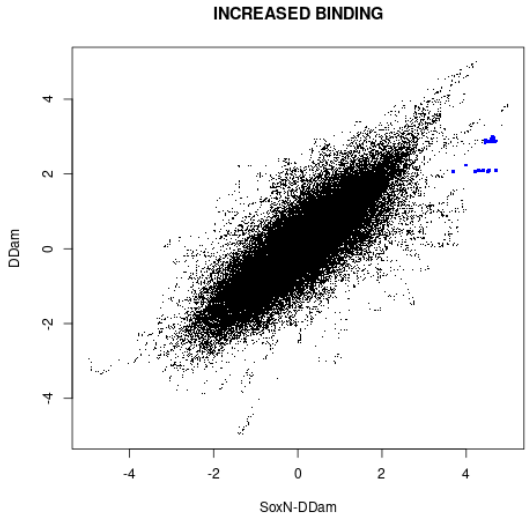

Figure 7: *Scatterplot of increased binding classification of SoxNDam vs. DDam, increased binding probes are highlighted in blue*

and SoxN is bound (SoxNDam). However, in SoxN mutants (SoxN-DDam) Dichaete binds at this location to compensate for the loss of SoxN (Figure 8). Probes above the bound.cutoff for which the average bound signal of set 1 and set 2 are larger than the diff.cutoff to the non-bound set 3 are identified. The probes are again filtered into regions using the `probes` and `probe.max.spacing` parameters and reported in a bed file, which gives the chromosome, start, end, name and score.

Compare set SoxNDam + SoxN-DDam vs. DDam

```
> compR <- compensationRegions(SGR, sgrset=c(1,2,3), bound.cutoff,
+         diff.cutoff, probes, probe.max.spacing)
> head(compR)
```

```
                                    name class.group  chr   start     end
1 SoxNDam.SoxN-DDam.vs.DDam.compensation           1 chrX 1512681 1513656
2 SoxNDam.SoxN-DDam.vs.DDam.compensation           1 chrX 2411543 2413073
3 SoxNDam.SoxN-DDam.vs.DDam.compensation           1 chrX 2456797 2457742
4 SoxNDam.SoxN-DDam.vs.DDam.compensation           1 chrX 2944219 2945314
5 SoxNDam.SoxN-DDam.vs.DDam.compensation           1 chrX 3616693 3617308
     score nProbes
1 1.900121      19
2 2.508937      29
3 2.880944      18
4 2.701074      21
5 2.243366      12
```

Visualise a compensation region in genomic context (Figure 8).

```
> plot(SGR, probeAnno, chrom="X", xlim=c(2943000,2947000), ylim=c(-1,4))
```

To plot the probes showing compensation (before filtering into regions) in colour (Figure 9).

```
> plotBoundProbes(SGR, sgrset=c(1,2,3), method="compensation", bound.cutoff,
+         diff.cutoff, pcex=4)
```

9 Concluding Remarks

The package *SimBindProfiles* facilitates the comparison of ChIP-chip or DamID profiles generated on the same microarray platform. It provides functions for data import, normalization and analysis. High-level plots for quality assessment are available. While this analysis approach worked well with our data, we do not claim it is the definite algorithm for this task.

This vignette was generated using the following package versions:

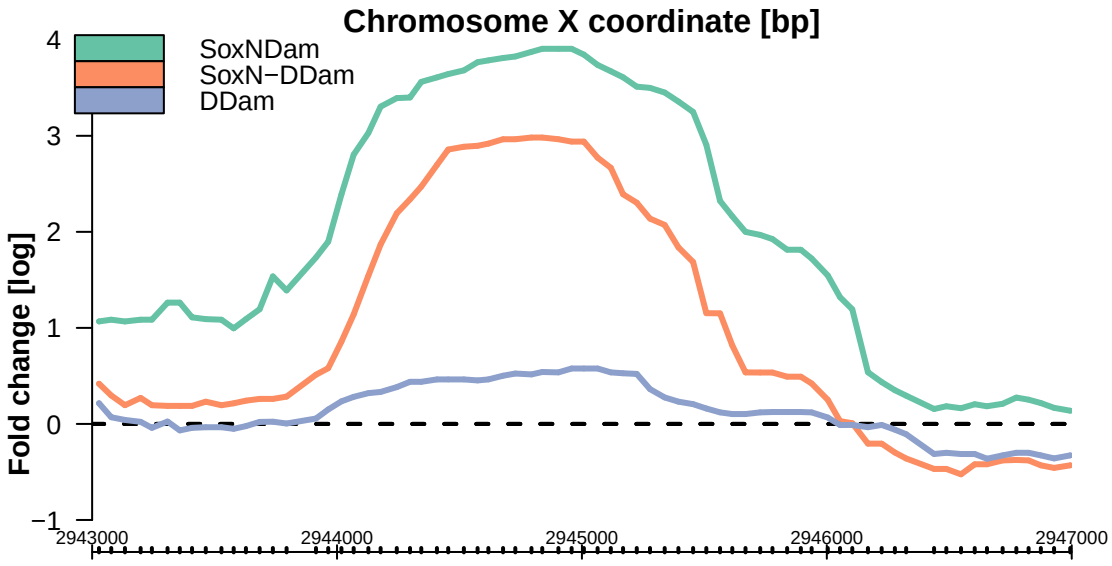

Figure 8: *Probe intensities along chromosome, green = SoxN-DDam, orange = SoxN-DDam, blue = DDam.*

