Chicdiff Vignette

Jonathan Cairns, William Orchard, Valeriya Malysheva, Mikhail Spivakov

23 January, 2019

Contents

1 Introduction

Chicdiff is a method for detecting statistically significant differential interactions in Capture Hi-C data. This vignette will walk you through a typical Chicdiff analysis.

The methodology behind Chicdiff is presented in this preprint. Briefly, Chicdiff combines moderated differential testing for count data implemented in DESeq2 with CHi-C-specific procedures for signal normalisation and p-value weighting. To increase power, Chicdiff also combines reads across several fragments surrounding each significantly interacting region for each bait. More specifically, Chicdiff uses background estimates from Chicago to construct a custom scaling matrix that is used in differential testing with DESeq2, and the Wald test p-values from DESeq2 are submitted to the weighting procedure as described below.

Due to strong distance-dependent variation of signal-to-noise ratios and effect sizes in CHi-C data, Chicdiff performs non-uniform multiple testing correction, such that weaker signals at longer distances are corrected more vigorously. It uses the IHW package to learn the weights on the p-values using interaction distance as a covariate such that the overall fraction of rejected null hypotheses is maximised. Since sets of interactions submitted for Chicdiff testing typically have non-uniform Chicdiff p-value distributions due to selection bias, interactions from a 'training set' sampled from the entire Capture Hi-C dataset are used for weight training. The weights learned this way then are applied to the p-values in the test set, and the resulting weighted p-values are adjusted for multiple testing and reported to the user.

In this vignette, we use a small subset of the Capture Hi-C data for naive CD4+ T cells and Monocytes data from Javierre et al. (Cell 2016) provided in package *ChicdiffData*:

```
library(Chicdiff)
```

```
library(ChicdiffData)
```

WARNING The dataset in *ChicdiffData* package contains only data for chromosome 19 from two replicates of monocytes and naive CD4+ T cells (compared with three and four replicates, respectively, released in the paper).

The example dataset from *ChicdiffData* uses ~0.6 Gb RAM and takes minutes to process on a standard laptop machine. A typical job for two biological replicates of genome-wide CHi-C data in each condition takes ~30-60 min and uses ~10-15 Gb RAM. High-coverage datasets with more replicates will take even longer to process and use more RAM.

2 Workflow

2.1 Input files required

Before you start, you will need:

1. Two restriction map information files ("design files"):

- Restriction map file (.rmap) - a bed file containing coordinates of the restriction fragments. By default, 4 columns: chr, start, end, fragmentID.
- Bait map file (.baitmap) - a bed file containing coordinates of the baited restriction fragments, and their associated annotations. By default, 5 columns: chr, start, end, fragmentID, baitAnnotation. The regions specified in this file, including their fragmentIDs, must be an exact subset of those in the .rmap file. The baitAnnotation is a text field that is used only to annotate the output and plots.

We recommend that you put both of these files into the same directory (that is referred to as designDir). An example of a valid design folder, for a sample of the CD4+ and Monocytes data used in this vignette, is provided in the ChicdiffData package, and can be accessed as follows.

```
dataPath <- system.file("extdata", package="ChicdiffData")
testDesignDir <- file.path(dataPath, "designDir")
dir(testDesignDir)
```

```
## [1] "chr19_GRCh37_HindIII.baitmap" "chr19_GRCh37_HindIII.rmap"
```

NOTE These are same design files that are used to run Chicago.

2. One or more *peakfile(s)* defining interactions of interest. Typically *peakfile(s)* are generated by `chicagoTools/makePeakMatrix.R` from the results of Chicago runs on either individual replicates or merged replicates for each condition. The sample IDs from the *peakfiles* will be used as primary identifiers for all other types of input files (see below). If multiple *peakfiles* are given, they will be merged by Chicdiff.

An example *peakfile* (replicate-level) is provided in the ChicdiffData package:

```
peakFiles <- file.path(dataPath,"peakMatrix_cd4_mono_unmerged.txt")
```

3. Count data files in Chicago input format, *.chinput*. You can obtain *.chinput* files from aligned Capture Hi-C BAM files by running `chicagoTools/bam2chicago.sh`. (To obtain BAM files from raw fastq files, you will need to use a Hi-C alignment & QC pipeline such as HiCUP.

