

# Package: HiContacts (via r-universe)

June 27, 2024

**Title** Analysing cool files in R with HiContacts

**Version** 1.7.0

**Date** 2022-08-16

**Description** HiContacts provides a collection of tools to analyse and visualize Hi-C datasets imported in R by HiCExperiment.

**License** MIT + file LICENSE

**URL** <https://github.com/js2264/HiContacts>

**BugReports** <https://github.com/js2264/HiContacts/issues>

**Depends** R (>= 4.2), HiCExperiment

**Imports** InteractionSet, SummarizedExperiment, GenomicRanges, IRanges, GenomeInfoDb, S4Vectors, methods, BiocGenerics, BiocIO, BiocParallel, RSpectra, Matrix, tibble, tidyr, dplyr, readr, stringr, ggplot2, ggrastr, scales, stats, utils

**Suggests** HiContactsData, rtracklayer, GenomicFeatures, Biostrings, BSgenome.Scerevisiae.UCSC.sacCer3, WGCNA, Rfast, terra, patchwork, testthat (>= 3.0.0), BiocStyle, knitr, rmarkdown

**biocViews** HiC, DNA3DStructure

**Encoding** UTF-8

**VignetteBuilder** knitr

**LazyData** false

**Roxygen** list(markdown = TRUE)

**RoxygenNote** 7.2.3

**Repository** <https://bioc.r-universe.dev>

**RemoteUrl** <https://github.com/bioc/HiContacts>

**RemoteRef** HEAD

**RemoteSha** a48dfc82aab6eaea51468f9ce11e85f835fecbe

## Contents

arithmetics . . . . .	2
cisTransRatio . . . . .	5
Contacts . . . . .	6
distanceLaw . . . . .	7
getCompartments . . . . .	8
getDiamondInsulation . . . . .	9
getLoops . . . . .	10
HiContacts-plots . . . . .	10
palettes . . . . .	11
plot4C . . . . .	11
plotMatrix . . . . .	12
plotPs . . . . .	15
plotSaddle . . . . .	16
plotScalogram . . . . .	16
scalogram . . . . .	17
tracks . . . . .	18
virtual4C . . . . .	18
<b>Index</b>	<b>20</b>

---

 arithmetics

*HiContacts arithmetics functionalities*


---

## Description

Different arithmetic operations can be performed on Hi-C contact matrices:

- normalize a contact matrix using iterative correction;
- detrend a contact matrix, i.e. remove the distance-dependent contact trend;
- autocorrelate a contact matrix: this is typically done to highlight large-scale compartments;
- divide one contact matrix by another;
- merge multiple contact matrices;
- despeckle (i.e. smooth out) a contact matrix out;
- aggregate (average) a contact matrices over a set of genomic loci of interest;
- boost Hi-C signal by enhancing long-range interactions while preserving short-range interactions (this is adapted from Boost-HiC);
- subsample interactions using a proportion or a fixed number of final interactions.
- coarsen a contact matrix to a larger (coarser) resolution

**Usage**

```

## S4 method for signature 'HiCExperiment'
aggregate(
  x,
  targets,
  flankingBins = 51,
  maxDistance = NULL,
  BPPARAM = BiocParallel::bpparam()
)

detrond(x, use.scores = "balanced")

autocorrelate(x, use.scores = "balanced", detrend = TRUE, ignore_ndiags = 3)

divide(x, by, use.scores = "balanced", pseudocount = 0)

## S4 method for signature 'HiCExperiment,HiCExperiment'
merge(x, y, ..., use.scores = "balanced", FUN = mean)

despeckle(x, use.scores = "balanced", focal.size = 1)

boost(x, use.scores = "balanced", alpha = 1, full.replace = FALSE)

coarsen(x, bin.size)

## S4 method for signature 'HiCExperiment'
normalize(
  object,
  use.scores = "count",
  niters = 200,
  min.nnz = 10,
  mad.max = 3
)

subsample(x, prop)

