Package 'IWTomics'

November 12, 2026

Type Package

Version 1.36.0

Date 2026-01-05

Title Interval-Wise Testing for Omics Data

Author Marzia A Cremona, Alessia Pini, Francesca Chiaromonte, Simone Vantini

Maintainer Marzia A Cremona <mac78@psu.edu>

Description Implementation of the Interval-Wise Testing (IWT) for omics data. This inferential procedure tests for differences in ``Omics'' data between two groups of genomic regions (or between a group of genomic regions and a reference center of symmetry), and does not require fixing location and scale at the outset.

Depends GenomicRanges

Imports parallel,gtable,grid,graphics,methods,IRanges,KernSmooth,fda,S4Vectors,grDevices,stats,utils,tools

Suggests XVector, knitr

VignetteBuilder knitr

biocViews StatisticalMethod, MultipleComparison, DifferentialExpression, DifferentialMethylation, DifferentialPeakCalling, GenomeAnnotation, DataImport

License GPL (>=2)

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

git_branch RELEASE_3_23

git_last_commit 2df7882

git_last_commit_date 2026-04-28

Repository Bioconductor 3.23

Date/Publication 2026-11-11

Contents

IWTomics-package 2
ETn_example 4

features_example 5
IWTomics-internal 6
IWTomicsData-class 6
IWTomicsTest 11
plot-IWTomicsData 15
plotSummary 18
plotTest 21
regionsFeatures_center 23
regionsFeatures_scale 24
regions_example 24
smooth-IWTomicsData 25

Index **28**

`IWTomics-package` *Interval-Wise Testing for Omics Data*

Description

Implementation of the Interval-Wise Testing for "Omics" data, an extended version of the Interval-Wise Testing for functional data presented in Pini and Vantini (2017). This inferential procedure tests for differences in "Omics" data between two groups of genomic regions (or between a group of genomic regions and a reference curve), and does not require fixing location and scale at the outset.

Author(s)

Marzia A Cremona, Alessia Pini, Francesca Chiaromonte, Simone Vantini

Maintainer: Marzia A Cremona <mac78@psu.edu>

References

A Pini and S Vantini (2017). Interval-Wise Testing for functional data. *Journal of Nonparametric Statistics*.

Examples

```
## ------------------------------------------------------------------------------------------
## ------------------------------------------------------------------------------------------
## EXAMPLE ON REAL DATA
## ------------------------------------------------------------------------------------------
## ------------------------------------------------------------------------------------------
## ETn Recombination hotspots data
## Two region datasets:
## ETns fixed 64-kb flanking regions and 64-kb control regions in mouse
## One feature measured in 1-kb windows:
## Recombination hotspots content
## ?ETn_example for details on the dataset
data(ETn_example)
ETn_example
```

```
## ------------------------------------------------------------------------------------------
## PLOT DATA
## ------------------------------------------------------------------------------------------
## Plot the pointwise boxplot and averages of the curves in ETn and control regions
## (note that the box of the pointwise boxplot is zero, but the average is not)
plot(ETn_example)

## ------------------------------------------------------------------------------------------
## PERFORM THE TEST
## ------------------------------------------------------------------------------------------
## Two sample test to compare recombination hotspots
## in ETn regions vs control regions
ETn_test=IWTomicsTest(ETn_example,
                      id_region1='ETn_fixed',id_region2='Control')
## Adjusted p-value
adjusted_pval(ETn_test)

## Plotting the test results
plotTest(ETn_test)

## Adjusted p-value lowering the scale of the test
adjusted_pval(ETn_test,scale_threshold=10)

## Plotting the test results, lowering the scale of the test
plotTest(ETn_test,scale_threshold=10)

## Summary plot of the two sample test
## x11(12,2)
plotSummary(ETn_test,groupby='feature',scale_threshold=10,
            align_lab='Integration site')


## ------------------------------------------------------------------------------------------
## ------------------------------------------------------------------------------------------
## EXAMPLE ON SIMULATED DATA
## ------------------------------------------------------------------------------------------
## ------------------------------------------------------------------------------------------
examples_path <- system.file("extdata",package="IWTomics")
## Four region datasets:
## three different types of elements and one control
datasets=read.table(file.path(examples_path,"datasets.txt"),
                    sep="\t",header=TRUE,stringsAsFactors=FALSE)
datasets
## Two different features measured in all four types regions
features_datasetsTable=read.table(file.path(examples_path,"features_datasetsTable.txt"),
                                  sep="\t",header=TRUE,stringsAsFactors=FALSE)
features_datasetsTable

## ------------------------------------------------------------------------------------------
## GET DATA AND PLOT
## ------------------------------------------------------------------------------------------
## Get genomic regions for the four region datasets,
```

```
## and the two features from Table files for each region dataset
regionsFeatures=IWTomicsData(datasets$regionFile,features_datasetsTable[,3:6],'center',
                             datasets$id,datasets$name,
                             features_datasetsTable$id,features_datasetsTable$name,
                             path=file.path(examples_path,'files'))

## Plot the pointwise boxplot of the curves in the different region datasets
plot(regionsFeatures)

## ------------------------------------------------------------------------------------------
## PERFORM THE TEST
## ------------------------------------------------------------------------------------------
## Two sample test for the two features in the comparisons
## 'elem1' vs 'control', 'elem2' vs 'control'
## and 'elem3' vs 'control'
regionsFeatures_test=IWTomicsTest(regionsFeatures,id_region1=c('elem1','elem2','elem3'),
                                            id_region2=c('control','control','control'))
## Adjusted p-value for each comparison and each feature
adjusted_pval(regionsFeatures_test)

## Plotting the results of the two sample test
plotTest(regionsFeatures_test)

## Summary plot of the two sample tests
## x11(10,5)
plotSummary(regionsFeatures_test,groupby='feature',align_lab='Center')
```