Example *.chinput* files are provided in the ChicdiffData package, as follows:

```
testDataPath_CD4 <- file.path(dataPath, "CD4")
testDataPath_Mono <- file.path(dataPath, "monocytes")
dir(testDataPath_CD4)
```

```
## [1] "unitTest_CD41.chinput" "unitTest_CD42.chinput"
```

```
dir(testDataPath_Mono)
```

```
## [1] "unitTest_Mono2.chinput" "unitTest_Mono3.chinput"
```

Information about *.chinput* files per condidtion needs to be presented to Chicdiff in the follwing way:

```
countData <- list(
    CD4 = c(NCD4_22 = file.path(testDataPath_CD4, "unitTest_CD41.chinput")
,
            NCD4_23 = file.path(testDataPath_CD4, "unitTest_CD42.chinput")
            ),

    Mono = c(Mon_2 = file.path(testDataPath_Mono, "unitTest_Mono2.chinput"
),
            Mon_3 = file.path(testDataPath_Mono, "unitTest_Mono3.chinput")
            )
  )
```

NOTE The names of every file should match the column name of the correpsonding data set in the *peakfile*

IMPORTANT If Peakfile(s) were generated at the sample level (as opposed to per replicate), the elements in each of the vectors should not be named.

4. Chicago output files for each separate replicate (saved as either *.Rds* or *.Rda* images). These files contain information about expected read counts per interaction that will be used in the normalisation procedure.

Example *.Rda* files are provided in the ChicdiffData package, as follows:

```
testDataPath_rda <- system.file("data", package="ChicdiffData")
dir(testDataPath_rda)
```

```
## [1] "unitTest_CD41.RDa"  "unitTest_CD42.RDa"  "unitTest_Mono2.RDa"
## [4] "unitTest_Mono3.RDa"
```

To run Chicdff *.Rda* files have to be presented in the follwing way:

```
chicagoData <- list(
  CD4 = c(NCD4_22 = file.path(testDataPath_rda, "unitTest_CD41.RDa"),
          NCD4_23 = file.path(testDataPath_rda, "unitTest_CD42.RDa")
          ),

  Mono = c(Mon_2 = file.path(testDataPath_rda, "unitTest_Mono2.RDa"),
           Mon_3 = file.path(testDataPath_rda, "unitTest_Mono3.RDa")
  )
```

```
)
```

NOTE As for *countData*, the names of every file should match the column name of the correpsonding data set in the *peakfile* file.

IMPORTANT If Peakfile(s) were generated at the sample level (as opposed to per replicate), the elements in each of the vectors should not be named.

2.2 Example workflow

We run Chicdiff on the test data as follows. First, we set up chicdiff experiment settings with `setChicdiffExperiment()`:

```
chicdiff.settings <- setChicdiffExperiment(designDir = testDesignDir, chic
agoData = chicagoData, countData = countData, peakfiles = peakFiles, outpr
efix="test")
```

Note that in addition to returning the settings list, the function will save it as an Rds file ***_settings.Rds*** that can be reused.

OPTIONAL: You can modify settings directly in the command line (see `?defaultChicdiffSettings`). For example, we can switch the running mode to parallel (which speeds up the runtime, but increases the memory requirements):

```
chicdiff.settings <- setChicdiffExperiment(designDir = testDesignDir, chic
agoData = chicagoData, countData = countData, peakfiles = peakFiles, outpr
efix="test", settings = list(parallel=TRUE))
```

Alternatively, you can provide custom parameters in a *settingsFile*:

```
settingsFile = file.path(dataPath,"SettingsFile.txt")
chicdiff.settings <- setChicdiffExperiment(designDir = testDesignDir, chic
agoData = chicagoData, countData = countData, peakfiles = peakFiles, outpr
efix="test", settingsFile = settingsFile)
```

NOTE Settings provided this way override settings from settingsFile. Both override default settings.

Finally, we run the pipeline with `chicdiffPipeline()`:

```
output <- chicdiffPipeline(chicdiff.settings)
```

The pipeline will define extended regions based on the `RUextend` option (by default, 5 fragments each way from each interaction peak at the other end of the bated interactions), perform normalisation, differential testing, weights estimation based on the 'test set' and p-value weighting and multiple testing correction.