```

**Arguments**

<code>x, y, object</code>	a <code>HiCExperiment</code> object
<code>targets</code>	Set of chromosome coordinates for which interaction counts are extracted from the Hi-C contact file, provided as a <code>GRanges</code> object (for diagonal-centered loci) or as a <code>GInteractions</code> object (for off-diagonal coordinates).
<code>flankingBins</code>	Number of bins on each flank of the bins containing input targets.
<code>maxDistance</code>	Maximum distance to use when compiling distance decay
<code>BPPARAM</code>	<code>BiocParallel</code> parameters
<code>use.scores</code>	Which scores to use to perform operations
<code>detrend</code>	Detrend matrix before performing autocorrelation

<code>ignore.ndiags</code>	ignore N diagonals when calculating correlations
<code>by</code>	a <code>HiCExperiment</code> object
<code>pseudocount</code>	Add a pseudocount when dividing matrices (Default: 0)
<code>...</code>	<code>HiCExperiment</code> objects. For aggregate, targets (a set of <code>GRanges</code> or <code>GInteractions</code> ).
<code>FUN</code>	merging function
<code>focal.size</code>	Size of the smoothing rectangle
<code>alpha</code>	Power law scaling factor. As indicated in Boost-HiC documentation, the alpha parameter influences the weighting of contacts: if $\alpha < 1$ long-range interactions are prioritized; if $\alpha \gg 1$ short-range interactions have more weight when computing the distance matrix.
<code>full.replace</code>	Whether to replace the entire set of contacts, rather than only filling the missing interactions ( <code>count=0</code> ) (Default: <code>FALSE</code> )
<code>bin.size</code>	Bin size to coarsen a <code>HiCExperiment</code> at
<code>niters</code>	Number of iterations for ICE matrix balancing
<code>min.nnz</code>	Filter bins with less than <code>min.nnz</code> non-zero elements when performing ICE matrix balancing
<code>mad.max</code>	Filter out bins whose log coverage is less than <code>mad.max</code> median absolute deviations below the median bin log coverage.
<code>prop</code>	Float between 0 and 1, or integer corresponding to the # of

**Value**

a `HiCExperiment` object with extra scores

**Examples**

```
#### -----
#### Normalize a contact matrix
#### -----

library(HiContacts)
contacts_yeast <- contacts_yeast()
normalize(contacts_yeast)

#### -----
#### Detrending a contact matrix
#### -----

detrend(contacts_yeast)

#### -----
#### Auto-correlate a contact matrix
#### -----

autocorrelate(contacts_yeast)
```

```

#### -----
#### Divide 2 contact matrices
#### -----

contacts_yeast <- refocus(contacts_yeast, 'II')
contacts_yeast_eco1 <- contacts_yeast_eco1() |> refocus('II')
divide(contacts_yeast_eco1, by = contacts_yeast)

#### -----
#### Merge 2 contact matrices
#### -----

merge(contacts_yeast_eco1, contacts_yeast)

#### -----
#### Despeckle (smoothen) a contact map
#### -----

despeckle(contacts_yeast)

#### -----
#### Aggregate a contact matrix over centromeres, at different scales
#### -----

contacts <- contacts_yeast() |> zoom(resolution = 1000)
centros <- topologicalFeatures(contacts, 'centromeres')
aggregate(contacts, targets = centros, flankingBins = 51)

#### -----
#### Enhance long-range interaction signal
#### -----

contacts <- contacts_yeast() |> zoom(resolution = 1000) |> refocus('II')
boost(contacts, alpha = 1)

#### -----
#### Subsample & "coarsen" contact matrix
#### -----

subcontacts <- subsample(contacts, prop = 100000)
coarsened_subcontacts <- coarsen(subcontacts, bin.size = 4000)

```

---

cisTransRatio

*cisTransRatio*


---

### Description

Quickly computes a cis-trans ratio of interactions.

### Usage

```
cisTransRatio(x)
```

**Arguments**

x                    A HiCExperiment object over the full genome

**Value**

a tibble, listing for each chr. the % of cis/trans interactions

**Examples**

```
library(HiContacts)
full_contacts_yeast <- contacts_yeast(full = TRUE)
cisTransRatio(full_contacts_yeast)
```

---

Contacts

*Contacts*

---

**Description**

This function has been deprecated in favor of the generic HiCExperiment() constructor (from HiCExperiment package).