`ETn_example` *ETn Recombination hotspots data*

Description

Recombination hotspots data in the regions surrounding fixed ETns (elements of the Early Transposon family of active Endogenous Retroviruses in mouse) and in control regions.

Usage

```
data(ETn_example)
```

Format

`ETn_example` is an object of class `"IWTomicsData"`, with two region datasets "ETn fixed" and "Control" with center alignment and one feature "Recombination hotspots content".

In particular, the region dataset "ETn fixed" contains 1296 genomic regions of 64 kb surrounding fixed ETns elements (32-kb flanking sequences upstream and 32-kb flanking sequences downstream of each element). The region dataset "Control" contains 1142 regions of 64 kb without elements, used as control in the test. The regions are aligned around their center (i.e. around the ETn integration sites).

Recombination hotspots measurements are associated to each "ETn fixed" and "Control" region. In particular, this feature is measured in 1-kb windows, so that each region is associated to a recombination hotspots curve made of 64 values. The measurement used is the feature content, i.e. the fraction of the 1-kb window that is covered by recombination hotspots.

Value

`"IWTomicsData"` object.

Source

Data have been collected and pre-processed by: R Campos-Sanchez, MA Cremona, A Pini, F Chiaromonte and KD Makova (2016). Integration and fixation preferences of human and mouse endogenous retroviruses uncovered with Functional Data Analysis. *PLoS Computational Biology*. 12(6): 1-41.

Fixed ETn positions come from: Y Zhang, IA Maksakova, L Gagnier, LN van de Lagemaat, DL Mager (2008). Genome-wide assessments reveal extremely high levels of polymorphism of two active families of mouse endogenous retroviral elements. *PLoS Genetics*. 4: e1000007.

Recombination hotspots data come from: H Brunschwig, L Levi, E Ben-David, RW Williams, B Yakir, S Shifman (2012). Fine-scale maps of recombination rates and hotspots in the mouse genome. *Genetics*. 191: 757-764.

Examples

```
data(ETn_example)
ETn_example

plot(ETn_example,type='boxplot')
```

---

features_example *Example of features*

Description

Example of features used to illustrate usage of `"IWTomicsData"` object constructor. It contains measurements of two different genomic features, "ftr1" and "ftr2", corresponding to four different region datasets: "elem1", "elem2", "elem3" and "control".

Usage

```
data(features_example)
```

Format

Each element of the list is a list of matrices with measurements corresponding to four different region datasets: "elem1", "elem2", "elem3" and "control". Columns correspond to measurements in a region.

Value

list of two elements, one for each of the two features "ftr1" and "ftr2".

Examples

```
data(features_example)
```

`IWTomics-internal` *IWTomics internal functions*

Description

IWTomics internal functions

`IWTomicsData-class` *Class* `"IWTomicsData"`

Description

The class `"IWTomicsData"` defines a container for storing a collection of aligned genomic region datasets, and their associated feature measurements, to be used as input for the Interval-Wise Testing of "Omics" data.

Details

An object of class `"IWTomicsData"` is a list of genomic locations organized in different region datasets and aligned (e.g. around their center). Multiple genomic feature measurements are associated to each location. In particular, each feature is measured in windows of fixed size inside each location. As a consequence, a vector of measurement is associated to each pair of locations and features. This information is stored in the following slots:

`metadata`**:** list with `region_datasets` and `feature_datasets` components. The component `region_datasets` is a data frame with names, file names and size of each region dataset. The component `feature_datasets` is a data frame with names, file names and resolution of each feature.

`regions`**:** ?`"GRangesList"` object containing the genomic locations of each region dataset.

`alignment`**:** string indicating the region alignment type. Can be `"left"`, `"right"`, `"center"` or `"scale"`.

`features`**:** list of matrix lists, with columns of aligned feature measurments corresponding to each feature in each region dataset.

`length_features`**:** list of vector lists, with the number of measurments corresponding to each feature in each region dataset.