For each bait-region interaction, the output data table lists the log2FoldChange of normalised interaction read counts, along with raw, weighted, and weighted adjusted p-values (`pvalue`, `weighted_pvalue` and `weighted_padj`, respectively).

```
head(output)
```

```
##    group baseMean log2FoldChange     lfcSE      stat     pvalue      pa
```

```
dj
## 1:     1 98.04145      0.4654394 0.3430257 1.3568645 0.17482427 0.42874
43
## 2:     1 99.90272      0.5800085 0.3265890 1.7759582 0.07573981 0.27085
60
## 3:     1 58.45174      0.4928256 0.4360129 1.1303006 0.25834959 0.52318
48
## 4:     1 61.89843      0.5010415 0.4122627 1.2153453 0.22423442 0.48726
60
## 5:     1 60.04236      0.5523034 0.4108116 1.3444201 0.17881258 0.43406
86
## 6:     1 35.74176      0.3311124 0.4737406 0.6989319 0.48459458 0.71690
37
##    baitID  maxOE  minOE regionID OEchr OEstart  OEend baitchr baitstart
## 1: 320526 320524 320517      100    19  436227 524426      19    530387
## 2: 320526 320536 320528      101    19  542386 622492      19    530387
## 3: 320526 320544 320534      102    19  594189 749359      19    530387
## 4: 320526 320545 320535      103    19  596117 753972      19    530387
## 5: 320526 320546 320536      104    19  598270 763809      19    530387
## 6: 320526 320550 320540      105    19  638626 864727      19    530387
##    baitend    avDist     uniform      shuff avgLogDist avWeights weight
## 1:  539467 -53060.62 0.009665534 0.05048031   10.87919  7.148253 2.8532
## 2:  539467  45032.11 0.430116629 0.44131718   10.71513  7.148253 2.8532
## 3:  539467 111436.45 0.092592717 0.06689947   11.62121  7.148253 2.8532
## 4:  539467 125665.00 0.303875251 0.02621149   11.74137  7.148253 2.8532
## 5:  539467 140364.82 0.067529660 0.19320485   11.85200  7.148253 2.8532
## 6:  539467 214648.36 0.163497301 0.10922867   12.27676  7.148253 2.8532
##    weighted_pvalue weighted_padj
## 1:      0.06127305     0.2951175
## 2:      0.02654557     0.1621728
## 3:      0.09054731     0.3850313
## 4:      0.07859050     0.3491772
## 5:      0.06267089     0.2997665
## 6:      0.16984248     0.5832029
```

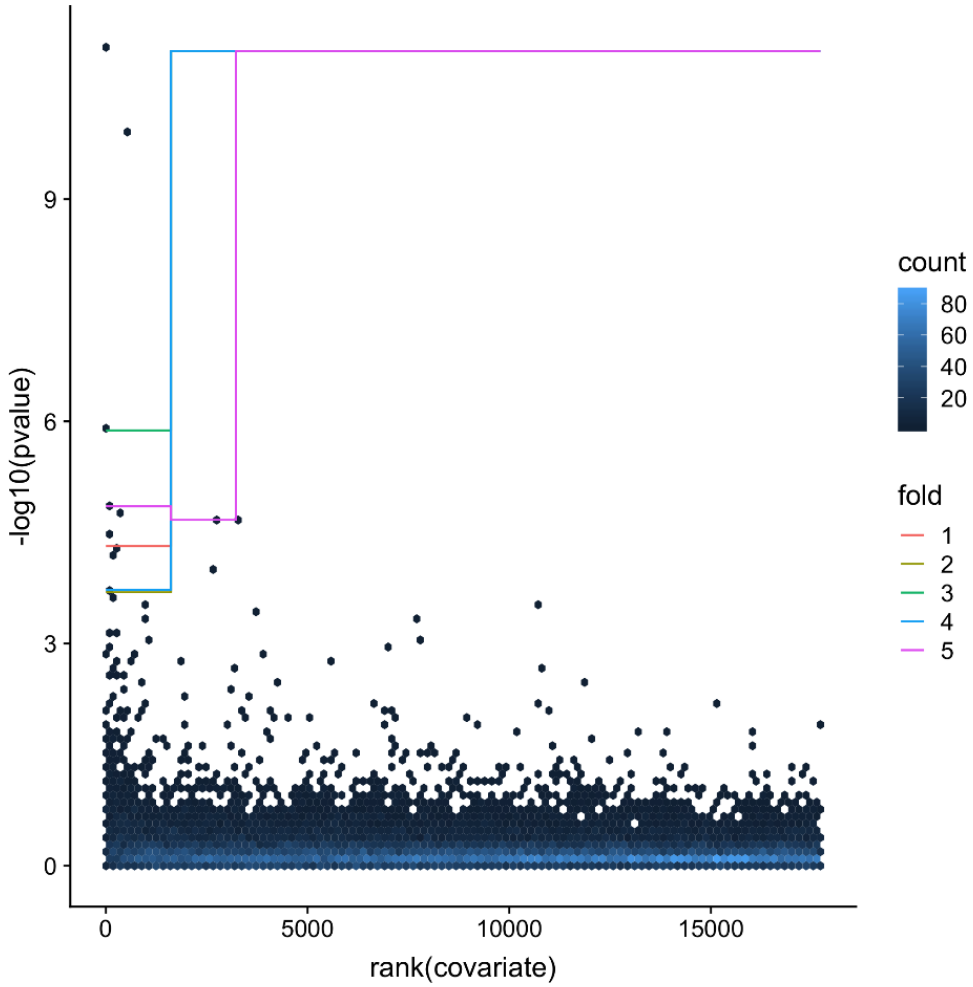
In addition to returning the output data table, `chicdiffPipeline` will save it as an Rds file _results.Rds. It will also save the final count table along with Chicago parameter estimates for each interaction in the Rds file *_countput.Rds*. Further output files can be generated with the setting `saveAuxFiles=TRUE` - see the `?chicdiffpipeline` for details.