**Usage**

```
Contacts(
  file,
  resolution = NULL,
  focus = NULL,
  metadata = list(),
  topologicalFeatures = S4Vectors::SimpleList(loops =
    S4Vectors::Pairs(GenomicRanges::GRanges(), GenomicRanges::GRanges()), borders =
    GenomicRanges::GRanges(), compartments = GenomicRanges::GRanges(), viewpoints =
    GenomicRanges::GRanges()),
  pairsFile = NULL
)
```

**Arguments**

file	Path to a (m)cool file
resolution	Resolution to use with mcool file
focus	focus Chr. coordinates for which interaction counts are extracted from the (m)cool file, provided as a character string (e.g. "II:4001-5000"). If not provided, the entire (m)cool file will be imported.
metadata	list of metadata
topologicalFeatures	topologicalFeatures provided as a named SimpleList
pairsFile	Path to an associated .pairs file

**Value**

a new HiCExperiment object.

**Examples**

```
library(HiContacts)
library(HiContactsData)
mcool_path <- HiContactsData::HiContactsData('yeast_wt', 'mcool')
Contacts(mcool_path, resolution = 1000)
```

---

distanceLaw	<i>Compute the law of distance-dependent contact frequency, a.k.a. P(s)</i>
-------------	---

---

**Description**

P(s) will be approximated if no pairs are provided, or the exact P(s) will be computed if a .pairs file is added to the HiCExperiment object using pairsFile(x) <- "...".

**Usage**

```
distanceLaw(x, coords, ...)

## S4 method for signature 'GInteractions,missing'
distanceLaw(x, by_chr = FALSE)

## S4 method for signature 'HiCExperiment,missing'
distanceLaw(
  x,
  by_chr = FALSE,
  filtered_chr = c("XII", "chrXII", "chr12", "12", "Mito", "MT", "chrM")
)

## S4 method for signature 'PairsFile,missing'
distanceLaw(
  x,
  by_chr = FALSE,
  filtered_chr = c("XII", "chrXII", "chr12", "12", "Mito", "MT", "chrM"),
  chunk_size = 1e+05
)

## S4 method for signature 'HiCExperiment,GRanges'
distanceLaw(x, coords, chunk_size = 1e+05)

## S4 method for signature 'PairsFile,GRanges'
distanceLaw(x, coords, chunk_size = 1e+05)

localDistanceLaw(x, coords = coords)
```

**Arguments**

x	A HiCExperiment object
coords	GRanges to specify which genomic loci to use when computing P(s)
...	Arguments passed to corresponding method
by_chr	by_chr
filtered_chr	filtered_chr
chunk_size	For pairs files which do not fit in memory, pick a number of pairs to parse by chunks (1e7 should be a good compromise)

**Value**

a tibble

**Examples**

```

contacts_yeast <- contacts_yeast()
ps <- distanceLaw(contacts_yeast)
ps
local_ps <- localDistanceLaw(
  contacts_yeast,
  GenomicRanges::GRanges(
    c("telomere" = "II:1-20000", "arm" = "II:300001-700000")
  )
)
local_ps

```

---

getCompartments	<i>Contact map compartments</i>
-----------------	---------------------------------

---

**Description**

Computes eigen vectors for each chromosome using cis contacts and extract chromosome compartments.

**Usage**

```

getCompartments(
  x,
  resolution = NULL,
  genome = NULL,
  chromosomes = NULL,
  neigens = 3,
  sort_eigens = FALSE,
  BPPARAM = BiocParallel::bpparam()
)

```



**Arguments**

x	A HiCExperiment object over a full genome
resolution	Which resolution to use to compute eigen vectors
genome	a BSgenome or DNASTringSet object associated with the Hi-C contact matrix.
chromosomes	character or integer vector indicating which
neigens	Numver of eigen vectors to extract
sort_eigens	Can be FALSE or one of c('Spearman', 'Pearson')
BPPARAM	BiocParallel parallelization settings

**Value**

A HiCExperiment object with additional eigens metadata containing the normalized eigenvectors and a new "compartments" topologicalFeatures storing A and B compartments as a GRanges object.