`test`**:** (optional) list with `input` and `result`, containing test input and results. In particular, `input` is a list with components:

- `id_region1`: identifier(s) of the region dataset(s) tested.

- id_region2: identifier(s) of the region dataset(s) tested for two sample test.
- id_features_subset: vector with the identifiers of the features tested.
- mu: the center of symmetry under the null hypothesis in one sample test, or the difference between the two populations in two sample test.
- statistics: test statistics used in the test.
- probs: probabilities corresponding to the quantiles in test statistics "quantile".
- max_scale: the maximum interval length used for the p-value adjustment.
- paired: if TRUE, the test was paired.
- B: number of permutation used in the test.

Each element of the list result is a list of test results for the features tested. Each test result is a a list with components:

- test: string vector indicating the type of test performed, "1pop" or "2pop" for one sample and two sample tests, respectively.
- mu: the center of symmetry under the null hypothesis in one sample test, or the difference between the two populations in two sample test, for the particular test considered.
- max_scale the maximum interval length used for the p-value adjustment.
- T0_plot: value of the test statistics without squaring (used by plotSummary to draw the summary plot).
- adjusted_pval: adjusted p-value curve, i.e. adjusted p-values for each point of the curves. The adjustment is done considering max_scale as length.
- adjusted_pval_matrix: matrix of size the number of points in the curves with the adjusted p-value curves for each possible scale up to max_scale. Row i of the matrix contains the adjusted p-value curve with correction done up to scale p-i+1 (the matrix contains NA for scale greater than max_scale).
- unadjusted_pval: p-value curve, i.e. raw p-values for each point of the curves.
- pval_matrix: matrix of size the number of points in the curves with the raw p-values of the multivariate tests. The element (i,j) of the matrix contains the p-value of the joint NPC test of the components j,j+1,...,j+(p-i) (the matrix contains NA for scale greater than max_scale).
- exact: logical value indicating whether the exact p-values have been computed.
- notNA: vector of logical vectors indicated the points of the curves where the test was performed (used by plotSummary to draw the summary plot).

Objects of class "IWTomicsData" can be initialized from BED or Table files, or they can be directly created by supplying a "GRangesList" of genomic region datasets and a "list" with the aligned feature measurements (see Contructors). The optional slot test is filled by the function IWTomicsTest that performs the Interval-Wise Testing.

Constructors

IWTomicsData(x, y, alignment='center', id_regions=NULL, name_regions=NULL, id_features=NULL, name_features=NULL, path=NULL, start.are.0based=TRUE, header=FALSE, ...): creates a "IWTomicsData" object from BED or Table files.

IWTomicsData(x, y, alignment='center', id_regions=NULL, name_regions=NULL, id_features=NULL, name_features=NULL, length_features=NULL): creates a "IWTomicsData" object from genomic regions datasets and feature measurements.

`x` vector with the names of the region files containing the region datasets to be loaded. BED and Table formats currently supported. *Alternative constructor*: `"GRangesList"` object with genomic locations of each region dataset.

`y` vector with the names of the feature files, or dataframe with columns of feature file names corresponding to the different region datasets. Each feature must be measured in windows of a fixed size inside all the regions. BED and Table formats currently supported: either a row (with 4 columns chr, start, end, measure) for each window, or a row for each region (with columns chr, start, end, value1, ..., valueN). Note that all files must be sorted. *Alternative constructor*: list of matrix lists, with columns of aligned feature measurments corresponding to each feature in each region dataset.

`alignment` region alignment. Possible types are:

- `"left"` for alignment of the starting positions,
- `"right"` for alignment of the ending positions,
- `"center"` for alignment of the central positions (default),
- `"scale"` for scaling all regions to the same length.

`id_regions` vector with the identifiers of the region datasets. If `NULL`, `file_regions` or `names(regions)` are used.

`name_regions` vector with the names of the region datasets to be used in the output. If `NULL`, the identifiers `id_regions` are used.

`id_features` vector with the identifiers of the features. If `NULL`, `file_features` or `names(features)` are used.

`name_features` vector with the names of the features to be used in the output plots. If `NULL`, the identifiers `id_features` are used.

`path` the directory that contains the files. If `NULL`, the current working directory is used.

`start.are.0based` if `TRUE` (default) the start position in the region files are considered to be 0-based, and converted to 1-based in the `"IWTomicsData"` object in output.

`header` `TRUE` or `FALSE` (default) indicating if the files contain the names of the variables as their first lines.

`length_features` list of vector lists, with the number of measurments corresponding to each feature in each region dataset.

`...` additional parameters in input to `read.delim`.

Accessors

In the following code, `x` is a `"IWTomicsData"` object.

`nRegions(x)`: get the number of region datasets.

`nFeatures(x)`: get the number of features.

`dim(x)`: get the dimension of the object (number of region datasets, number of features).

`lengthRegions(x)`: get the number of locations in each region dataset.

`lengthFeatures(x)`: get a list of vector list, with the number of measurements corresponding to each feature in each region dataset.