2.3 Output diagnostic plots

`chicdiffPipeline()` produces several plots that are saved in the working directory. The plots produced for the chr19 datasets are provided as part of the *ChicdiffData* package. Note that with genome-wide data, the trends described below should be more pronounced.

2.3.1 Decision boundary

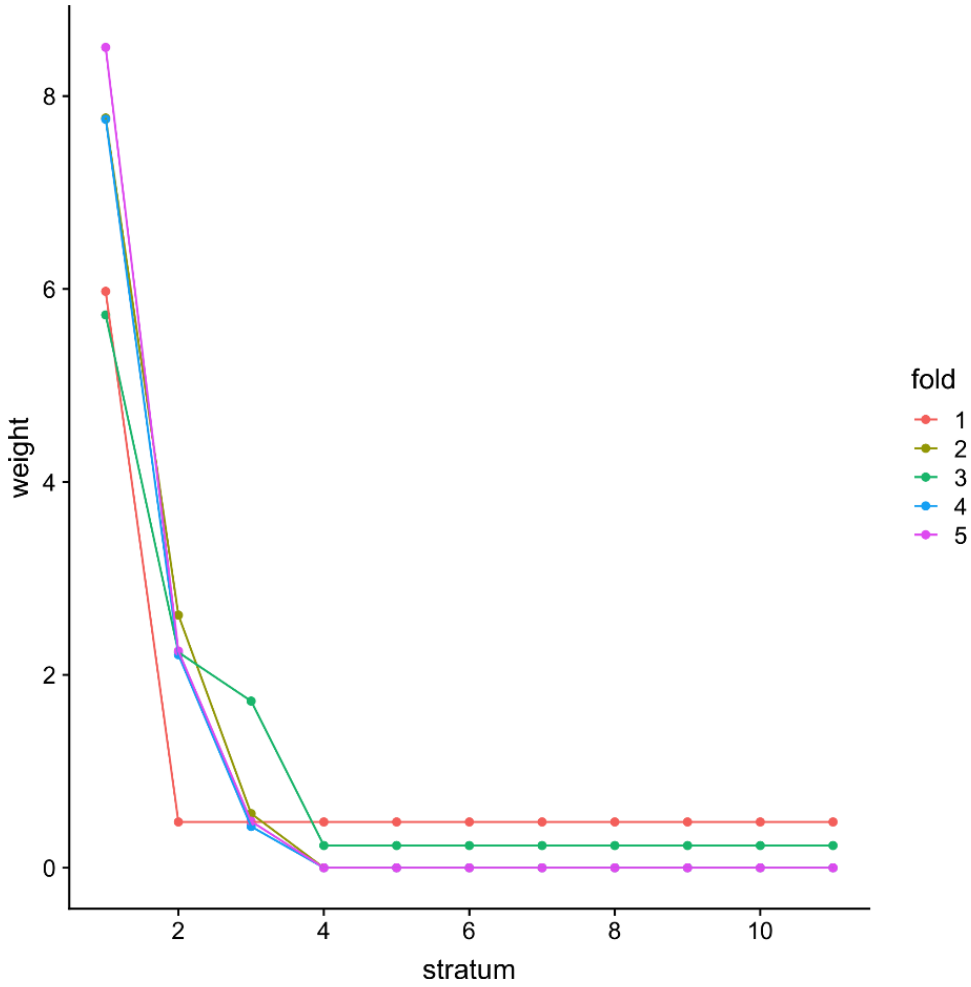
```
resultsPath <- file.path(dataPath, "CD4_Mono_results")

IHWdecisionBoundaryPlot <- png::readPNG(file.path(resultsPath,"test_IHWdec
isionBoundaryPlot.png"))
grid::grid.raster(IHWdecisionBoundaryPlot)
```



