Package ‘fourSynergy’

May 8, 2026

Type Package

Title Ensemble algorithm for 4C-seq data

Version 1.0.0

Description fourSynergy is an ensemble algorithm leveraging synergies among the existing 4C-seq algorithms r3C-seq, peakC, r.4cker and fourSig. It uses a weighted voting approach to perform improved interaction calling. fourSynergy supports also differential interaction calling.

License LGPL-3

Encoding UTF-8

Depends R (>= 4.5.0), GenomicRanges

Imports magrittr, dplyr, ggplot2, tibble, org.Hs.eg.db, org.Mm.eg.db, reshape2, tidyr, methods, jsonlite, karyoploteR, TxDb.Mmusculus.UCSC.mm10.knownGene, TxDb.Hsapiens.UCSC.hg19.knownGene, bamsignals, DESeq2, GenomeInfoDb, graphics, stringr, yaml

VignetteBuilder knitr

Suggests testthat (>= 3.0.0), knitr, rmarkdown, BiocStyle

biocViews Sequencing, Software, DifferentialPeakCalling

BugReports https://github.com/sophiewind/fourSynergy/issues

URL https://github.com/sophiewind/fourSynergy

Roxygen list(markdown = TRUE)

RoxygenNote 7.3.3

git_url https://git.bioconductor.org/packages/fourSynergy

git_branch RELEASE_3_23

git_last_commit 0258f45

git_last_commit_date 2026-04-28

Repository Bioconductor 3.23

Date/Publication 2026-05-07

Author Sophie Wind [aut, cre] (ORCID: <https://orcid.org/0009-0002-1174-8201>), Carolin Walter [aut, fnd] (ORCID: <https://orcid.org/0000-0002-5155-7728>)

Maintainer Sophie Wind <sophie.wind@uni-muenster.de>

Contents

`fourSynergy-package` *fourSynergy: Ensemble based interaction calling in 4C-seq data*

Description

fourSynergy is an ensemble algorithm leveraging synergies among the existing 4C-seq algorithms r3C-seq, peakC, r.4cker and fourSig. It uses a weighted voting approach to perform improved interaction calling. fourSynergy supports also differential interaction calling.

Author(s)

Maintainer: Sophie Wind <`sophie.wind@uni-muenster.de`> (ORCID)

Authors:

- Carolin Walter <`carolin.walter@uni-muenster.de`> (ORCID) [funder]

See Also

Useful links:

- `https://github.com/sophiewind/fourSynergy`
- Report bugs at `https://github.com/sophiewind/fourSynergy/issues`

`checkConfig` *checkConfig*

Description

checkConfig

Usage

```
checkConfig(config)
```

Arguments

`config`	config file with path.

Value

TRUE if config is valid.

Examples

```
config <- system.file("extdata", "Datasets", "Demo", "info.yaml",
    package = "fourSynergy")
checkConfig(config)
```

`consensusIa` *consensusIa*

Description

This function performs an optimized weighted voting of 4C-seq tools.

Usage

```
consensusIa(ia, model = "F1")
```

Arguments

`ia`	fourSynergy object with interactions from all base tools (peakC, r3c-seq, fourSig, r4cker) and other relevant information.
`model`	Selected optimization model. Either 'F1' or 'AUPRC'.

Value

fourSynergy object with interactions from all base tools and weighted voting results.

Examples

```
config <- system.file("extdata", "Datasets", "Demo", "info.yaml",
    package = "fourSynergy"
)
res_path <- system.file("extdata", "results", "Demo",
    package = "fourSynergy"
)
tracks <- system.file("extdata", "results", "Demo", "alignment",
    package = "fourSynergy"
)
sia <- createIa(res_path = res_path, config = config, tracks = tracks)
sia <- consensusIa(ia = sia, model = "AUPRC")
```

`createIa` *createIa*

Description

This function reads the interaction bed files created by the pipeline and transfers this information into an GrangesList.

Usage

```
createIa(res_path = character(), config = list(), tracks = "")
```

Arguments

`res_path`	Path to results created by the pipeline. Typically stored in the `results/[dataset]/nearbait_area.bed`.
`config`	Path of config file.
`tracks`	Path to alignment files.

Value

fourSynergy object with interactions from all base tools.