`resolution(x)`: get the measurement resolution for each feature.

`metadata(x)`: get the metadata associated with the object, i.e. a list with `region_datasets` and `feature_datasets` components.

`regions(x)`: get the `"GRangesList"` object containing the genomic locations of each region dataset.

`features(x)`: get a list of matrix lists, with columns of aligned feature measurements corresponding to each feature in each region dataset.

`idRegions(x)`: get the identifiers of the region datasets.

`idFeatures(x)`: get the identifiers of the features.

`nameRegions(x)`: get the names of the region datasets.

`nameFeatures(x)`: get the names of the features.

`alignment(x)`: get the region alignment.

`testInput(x)`: get the test input (if present).

`nTests(x)`: get the number of tests present.

`idRegionsTest(x)`, `idRegionsTest(x,test)`: get the identifiers of the region datasets in the different tests. The (optional) argument `test` indicates the indices of the tests to be considered.

`idFeatuersTest(x)`: get the identifiers of the features tested.

`adjusted_pval(x)`, `adjusted_pval(x,test,id_features_subset,scale_threshold)`: get the adjusted p-values of the different tests. The (optional) argument `test` indicates the indices of the tests to be considered. The (optional) argument `id_features_subset` is a vector with the identifiers of the features to be cosidered. The (optional) argument `scale_threshold` is the threshold on the test scale (maximum interval length for the p-value adjustment) for the adjusted p-value computation. Can be either a scalar (the same length for all features) or a vector (a length for each feature) or a list of vectors (a vector for each test). See `IWTomicsTest` for more details.

Subsetting

In the following code, `x` is a `"IWTomicsData"` object. The optional slot `test`, if present in `x`, is deleted when using subsetting methods.

`x[i,j]`: extract region dataset `i` and feature `j` in a new `"IWTomicsData"` object. Both `i` and `j` can be logical vectors, numeric vectors, character vectors (with region dataset and feature identifiers, respectively), or missing.

Combining

In the following code, `x` is a `"IWTomicsData"` object. The optional slot `test`, if present in `x`, is deleted when using combining methods.

`c(x,...)` and `merge(x,...)`: create a new `"IWTomicsData"` object combining `x` with the `"IWTomicsData"` objects in `....` Any object in `...` must have the same region alignment as `x`, and region datasets and features present in multiple objects must coincide.

`rbind(x,...)`: create a new `"IWTomicsData"` object combining the features in `x` with the features in the `"IWTomicsData"` objects `....` Region datasets in `x` and any object in `...` must coincide and have the same region alignment.

`cbind(x,...)`: create a new `"IWTomicsData"` object combining the region datasets in `x` with the region datasets in the `"IWTomicsData"` objects `....` Features in `x` and any object in `...` must coincide and the `"IWTomicsData"` objects must have the same region alignment.

Other methods

`show(x)`: The `show` method prints the number of region datasets, their alignment type and the number of features in the `"IWTomicsData"` object. It also displays names and size of the region datasets, and names and resolution of the features. If the slot `test` is present in `x`, the `show` method prints also the comparisons present.

Author(s)

Marzia A Cremona, Alessia Pini, Francesca Chiaromonte, Simone Vantini

See Also

`plot` method to plot `"IWTomicsData"` objects; `smooth` method to smooth curves in `"IWTomicsData"` objects; `IWTomicsTest` for the Interval-Wise Testing.

Examples

```
examples_path <- system.file("extdata",package="IWTomics")
datasets=read.table(file.path(examples_path,"datasets.txt"),
                    sep="\t",header=TRUE,stringsAsFactors=FALSE)
features_datasetsBED=read.table(file.path(examples_path,"features_datasetsBED.txt"),
                                sep="\t",header=TRUE,stringsAsFactors=FALSE)
features_datasetsTable=read.table(file.path(examples_path,"features_datasetsTable.txt"),
                                  sep="\t",header=TRUE,stringsAsFactors=FALSE)
data(regions_example)
data(features_example)

## ------------------------------------------------------------------------------------------
## CONSTRUCTION
## ------------------------------------------------------------------------------------------
## Get genomic regions for four region datasets,
## and two features for each region dataset

## From BED files (check for consistency, time consuming)
regionsFeaturesBED=IWTomicsData(datasets$regionFile,features_datasetsBED[,3:6],
                                'center',datasets$id,datasets$name,
                                features_datasetsBED$id,features_datasetsBED$name,
                                path=file.path(examples_path,'files'))
regionsFeaturesBED

## From Table files (less checks for consistency, more efficient)
regionsFeaturesTable=IWTomicsData(datasets$regionFile,features_datasetsTable[,3:6],
                                  'center',datasets$id,datasets$name,
                                  features_datasetsTable$id,features_datasetsTable$name,
                                  path=file.path(examples_path,'files'))
regionsFeaturesTable

## From genomic regions datasets and feature measurements.
regionsFeatures=IWTomicsData(regions_example,features_example,alignment='center')
regionsFeatures
```

```
## ------------------------------------------------------------------------------------------
## SUBSETTING
## ------------------------------------------------------------------------------------------
## Extract a subset of region datasets and/or of features

## Get the first region dataset and the second features
regionsFeaturesBED[1,2]

## Get the first region dataset and the second features, using identifiers
regionsFeaturesBED['elem1','ftr2']

## Get the first two region datasets for all the features
regionsFeaturesBED[1:2,]

## Get all region datasets for the first feature
regionsFeaturesBED[,1]

## ------------------------------------------------------------------------------------------
## COMBINING
## ------------------------------------------------------------------------------------------
data1=regionsFeaturesBED[1:2,1]
data2=regionsFeaturesBED[1:2,2]
data3=regionsFeaturesBED[2:3,]
data4=regionsFeaturesBED[4,]

## Merge different objects
data1
data2
c(data1,data2)
merge(data1,data2)

## Combine different features
data1
data2
cbind(data1,data2)

## Combine different regions
data3
data4
rbind(data3,data4)

## Combine methods together
rbind(cbind(data1,data2),data3,data4)
```