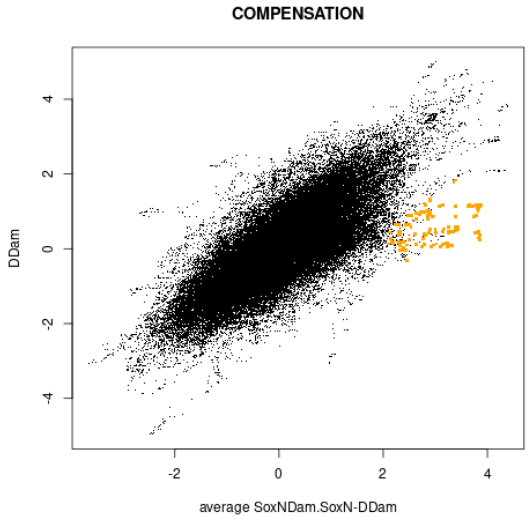

Figure 9: *Scatterplot of compensation classification of SoxNDam + SoxN-DDam vs. DDam. Probes which are bound in SoxNDam + SoxN-DDam but not in DDAm are highlighted in orange.*

- R version 4.2.0 RC (2022-04-19 r82224), `x86_64-pc-linux-gnu`
- Locale: `LC_CTYPE=en_US.UTF-8`, `LC_NUMERIC=C`, `LC_TIME=en_GB`, `LC_COLLATE=C`, `LC_MONETARY=en_US.UTF-8`, `LC_MESSAGES=en_US.UTF-8`, `LC_PAPER=en_US.UTF-8`, `LC_NAME=C`, `LC_ADDRESS=C`, `LC_TELEPHONE=C`, `LC_MEASUREMENT=en_US.UTF-8`, `LC_IDENTIFICATION=C`
- Running under: `Ubuntu 20.04.4 LTS`
- Matrix products: default
- BLAS: `/home/biocbuild/bbs-3.15-bioc/R/lib/libRblas.so`
- LAPACK: `/home/biocbuild/bbs-3.15-bioc/R/lib/libRlapack.so`
- Base packages: base, datasets, grDevices, graphics, grid, methods, stats, utils
- Other packages: Biobase 2.56.0, BiocGenerics 0.42.0, Matrix 1.4-1, RColorBrewer 1.1-3, Ringo 1.60.0, SimBindProfiles 1.34.0, lattice 0.20-45, limma 3.52.0
- Loaded via a namespace (and not attached): AnnotationDbi 1.58.0, BiocManager 1.30.17, Biostrings 2.64.0, DBI 1.1.2, GenomeInfoDb 1.32.0, GenomeInfoDbData 1.2.8, IRanges 2.30.0, KEGGREST 1.36.0, KernSmooth 2.23-20, R6 2.5.1, RCurl 1.98-1.6, RSQLite 2.2.12, Rcpp 1.0.8.3, S4Vectors 0.34.0, XML 3.99-0.9, XVector 0.36.0, affy 1.74.0, affyio 1.66.0, annotate 1.74.0, assertthat 0.2.1, bit 4.0.4, bit64 4.0.5, bitops 1.0-7, blob 1.2.3, cachem 1.0.6, cli 3.3.0, colorspace 2.0-3, compiler 4.2.0, crayon 1.5.1, dplyr 1.0.8, ellipsis 0.3.2, fansi 1.0.3, fastmap 1.1.0, genefilter 1.78.0, generics 0.1.2, ggplot2 3.3.5, glue 1.6.2, gtable 0.3.0, httr 1.4.2, lifecycle 1.0.1, magrittr 2.0.3, mclust 5.4.9, memoise 2.0.1, munsell 0.5.0, pillar 1.7.0, pkgconfig 2.0.3, png 0.1-7, preprocessCore 1.58.0, purrr 0.3.4, rlang 1.0.2, scales 1.2.0, splines 4.2.0, stats4 4.2.0, survival 3.3-1, tibble 3.1.6, tidyselect 1.1.2, tools 4.2.0, utf8 1.2.2, vctrs 0.4.1, vsn 3.64.0, xtable 1.8-4, zlibbioc 1.42.0

Acknowledgments

Many thanks to Enrico Ferrero who generated the data sets and our conductive discussions about the analysis approach, Steven Russell for helpful suggestions and Robert Stojnic for source code contributions to SimBindProfiles.

References

[1] S. Falcon, M. Morgan, and R. Gentleman. *An Introduction to Bioconductor's ExpressionSet Class*. http://wiki.biostat.berkeley.edu/~bullard/courses/Tmexico-08/resources/ExpressionSetIntroduction.pdf, February 2007.

[2] G. K. Smyth. Limma: linear models for microarray data. In R. Gentleman, V. Carey, W. Huber, R. Irizarry, and S. Dudoit, editors, *Bioinformatics and Computational Biology Solutions Using R and Bioconductor*, pages 397–420. Springer, 2005.

[3] J. Toedling, O. Sklyar, T. Krueger, J. J. Fischer, S. Sperling, and W. Huber. Ringo - an R/Bioconductor package for analyzing ChIP-chip readouts. *BMC Bioinformatics*, 8:221, 2007.

[4] B. Zacher, J. Soeding, P. F. Kuan, M. Siebert, and A. Tresch. *Starr: Simple tiling array analysis of Affymetrix ChIP-chip data*, 2009. R package version 1.14.1.