Examples

```
config <- system.file("extdata", "Datasets", "Demo", "info.yaml",
    package = "fourSynergy"
)
res_path <- system.file("extdata", "results", "Demo",
    package = "fourSynergy"
)
tracks <- system.file("extdata", "results", "Demo", "alignment",
    package = "fourSynergy"
)
ia <- createIa(res_path = res_path, config = config, tracks = tracks)
```

`createKaryoplot` *Internal function to create karyoplot*

Description

Internal function to create karyoplot

Usage

```
createKaryoplot(ia, type = 1, cex = 1, cex.axis = 1, cex.lab = 1, cex.main = 1)
```

Arguments

`ia`	fourSynergy object with interactions from all base tools (peakC, r3c-seq, fourSig, r4cker) and other relevant information.
`type`	Plottype.
`cex`	character expansion
`cex.axis`	character expansion axis
`cex.lab`	character expansion labels
`cex.main`	character expansion main

Value

karyoplot base

`differentialAnalysis` *differentialAnalysis*

Description

This function performs differential analysis to identify differential interacting regions using DESeq2.

Usage

```
differentialAnalysis(ia, fitType = "local")
```

Arguments

`ia`	fourSynergy object with interactions from all base tools (peakC, r3c-seq, fourSig, r4cker) and other relevant information.
`fitType`	Parameter for DESeq2s estimateDispersions(). Should be either "parametric", "local", "mean", or "glmGamPoi" for the type of fitting of dispersions to the mean intensity.

Value

sia object with GRanges of DESeq results in the diff slot.

References

https://doi.org/10.1186/s13059-014-0550-8

Examples

```
config <- system.file("extdata", "Datasets", "Demo", "info.yaml",
    package = "fourSynergy"
)
res_path <- system.file("extdata", "results", "Demo",
    package = "fourSynergy"
)
tracks <- system.file("extdata", "results", "Demo", "alignment",
    package = "fourSynergy"
)
sia <- createIa(res_path = res_path, config = config, tracks = tracks)
sia <- consensusIa(ia = sia, model = "AUPRC")
sia <- differentialAnalysis(ia = sia, fitType = "mean")
```

`fourSynergy-accessors` *fourSynergy accessors*

Description

Standard getter methods for `fourSynergy` object slots.

Usage

```
getMetadata(object)

getExpInteractions(object)

getCtrlInteractions(object)

getExpConsensus(object)

getCtrlConsensus(object)

getViewpoint(object)

getVirtualFragmentLibrary(object)

getTracks(object)

getDifferential(object)
```

```
getDDS(object)
```

Arguments

`object`	A `fourSynergy` object

Value

Various slots of the object (see individual methods)

Functions

- `getMetadata()`: Get metadata
- `getExpInteractions()`: Get experimental interactions
- `getCtrlInteractions()`: Get control interactions
- `getExpConsensus()`: Get experimental consensus
- `getCtrlConsensus()`: Get control consensus
- `getViewpoint()`: Get viewpoint
- `getVirtualFragmentLibrary()`: Get virtual fragment library
- `getTracks()`: Get tracks
- `getDifferential()`: Get differential expression
- `getDDS()`: Get DESeq results

`fourSynergy-class` *fourSynergy Class*

Description

S4 class storing data collected from 4C-seq analyses.

Slots

`metadata`	Experimental metadata from config file.
`expInteractions`	Base tool interactions found in the experiment.
`ctrlInteractions`	Base tool interactions found in the control.
`expConsensus`	Consensus interactions found in the experiment.
`ctrlConsensus`	Consensus interactions found in the control.
`vp`	Viewpoint position.
`vfl`	Virtual fragment library.
`tracks`	Path to the alignment files.
`differential`	Results of differential interaction calling (res).
`dds`	Results of differential interaction calling (dds).

`fourSynergy-setters` *fourSynergy setters*

Description

Standard setter methods for `fourSynergy` object slots.

Usage

```
setDifferential(object, value)

setDds(object, value)

setExpConsensus(object, value)

setctrlConsensus(object, value)
```

Arguments

`object`	A `fourSynergy` object
`value`	Replacement value for the respective slot

Value

Updated `fourSynergy` object (invisibly)

Functions

- `setDifferential()`: Set setDifferential slot
- `setDds()`: Set setDds slot
- `setExpConsensus()`: Set setExpConsensus slot
- `setctrlConsensus()`: Set setCtrlConsensus slot

`plotBaseTracks` *plotBaseTracks*

Description

plotBaseTracks

Usage

```
plotBaseTracks(ia, highlight_regions = NULL, max_range = 3000)
```

Arguments

`ia`	fourSynergy object with interactions from all base tools
`highlight_regions`	regions to highlight in the plot
`max_range`	maximum plotting range