`IWTomicsTest` *Interval-Wise Testing*

Description

The function implements the Interval-Wise Testing for omics data (both one sample and two sample tests), an extended version of the Interval-Wise Testing for functional data presented in Pini and

Vantini (2017). This inferential procedure tests for differences in feature measurements between two region datasets (two sample test) or between a region dataset and a reference curve (one sample test).

Usage

```
IWTomicsTest(regionsFeatures,
  id_region1=idRegions(regionsFeatures)[1], id_region2=NULL,
  id_features_subset=idFeatures(regionsFeatures), mu=0,
  statistics="mean", probs=0.5, max_scale=NULL, paired=FALSE, B=1000)
```

Arguments

regionsFeatures	"IWTomicsData" object.
id_region1	identifier(s) of the region dataset(s) to be tested. If a vector is provided, a test will be performed for each element of the vector.
id_region2	identifier(s) of the region dataset(s) to be tested for two sample test. If NULL or empty string, one sample test is performed.
id_features_subset	vector with the identifiers of the features to be tested.
mu	the reference curve (center of symmetry) under the null hypothesis in one sample test, or the difference between the two populations in two sample test. Can be either a constant (the same constant curve for all features), a vector of constants (a constant curve for each feature), or a list of vectors containing its measurements for each feature (on the same grid as the features). Default mu=0.
statistics	test statistics to be used in the test. Possible test statistics are "mean", "median", "variance" and "quantile".
probs	probabilities corresponding to the quantiles in test statistics "quantile". If multiple quantiles are selected, the test statistics is the sum of the statistics on the different quantiles.
max_scale	the maximum interval length to be used for the p-value adjustment, i.e. the maximum number of consecutive windows to be employed (can range from 1 to the maximum number of consecutive measurements present for the feature tested). Can be either a scalar (the same length for all features) or a vector (a length for each feature) or a list of vectors (a vector for each test). If NULL, the maximum possible interval length is used.
paired	if TRUE, a paired (two sample) test is performed.
B	number of iterations of the MC algorithm to evaluate the p-values of the permutation tests. If B is greater than the number of possible permutations, exact p-values are computed.

Value

The function IWTomicsTest returns an object of class "IWTomicsData" containing the region datasets and feature datasets tested, and with the test input and result in the optional slot test.

Note

In this implementation of Interval-Wise Testing, the smallest scale considered corresponds to the measurement resolution, i.e. the univariate (unadjusted) tests are done on each measurement window. To change the smallest scale considered the method `smooth` can be employed.

If the region alignment is `"scale"`, the function `smooth` must be used before applying the Interval-Wise Testing in order to measure the features over the grid in all the regions.

Author(s)

Marzia A Cremona, Alessia Pini, Francesca Chiaromonte, Simone Vantini

References

A Pini and S Vantini (2017). Interval-Wise Testing for functional data. *Journal of Nonparametric Statistics*.

See Also

`IWTomicsData` for `"IWTomicsData"` class, constructors, accessors and methods; `plot` method to plot `"IWTomicsData"` objects; `smooth` to smooth curves before testing; `plotTest` to plot the test results; `plotSummary` to draw a summary plot of the test results.