**Examples**

```
library(HiContacts)
full_contacts_yeast <- contacts_yeast(full = TRUE)
comps <- getCompartments(full_contacts_yeast)
metadata(comps)$eigens
```

---

getDiamondInsulation    *Contact map insulation*

---

**Description**

Computes diamond insulation score along the entire genome

**Usage**

```
getDiamondInsulation(x, window_size = NULL, BPPARAM = BiocParallel::bpparam())

getBorders(x, weak_threshold = 0.2, strong_threshold = 0.5)
```

**Arguments**

x	A HiCExperiment object over a full genome
window_size	Which window size to use to compute diamond insulation score (default: 10 * resolution)
BPPARAM	BiocParallel parallelization settings
weak_threshold	Less stringent cutoff to call borders in the diamond insulation score
strong_threshold	More stringent cutoff to call borders in the diamond insulation score

**Value**

a HiCExperiment object with additional insulation metadata, containing the diamond insulation score computed

**Examples**

```
library(HiContacts)
hic <- contacts_yeast() |>
  refocus('II:1-300000') |>
  zoom(1000)
diams <- getDiamondInsulation(hic)
getDiamondInsulation(diams)
```

---

getLoops	<i>Finding loops in contact map</i>
----------	-------------------------------------

---

**Description**

Find loops using chromosight.

This function is actually provided by the HiCool package rather than the HiContacts package. HiCool provides a self-managed conda environment, and this limits

**Usage**

```
getLoops(...)
```

**Arguments**

... Parameters passed to HiCool::getLoops().

---

HiContacts-plots	<i>HiContacts plotting functionalities</i>
------------------	--

---

**Description**

Several plots can be generated in HiContacts:

- Hi-C contact matrices
- Distance-dependant interaction frequency decay (a.k.a. "Distance law" or "P(s)")
- Virtual 4C profiles
- Scalograms
- Saddle plots

---

palettes

*Matrix palettes*

---

**Description**

Matrix palettes

**Usage**

bwrColors()

bbrColors()

bgrColors()

afmhotrColors()

coolerColors()

rainbowColors()

**Value**

A vector of colours carefully picked for Hi-C contact heatmaps

**Examples**

```
bwrColors()
bbrColors()
bgrColors()
afmhotrColors()
coolerColors()
rainbowColors()
```

---

plot4C

*Plotting virtual 4C profiles*

---

**Description**

Plotting virtual 4C profiles

**Usage**

```
plot4C(x, mapping = ggplot2::aes(x = center, y = score, col = seqnames))
```

**Arguments**

x GRanges, generally the output of virtual4C()  
mapping aes to pass on to ggplot2 (default: ggplot2::aes(x = center, y = score, col = seqnames))

**Value**

ggplot

**Examples**

```
contacts_yeast <- contacts_yeast()
v4C <- virtual4C(contacts_yeast, GenomicRanges::GRanges('II:490001-510000'))
plot4C(v4C)
```

---

plotMatrix *Plotting a contact matrix*

---

**Description**

Plotting a contact matrix

**Usage**

```
plotMatrix(x, ...)

montage(x, ...)

## S4 method for signature 'HiCExperiment'
plotMatrix(
  x,
  compare.to = NULL,
  use.scores = "balanced",
  scale = "log10",
  maxDistance = NULL,
  loops = NULL,
  borders = NULL,
  tracks = NULL,
  limits = NULL,
  dpi = 500,
  rasterize = TRUE,
  symmetrical = TRUE,
  chrom_lines = TRUE,
  show_grid = FALSE,
  cmap = NULL,
  caption = TRUE
)
```

```
## S4 method for signature 'GInteractions'
plotMatrix(
  x,
  use.scores = NULL,
  scale = "log10",
  maxDistance = NULL,
  loops = NULL,
  borders = NULL,
  tracks = NULL,
  limits = NULL,
  dpi = 500,
  rasterize = TRUE,
  symmetrical = TRUE,
  chrom_lines = TRUE,
  show_grid = FALSE,
  cmap = NULL
)

## S4 method for signature 'matrix'
plotMatrix(
  x,
  scale = "log10",
  limits = NULL,
  dpi = 500,
  rasterize = TRUE,
  cmap = NULL
)

## S4 method for signature 'AggrHiCExperiment'
plotMatrix(
  x,
  use.scores = "balanced",
  scale = "log10",
  maxDistance = NULL,
  loops = NULL,
  borders = NULL,
  limits = NULL,
  dpi = 500,
  rasterize = TRUE,
  chrom_lines = TRUE,
  show_grid = FALSE,
  cmap = NULL,
  caption = TRUE
)

## S4 method for signature 'AggrHiCExperiment'
montage(
```

```

x,
use.scores = "balanced",
scale = "log10",
limits = NULL,
dpi = 500,
rasterize = TRUE,
cmap = NULL
)