This plot shows the implied decision boundaries for the unweighted p-values, as a function of the covariate. The dependence of the boundaries on the covariate (distance) demonstrates that this covariate is informative. The trend is for low p-values to be enriched at low distances.

2.3.2 Estimated weights

```
IHWweightPlot <- png::readPNG(file.path(resultsPath,"test_IHWweightPlot.pn
g"))
grid::grid.raster(IHWweightPlot)
```



It can be seen from this plot that the weights show a dependence on distance (stratum). As expected, the weight functions calculated on different random subsets (folds) of the data behave similarly. For the chr19 data at hand, interactions across longer distances get penalized and are assigned with a lower weight, while interactions over shorter distances are prioritized.

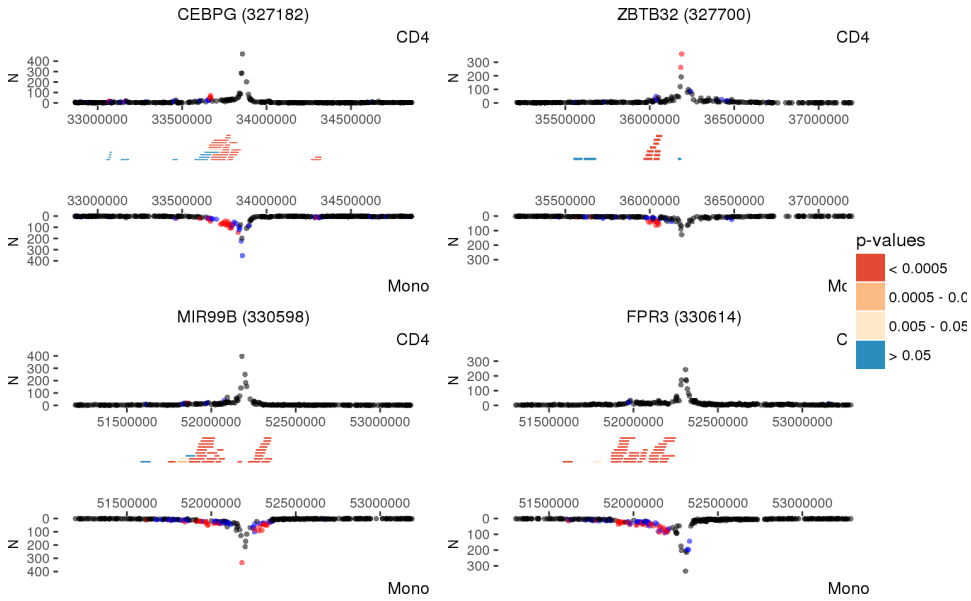
3 Vizualising differential interactions

The `plotdiffBaits()` function can be used to plot the raw read counts of interactions versus their linear distance from the respective bait fragment, as mirror images for the two compared conditions. The extended regions corresponding to the other ends of significant interactions are represented as intervals that are colour-coded according to their adjusted weighted pvalue.

Chicdiff pipeline automatically invokes `plotdiffBaits()` to generate the profiles for four random baits selected from the top 100 baits containing interactions with the lowest weighted p-value. Interactions within 1 Mb from bait are shown, and the datapoints on the plot are colour-coded according to the Chicago score of the respective interactions detected in each of the conditions separately (score>5: red; 3

To use `plotDiffBaits()` outside of the pipeline to plot the profiles of baits of interest, it needs to be provided with the output data table (saved in test_results.Rds by `chicdiffPipeline`, assuming that `outprefix="test"` by default), count table (saved in test_countput.Rds), baitmap file (from *Chicago* design directory) and a vector of baitIDs of interest (see `?plotDiffBaits` for the description of additional parameters).

```
outputRds <- file.path(resultsPath, "test_results.Rds")
countputRds <- file.path(resultsPath, "test_countput.Rds")
bmapRds <- file.path(testDesignDir, "chr19_GRCh37_HindIII.baitmap")

baits <- c(327182, 330614,330598, 327700)
plotDiffBaits(outputRds, countputRds, bmapRds, baits)
```



4 Prioritising individual fragments within differentially interacting regions

Unless the `RUexpand` setting is set to zero, Chicdiff works at the level of interactions between baits and pooled regions, containing multiple restriction fragments.