Examples

```
## ------------------------------------------------------------------------------------------
## ------------------------------------------------------------------------------------------
## EXAMPLE ON REAL DATA
## ------------------------------------------------------------------------------------------
## ------------------------------------------------------------------------------------------
## ETn Recombination hotspots data
## Two region datasets:
## ETns fixed 64-kb flanking regions and 64-kb control regions in mouse
## One feature measured in 1-kb windows:
## Recombination hotspots content
## ?ETn_example for details on the dataset
data(ETn_example)
ETn_example

## Two sample test to compare recombination hotspots
## in ETn regions vs control regions
ETn_test=IWTomicsTest(ETn_example,
                      id_region1='ETn_fixed',id_region2='Control')
## Adjusted p-value
adjusted_pval(ETn_test)

## Adjusted p-value lowering the scale of the test
adjusted_pval(ETn_test,scale_threshold=10)


## ------------------------------------------------------------------------------------------
## ------------------------------------------------------------------------------------------
```

```
## EXAMPLE ON SIMULATED DATA
## ------------------------------------------------------------------------------------------
## ------------------------------------------------------------------------------------------

## ------------------------------------------------------------------------------------------
## ------------------------------------------------------------------------------------------
## CURVE ALIGNMENT CENTER
## ------------------------------------------------------------------------------------------
## ------------------------------------------------------------------------------------------
data(regionsFeatures_center)

## One sample test for 'control' regions and feature 'ftr1'
regionsFeatures_test=IWTomicsTest(regionsFeatures_center,
                                  id_region1='control',id_features_subset='ftr1')
adjusted_pval(regionsFeatures_test)

## Plotting the results of the one sample test
plotTest(regionsFeatures_test,col=5)


## Two sample test for 'elem1', 'elem2' and 'elem3' vs 'control' regions and feature 'ftr1'
regionsFeatures_test=IWTomicsTest(regionsFeatures_center,
                                  id_region1=c('elem1','elem2','elem3'),
                                  id_region2=c('control','control','control'),
                                  id_features_subset='ftr1')
adjusted_pval(regionsFeatures_test)

## Plotting the results of the two sample test
plotTest(regionsFeatures_test)

## Summary plot of the two sample test
## x11(10,5)
plotSummary(regionsFeatures_test,groupby='feature',align_lab='Center')

##############################################################################################
## Not run:
##############################################################################################
## Using 'quantile' test statistics with multiple quantiles
regionsFeatures_test=IWTomicsTest(regionsFeatures_center,
                                  id_region1=c('elem1','elem2','elem3'),
                                  id_region2=c('control','control','control'),
                                  id_features_subset='ftr1',
                                  statistics='quantile',probs=c(0.25,0.75))
adjusted_pval(regionsFeatures_test)

## Plotting the results of the two sample test
plotTest(regionsFeatures_test)

## Summary plot of the two sample test
## x11(10,5)
plotSummary(regionsFeatures_test,groupby='feature',align_lab='Center')
##############################################################################################
## End(Not run)
```

```
##################################################################################################


## ------------------------------------------------------------------------------------------------
## ------------------------------------------------------------------------------------------------
## CURVE ALIGNMENT SCALE
## ------------------------------------------------------------------------------------------------
## ------------------------------------------------------------------------------------------------
data(regionsFeatures_scale)

## Smooth the curves to have measurements on the same grid
regionsFeatures_smooth=smooth(regionsFeatures_scale,type='locpoly',scale_grid=30)

## Two sample test for 'elem1', 'elem2' and 'elem3' vs 'control' regions and feature 'ftr2'
regionsFeatures_test=IWTomicsTest(regionsFeatures_smooth,
                                  id_region1=c('elem1','elem2','elem3'),
                                  id_region2=c('control','control','control'),
                                  id_features_subset='ftr2')
adjusted_pval(regionsFeatures_test)

## Plotting the results of the two sample test
plotTest(regionsFeatures_test)

## Summary plot of the two sample test
## x11(10,5)
plotSummary(regionsFeatures_test,groupby='feature')
```

---

plot-IWTomicsData *Plot* "IWTomicsData" *object*

Description

Method to plot objects of class "IWTomicsData". The function create graphical representations of the feature measurments in each region datasets, such as aligned curves, pointwise quantile curves, scatterplot and smoothed scatterplot.

Usage

```
## S4 method for signature 'IWTomicsData'
plot(x, type='boxplot',method='pearson',
  N_regions=pmin(lengthRegions(x),
                 ifelse(type=='curves',10,ifelse(type=='pairs',1000,+Inf))),
  probs=c(0.25,0.5,0.75), average=TRUE, size=TRUE,
  id_regions_subset=idRegions(x), id_features_subset=idFeatures(x),
  log_scale=FALSE, log_shift=0, col=1+seq_along(id_regions_subset),
  plot=TRUE, ask=TRUE, xlab='Windows', ylim=NULL,...)
```