```

### Arguments

x	A HiCExperiment object
...	Extra arguments passed to the corresponding method.
compare.to	Compare to a second HiC matrix in the lower left corner
use.scores	Which scores to use in the heatmap
scale	Any of 'log10', 'log2', 'linear', 'exp0.2' (Default: 'log10')
maxDistance	maximum distance. If provided, the heatmap is plotted horizontally
loops	Loops to plot on top of the heatmap, provided as GInteractions
borders	Borders to plot on top of the heatmap, provided as GRanges
tracks	Named list of bigwig tracks imported as R1e
limits	color map limits
dpi	DPI to create the plot (Default: 500)
rasterize	Whether the generated heatmap is rasterized or vectorized (Default: TRUE)
symmetrical	Whether to enforce a symmetrical heatmap (Default: TRUE)
chrom_lines	Whether to display separating lines between chromosomes, should any be necessary (Default: TRUE)
show_grid	Whether to display an underlying grid (Default: FALSE)
cmap	Color scale to use. (Default: bgrColors() if limits are c(-1, 1) and coolerColors() otherwise)
caption	Whether to display a caption (Default: TRUE)

### Value

ggplot object

### Examples

```

contacts_yeast <- contacts_yeast()
plotMatrix(
  contacts_yeast,
  use.scores = 'balanced',
  scale = 'log10',
  limits = c(-4, -1)
)

```

---

plotPs *Plotting a P(s) distance law*

---

### Description

Plotting a P(s) distance law

### Usage

```
plotPs(x, mapping, xlim = c(5000, 499000), ylim = c(1e-08, 1e-04))
```

```
plotPsSlope(x, mapping, xlim = c(5000, 499000), ylim = c(-3, 0))
```

### Arguments

x	the output data.frame of distanceLaw function
mapping	aes to pass on to ggplot2
xlim	xlim
ylim	ylim

### Value

ggplot

### Examples

```
## Single P(s)

contacts_yeast <- contacts_yeast()
ps <- distanceLaw(contacts_yeast)
plotPs(ps, ggplot2::aes(x = binned_distance, y = norm_p))

## Comparing several P(s)

contacts_yeast <- contacts_yeast()
contacts_yeast_eco1 <- contacts_yeast_eco1()
ps_wt <- distanceLaw(contacts_yeast)
ps_wt$sample <- 'WT'
ps_eco1 <- distanceLaw(contacts_yeast_eco1)
ps_eco1$sample <- 'eco1'
ps <- rbind(ps_wt, ps_eco1)
plotPs(ps, ggplot2::aes(x = binned_distance, y = norm_p, group = sample, color = sample))
plotPsSlope(ps, ggplot2::aes(x = binned_distance, y = slope, group = sample))
```

---

plotSaddle *Plotting saddle plots*

---

### Description

Plotting saddle plots

### Usage

```
plotSaddle(
  x,
  nbins = 50,
  limits = c(-1, 1),
  plotBins = FALSE,
  BPPARAM = BiocParallel::bpparam()
)
```

### Arguments

x	a HiCExperiment object with a stored eigens metadata
nbins	Number of bins to use to discretize the eigenvectors
limits	limits for color map being used
plotBins	Whether to plot the distribution of bins on top of the plot
BPPARAM	a BiocParallel registered method

### Value

ggplot

---

plotScalogram *Plotting scalograms*

---

### Description

Plotting scalograms

### Usage

```
plotScalogram(x, ylim = c(500, 1e+05))
```

### Arguments

x	GRanges, the output of scalogram()
ylim	Range of distances to use for y-axis in scalograms



**Value**

ggplot

**Examples**

```
contacts_yeast <- HiCExperiment::contacts_yeast()
pairsFile(contacts_yeast) <- HiContactsData::HiContactsData(
  'yeast_wt', format = 'pairs.gz'
)
scalo <- scalogram(contacts_yeast['II'])
plotScalogram(scalo)
```