Value

Track-plots for all treatments with interactions from base tools

Examples

```
config <- system.file("extdata", "Datasets", "Demo", "info.yaml",
    package = "fourSynergy"
)
res_path <- system.file("extdata", "results", "Demo",
    package = "fourSynergy"
)
tracks <- system.file("extdata", "results", "Demo", "alignment",
    package = "fourSynergy"
)
sia <- createIa(res_path = res_path, config = config, tracks = tracks)
plotBaseTracks(sia)
```

`plotConsensusIa` *plotConsensusIa*

Description

This function creates a karyotype plot displaying the interaction calls from the consensus approach.

Usage

```
plotConsensusIa(
  ia = GRangesList(),
  genes_of_interest = NULL,
  cex.chr = 1,
  cex.ideo = 0.6,
  cex.y.lab = 0.6,
  cex.y.track = 0.6,
  cex.vp = 1,
  cex.leg = 0.6,
  highlight_regions = NULL,
  plot_spider = FALSE,
  gene.name.cex = 1
)
```

Arguments

`ia`	fourSynergy object with interactions from all base tools (peakC, r3c-seq, fourSig, r4cker) and other relevant information.
`genes_of_interest`	Vector with genes of interest. Set to `all` if you want to plot all genes in this area.
`cex.chr`	character expansion of chromosome label.
`cex.ideo`	character expansion base numbers of ideogram.
`cex.y.lab`	character expansion for y labels.
`cex.y.track`	character expansion y axis track.
`cex.vp`	character expansion viewpoint label.
`cex.leg`	character expansion for legend.
`highlight_regions`	regions to highlight in the plot.
`plot_spider`	plotting connections from VP to interactions.
`gene.name.cex`	character expansion for gene names.

Value

karyoplot with calling results.

Examples

```
config <- system.file("extdata", "Datasets", "Demo", "info.yaml",
    package = "fourSynergy"
)
res_path <- system.file("extdata", "results", "Demo",
    package = "fourSynergy"
)
tracks <- system.file("extdata", "results", "Demo", "alignment",
    package = "fourSynergy"
)
sia <- createIa(res_path = res_path, config = config, tracks = tracks)
sia <- consensusIa(ia = sia, model = "AUPRC")
plotConsensusIa(ia = sia)
```

`plotConsensusTracks` *plotConsensusTracks*

Description

plotConsensusTracks

Usage

```
plotConsensusTracks(ia, highlight_regions = NULL, max_range = 3000)
```

Arguments

`ia`	fourSynergy object with interactions from all base tools
`highlight_regions`	regions to highlight in the plot
`max_range`	maximum plotting range

Value

Track-plots for all treatments with interactions from consensus tool

Examples

```
config <- system.file("extdata", "Datasets", "Demo", "info.yaml",
    package = "fourSynergy"
)
res_path <- system.file("extdata", "results", "Demo",
    package = "fourSynergy"
)
tracks <- system.file("extdata", "results", "Demo", "alignment",
    package = "fourSynergy"
)
sia <- createIa(res_path = res_path, config = config, tracks = tracks)
sia <- consensusIa(sia, model = "AUPRC")
plotConsensusTracks(sia)
```

`plotDiffIa` *plotDiffIa*

Description

This function creates a karyoplot with the differential interactions calls.

Usage

```
plotDiffIa(
  ia,
  genes_of_interest = NULL,
  cex.chr = 1,
  cex.y.lab = 0.6,
  cex.ideo = 0.6,
  cex.y.track = 0.6,
  cex.vp = 1,
  cex.leg = 0.6,
  plot_spider = FALSE,
  highlight_regions = NULL,
  gene.name.cex = 1
)
```

Arguments

`ia`	fourSynergy object with interactions from all base tools (peakC, r3c-seq, fourSig, r4cker) and more relevant information.
`genes_of_interest`	Vector with genes of interest. Set to `all` if you want to plot all genes in this area.
`cex.chr`	character expansion of chromosome label.
`cex.y.lab`	character expansion for y labels.
`cex.ideo`	character expansion base numbers of ideogram.
`cex.y.track`	character expansion y axis track.
`cex.vp`	character expansion viewpoint label.
`cex.leg`	character expansion for legend.
`plot_spider`	plotting connections from VP to interactions
`highlight_regions`	regions to highlight in the plot
`gene.name.cex`	character expansion for gene names.