Arguments

x	"IWTomicsData" object.
type	type of plot to be drawn. Possible types are: • "boxplot" for the pointwise quantiles curves (default), • "curves" for the (aligned) curves, • "pairs" for the scatterplot matrix of the different features (same resolution needed), • "pairsSmooth" for the smoothed scatterplot matrix of the different features (same resolution needed).
method	correlation coefficient to be computed and plotted if type="pairs" or type="pairsSmooth". Possible types are "pearson" (default), "kendall" and "spearman". See cor for details about the different methods.
N_regions	number of regions to be randomly chosen (for each region dataset) in "curves" and "pairs" type plot. Default plots a maximum of 10 and 1000 curves for "curves" and "pairs", respectively.
probs	probabilities corresponding to the quantiles to be drawn in "boxplot" type plot. Default plots quartile curves.
average	if TRUE, plot the mean curves.
size	if TRUE, plot the sample size in each position.
id_regions_subset	vector with the identifiers of the region datasets to be plotted.
id_features_subset	vector with the identifiers of the features to be plotted.
log_scale	plot the logarithm of the measurements.
log_shift	scalar with the shift to be applied before applying the logorithm, when log_scale=TRUE.
col	vector of plotting colors for the different region datasets. If type is "pairsSmooth", only the first element of col is used.
plot	it TRUE (default) a plot is drawn, otherwise a list of plot data is returned.
ask	if TRUE (default) the user is prompted before a new plot is drawn.
xlab	a title for the x axis.
ylim	the y limits of the plot.
...	additional plot parameters.

Value

plot returns a list with components depending on the plot type.

If type is "pairs" or "pairsSmooth" the list has components:

features_plot	matrix with the data plotted, one column for each feature considered.
features_cor	correlation matrix.
type	type of plot drawn.

If type is "boxplot" or "curves" the list has components:

`x_plot` a list of vectors with the abscissa for each feature.

`features_plot` if type is `"boxplot"`, a list of matrix lists with the data plotted for each feature and each region datasets. Each column represents one quantile drawn. If type is `"curves"`, a list of matrices with the data plotted for each feature. Each column represents one curve drawn.

`features_average` if `average` is `TRUE`, a list of matrices with the average curves for each feature. Each column represents the average curve for a region dataset.

`features_position_size` if `size` is `TRUE`, a list of matrices with the sample size for each feature. Each column represents the sample size in each position for a region dataset.

`type` type of plot drawn.

`col` vector of plotting colors for the different region datasets.

`col_plot` if type is `"boxplot"`, list of plotting colors for each feature (quantile curves colors and shaded band colors). If type is `"curves"`, vector of plotting colors correspondent to the different curves in `features_plot`.

Author(s)

Marzia A Cremona, Alessia Pini, Francesca Chiaromonte, Simone Vantini

See Also

`IWTomicsData` for `"IWTomicsData"` class, constructors, accessors and methods; `smooth` method to smooth curves in `"IWTomicsData"` objects; `IWTomicsTest` for the Interval-Wise Testing.

Examples

```
data(regionsFeatures_center)

## Plot the pointwise boxplot of the curves in the different region datasets
plot(regionsFeatures_center,type='boxplot')

## Plot all the curves in the different region datasets
plot(regionsFeatures_center,type='curves',N_regions=lengthRegions(regionsFeatures_center))

## Scatterplot of the measurements in the different region datasets
plot(regionsFeatures_center,type='pairs',N_regions=lengthRegions(regionsFeatures_center))

## Smooth scatterplot of the measurements in the 'control' region datasets
plot(regionsFeatures_center,type='pairsSmooth',id_regions_subset='control',
     N_regions=lengthRegions(regionsFeatures_center)['control'],col=5)
```

plotSummary *Summary plots of Interval-Wise Testing result*

Description

The function creates a graphical summary of the Interval-Wise Testing results. In particular, it draws a heatmap of the adjusted p-value curves at the chosen scale thresholds, grouped by the region datasets tested (test) or by the features tested (feature). Different raws in the heatmap correspond to different features (when grouping by locations) or to different tests (when grouping by feature). Color intensity is proportional to -log(p-value), i.e. darker colors correspond to lower p-values. Red means that the test statistics is higher in the first dataset tested than in the second one (or is positive in one sample test), while blue means that the test statistics is lower in the first datset tested than in the second one (or is negative in one sample test).

Usage

```
plotSummary(regionsFeatures, alpha=0.05, only_significant=FALSE,
  scale_threshold=NULL, nlevel=100, groupby='test',
  test=1:nTests(regionsFeatures), gaps_tests=NULL,
  id_features_subset=idFeaturesTest(regionsFeatures), gaps_features=NULL,
  ask=TRUE, filenames=NA, align_lab=NA, cellwidth=NA, cellheight=NA,
  xlab='Windows', ylab=ifelse(groupby=='test','Features','Tests'), ...)
```

Arguments

regionsFeatures	"IWTomicsData" object with test slot.
alpha	level of the hypothesis test used to select and display significant results. Default alpha=0.05.
only_significant	if TRUE, only the significant tests (p-value<=alpha in some position) will be plotted.
scale_threshold	threshold on the test scale (maximum interval length for the p-value adjustment) to be used in the adjusted p-value plot. Can be either a scalar (the same length for all features) or a vector (a length for each feature) or a list of vectors (a vector for each test). If NULL, the maximum possible interval length is used.
nlevel	number of desired color levels for the adjusted p-value heatmap.
groupby	how tests should be grouped for the summary plot. Possible types are: • "test" to draw a summary plot for all the tests about the same region dataset(s) comparison, • "feature" to draw a summary plot for all the tests about the same feature.
test	vector of indices of the tests to be plotted.
gaps_tests	vector of test indices that show where to put gaps in the heatmap between tests to be plotted. Only used if groupby is "feature".