---

scalogram

*Compute a scalogram of contacts*

---

**Description**

Compute a scalogram of contacts

**Usage**

```
scalogram(x, dist_min = 0, nbins = 250, probs = c(0.25, 0.5, 0.75))
```

**Arguments**

x	A HiCExperiment object
dist_min	Minimum distance for interactions to be considered.
nbins	Number of bins to divide each chromosome
probs	Quantiles of interactions

**Value**

a tibble

a tibble

**Examples**

```
contacts_yeast <- HiCExperiment::contacts_yeast()
pairsFile(contacts_yeast) <- HiContactsData::HiContactsData(
  'yeast_wt', format = 'pairs.gz'
)
scalo <- scalogram(contacts_yeast['II'])
scalo
```

---

tracks	<i>Aligning tracks with HiCExperiment objects</i>
--------	---

---

**Description**

Aligning tracks with HiCExperiment objects

**Usage**

```
## S4 method for signature 'HiCExperiment'
coverage(x, use.pairs = FALSE, bin.size = resolution(x))
```

**Arguments**

x	A HiCExperiment object over a full genome
use.pairs	logical. Whether to use pairsFile to compute Hi-C coverage
bin.size	if use.pairs == TRUE, to which resolution

**Value**

A HiCExperiment object with 2 added columns in regions(x)

**Examples**

```
mcool_file <- HiContactsData::HiContactsData('yeast_wt', format = 'mcool')
hic <- import(mcool_file, format = 'mcool', resolution = 1000)
coverage(hic)
```

---

virtual4C	<i>Computing virtual 4C profiles</i>
-----------	--------------------------------------

---

**Description**

From a (m)cool pre-imported in memory, computes a 4C profile using a user-specified viewpoint.

**Usage**

```
virtual4C(x, viewpoint, use.scores = "balanced")
```

**Arguments**

x	a HiCExperiment object
viewpoint	viewpoint, defined as a GRanges
use.scores	use.scores

**Value**

A tibble with the contact frequency of the viewpoint, per bin along the imported genomic range.

**Examples**

```
library(HiContacts)
contacts_yeast <- contacts_yeast()
v4C <- virtual4C(contacts_yeast, GenomicRanges::GRanges('II:490001-510000'))
v4C
```

# Index

afmhoTrColors (palettes), 11  
aggregate, HiCExperiment-method  
    (arithmetics), 2  
arithmetics, 2  
autocorrelate (arithmetics), 2

bbrColors (palettes), 11  
bgrColors (palettes), 11  
boost (arithmetics), 2  
bwrColors (palettes), 11

cisTransRatio, 5  
coarsen (arithmetics), 2  
Contacts, 6  
coolerColors (palettes), 11  
coverage, HiCExperiment-method (tracks),  
    18

despeckle (arithmetics), 2  
detrend (arithmetics), 2  
distanceLaw, 7  
distanceLaw, GInteractions, missing-method  
    (distanceLaw), 7  
distanceLaw, HiCExperiment, GRanges-method  
    (distanceLaw), 7  
distanceLaw, HiCExperiment, missing-method  
    (distanceLaw), 7  
distanceLaw, PairsFile, GRanges-method  
    (distanceLaw), 7  
distanceLaw, PairsFile, missing-method  
    (distanceLaw), 7  
divide (arithmetics), 2

getBorders (getDiamondInsulation), 9  
getCompartments, 8  
getDiamondInsulation, 9  
getLoops, 10

HiContacts-plots, 10

localDistanceLaw (distanceLaw), 7

merge, HiCExperiment, HiCExperiment-method  
    (arithmetics), 2  
montage (plotMatrix), 12  
montage, AggrHiCExperiment-method  
    (plotMatrix), 12

normalize, HiCExperiment-method  
    (arithmetics), 2

palettes, 11  
plot4C, 11  
plotMatrix, 12  
plotMatrix, AggrHiCExperiment-method  
    (plotMatrix), 12  
plotMatrix, GInteractions-method  
    (plotMatrix), 12  
plotMatrix, HiCExperiment-method  
    (plotMatrix), 12  
plotMatrix, matrix-method (plotMatrix),  
    12  
plotPs, 15  
plotPsSlope (plotPs), 15  
plotSaddle, 16  
plotScalogram, 16  
Ps (distanceLaw), 7

rainbowColors (palettes), 11

scalogram, 17  
subsample (arithmetics), 2

tracks, 18

virtual4C, 18