Value

DESeq2 results of differential interaction calling.

Examples

```
config <- system.file("extdata", "Datasets", "Demo", "info.yaml",
    package = "fourSynergy"
)
res_path <- system.file("extdata", "results", "Demo",
    package = "fourSynergy"
)
tracks <- system.file("extdata", "results", "Demo", "alignment",
    package = "fourSynergy"
)
sia <- createIa(res_path = res_path, config = config, tracks = tracks)
sia <- consensusIa(ia = sia, model = "AUPRC")
sia <- differentialAnalysis(ia = sia, fitType = "mean")
plotDiffIa(ia = sia)
```

`plotIaIndiviualTools` *This function creates a karyoplot with the interactions calls of the individual tools.*

Description

This function creates a karyoplot with the interactions calls of the individual tools.

Usage

```
plotIaIndiviualTools(
  ia,
  genes_of_interest = NULL,
  cex.chr = 1,
  cex.ideo = 0.6,
  cex.y.track = 0.6,
  cex.y.lab = 0.6,
  cex.vp = 1,
  cex.leg = 0.6,
  highlight_regions = NULL,
  gene.name.cex = 1
)
```

Arguments

`ia`	fourSynergy object with interactions from all base tools (peakC, r3c-seq, fourSig, r4cker) and other relevant information.
`genes_of_interest`	Vector with genes of interest. Set to `all` if you want to plot all genes in this area.
`cex.chr`	character expansion of chromosome label.
`cex.ideo`	character expansion base numbers of ideogram.
`cex.y.track`	character expansion y axis track.
`cex.y.lab`	character expansion y lab.
`cex.vp`	character expansion viewpoint label.
`cex.leg`	character expansion for legend.
`highlight_regions`	regions to highlight in the plot
`gene.name.cex`	character expansion for gene names.

Value

karyoplot with calling results.

Examples

```
config <- system.file("extdata", "Datasets", "Demo", "info.yaml",
    package = "fourSynergy"
)
res_path <- system.file("extdata", "results", "Demo",
    package = "fourSynergy"
)
tracks <- system.file("extdata", "results", "Demo", "alignment",
    package = "fourSynergy"
)
sia <- createIa(res_path = res_path, config = config, tracks = tracks)
plotIaIndiviualTools(ia = sia)
```

`plotpreTracks` *plotpreTracks*

Description

plotpreTracks

Usage

```
plotpreTracks(ia, highlight_regions = NULL)
```

Arguments

`ia`	fourSynergy object with interactions from all base tools.
`highlight_regions`	regions to highlight in the plot.

Value

list with read counts and Granges of bedfiles

`plotRegions` *Internal function to highlight regions in karyoplot.*

Description

Internal function to highlight regions in karyoplot.

Usage

```
plotRegions(ia, kp, highlight_regions)
```

Arguments

`ia`	fourSynergy object with interactions from all base tools (peakC, r3c-seq, fourSig, r4cker) and other relevant information.
`kp`	karyoplot.
`highlight_regions`	regions to highlight in the plot.

Value

karyoplot base

`plotTracks` *Internal function to plot tracks*

Description

Internal function to plot tracks

Usage

```
plotTracks(ia, kp, bgs, r0 = 0, r1 = 1, cex.vp = 1, cex.y.track = 0.6)
```

Arguments

`ia`	fourSynergy object with interactions from all base tools (peakC, r3c-seq, fourSig, r4cker) and other relevant information.
`kp`	Kayroplot object.
`bgs`	GrangesList of bedGraphs.
`r0`	r0 defines the vertical range of the data panel.
`r1`	r1 defines the vertical range of the data panel.

Value

karyoplot with tracks

`readAndTag` *readAndTag*

Description

Internal function to read base tools solutions in FourSynergy format.

Usage

```
readAndTag(file_path, tag, org)
```

Arguments

`file_path`	Path to the .bed files (`results/[dataset]/nearbait_area.bed`).
`tag`	tool name.
`org`	organism.

Value

GRanges with interaction calls and tool name as mcol.

`readBedGraph` *Internal function to read bedGraphs*

Description

Internal function to read bedGraphs

Usage

```
readBedGraph(ia)
```

Arguments

`ia`	fourSynergy object with interactions from all base tools. (peakC, r3c-seq, fourSig, r4cker) and other relevant information.

Value

GrangesList of bedGraph content

Index