To obtain indications of differential signals at individual fragments, Chicdiff provides the function `getCandidateInteractions()`. For fragments falling within multiple pooled regions, this function combines their differential p-values by either taking the minimum (default) or, more formally, using the harmonic mean method for dependent tests implemented in package `harmonicmeanp`, as determined by the `method` parameter. It is also possible to specify the maximum p-value cutoff (`pvcut`) for filtering the output.

In order to prioritise putative 'driver' interactions within the differentially interacting regions, `getCandidateInteractions` enables filtering them by minimum |asinh (Chicago score)| (`minDeltaAsinhScore`) between conditions. The asinh transformation helps compress the upper range of the scores, where differences between them are less interpretable compared with those in the lower range.

```
outCI <- getCandidateInteractions(chicdiff.settings = chicdiff.settings,
                    output = output, peakFiles = peakFiles,
                    pvcut = 0.05, minDeltaAsinhScore = 1)

head(outCI)
```

```
##    baitID   oeID baitChr baitstart baitend                        baitN
ame
## 1: 320543 320558      19    688314  711604             PALM;PRSS57;RPS2
```

```
P52
## 2: 320543 320559      19    688314  711604            PALM;PRSS57;RPS2
P52
## 3: 320571 320562      19   1155128 1228413            HMGB2P1;SBNO2;ST
K11
## 4: 320653 320713      19   1847648 1858037              CTB-31O20.6;RE
XO1
## 5: 320654 320712      19   1858038 1871184              CTB-31O20.8;KL
F16
## 6: 320656 320695      19   1872080 1893040 ABHD17A;CTB-31O20.2;CTB-31O2
0.9
##       NCD4_22   NCD4_23     Mon_2     Mon_3 min_weighted_padj
## 1: 0.53637482 0.7167207  5.133810 3.6048299      4.860384e-05
## 2: 0.05679063 0.4410137  5.635222 0.6161083      4.860384e-05
## 3: 1.87138155 1.0352457  5.262635 4.6468402      3.124040e-02
## 4: 1.14288589 1.3502229  6.085978 4.8231789      1.152692e-04
## 5: 0.86313303 0.0000000  5.865727 3.3073795      8.950682e-09
## 6: 2.95760996 2.9998770 13.895113 5.9483750      4.702047e-02
##    deltaAsinhScore       regionIDs
## 1:        1.589144         147,148
## 2:        1.611052         147,148
## 3:        1.134939 220,221,222,223
## 4:        1.352462             320
## 5:        1.808772             327
## 6:        1.178703     336,337,338
##                                                    log2FoldChanges
## 1:                                  2.02629426385277,2.06214996151406
## 2:                                  2.02629426385277,2.06214996151406
## 3: 1.13894588194587,1.12097182669164,1.19693084784918,1.18618479820274
## 4:                                                   1.87376012559898
## 5:                                                   1.86460149355567
## 6:                 1.22022652028753,1.14976350020938,1.23443618487335
##                                                                weighte
d_padj
## 1:                                  4.86038446350271e-05,0.000134049544
709904
## 2:                                  4.86038446350271e-05,0.000134049544
709904
## 3: 0.0774365301254193,0.0665288113014112,0.0312403954518645,0.034186194
499776
## 4:                                                     0.000115269246
808153
## 5:                                                     8.950682022249
61e-09
## 6:                  0.0504740260944529,0.0711886472401962,0.0470204669
023691
##                                                          OEranges
## 1:                                905873-1086647,906012-1086676
## 2:                                905873-1086647,906012-1086676
## 3: 949495-1126791,986107-1141490,1036370-1148123,1065923-1148123
## 4:                                                 2308625-2421217
## 5:                                                 2299803-2412609
## 6:              2211203-2262813,2213001-2264831,2223023-2271704
```