id_features_subset	vector with the identifiers of the features to be plotted.
gaps_features	vector of id_features_subset indices that show where to put gaps in the heatmap between features. Only used if groupby is "test".
ask	if TRUE (default) the user is prompted before a new plot is drawn.
filenames	file paths where to save the plots (one for each group of tests as defined by groupby). Filetypes are decided by the extension in the paths. Currently the following formats are supported: png, pdf, tiff, bmp, jpeg.
align_lab	a title for the alignment point. Ignored if region alignment is "scale".
cellwidth	individual cell width in points.
cellheight	individual cell height in points.
xlab	a title for the x axis.
ylab	a title for the y axis.
...	additional plot parameters.

Value

No value returned. The function produces a graphical output.

Note

This function uses a modified version of **pheatmap** package (https://cran.r-project.org/package=pheatmap).

Author(s)

Marzia A Cremona, Alessia Pini, Francesca Chiaromonte, Simone Vantini

See Also

IWTomicsData for "IWTomicsData" class, constructors, accessors and methods; IWTomicsTest function to perform the Interval-wise Testing; plotTest to draw detailed plots of the test results.

Examples

```
## ------------------------------------------------------------------------------------------
## CURVE ALIGNMENT CENTER
## ------------------------------------------------------------------------------------------
data(regionsFeatures_center)

## One sample test for different regions and features
regionsFeatures_test=IWTomicsTest(regionsFeatures_center,
                                  id_region1=c('elem1','elem2','elem3','control'),mu=1)

## Summary plots grouped by feature
## x11(10,5)
plotSummary(regionsFeatures_test,groupby='feature')
```

```
## Set scale thresholds for the different features
## x11(10,5)
plotSummary(regionsFeatures_test,groupby='feature',
            scale_threshold=c(25,30))

## Add a title for the alignment point
## x11(10,5)
plotSummary(regionsFeatures_test,groupby='feature',
            scale.threshold=c(25,30),align_lab='Center')

## Plot only significant tests
## x11(10,5)
plotSummary(regionsFeatures_test,groupby='feature',
            scale.threshold=c(25,30),align_lab='Center',
            only_significant=TRUE)

## Summary plots grouped by test
## x11(10,5)
plotSummary(regionsFeatures_test,groupby='test')


## Two sample test for all possible region comparisons (mu=0),
## and one sample test for 'elem3' (mu=1) for feature 'ftr1'
regionsFeatures_test=IWTomicsTest(regionsFeatures_center,
                      id_region1=c('elem1','elem2','elem3',
                                   'elem1','elem1','elem2','elem1'),
                      id_region2=c('control','control','control',
                                   'elem2','elem3','elem3',''),
                      id_features_subset='ftr1')

## Summary plots grouped by feature
## x11(10,5)
plotSummary(regionsFeatures_test,groupby='feature')

## Put gaps between different types of test and add a title for the alignment point
## x11(10,5)
plotSummary(regionsFeatures_test,groupby='feature',
            gaps_tests=c(3,6),align_lab='Center')

## Plot only significant tests
## x11(10,5)
plotSummary(regionsFeatures_test,groupby='feature',
            gaps_tests=c(3,6),align_lab='Center',only_significant=TRUE)

## Plot only the first three tests
## x11(10,5)
plotSummary(regionsFeatures_test,groupby='feature',
            test=1:3,align_lab='Center')

## Change scale threshold for the first test
## x11(10,5)
plotSummary(regionsFeatures_test,groupby='feature',
            test=1:3,align_lab='Center',scale_threshold=list(t1=8,t2=50,t3=50))
```

plotTest *Detailed plots of Interval-Wise Testing result*

Description

The function plotTest creates graphical representations of the Interval-Wise Testing results present in the test slot of a "IWTomicsData" object. In particular, it draws a heatmap of the adjusted p-value functions at each possible scale (with the different maximum interval lengths considered), a plot of the adjusted p-value curve at the chosen scale threshold and a plot of the feature measurements in the region datasets tested (aligned curves or pointwise quantile curves).

Usage

```
plotTest(regionsFeatures, alpha=0.05, scale_threshold=NULL, nlevel=100, type="boxplot",
  N_regions=pmin(lengthRegions(regionsFeatures),10),
  probs=c(0.25,0.5,0.75), average=TRUE, size=TRUE,
  id_features_subset=idFeatures(regionsFeatures),
  col=1+seq_len(nRegions(regionsFeatures)),
  ask=TRUE, xlab="Windows", ylim=NULL, ...)
```

Arguments

regionsFeatures	"IWTomicsData" object with test slot.
alpha	level of the hypothesis test used to select and display significant results. Default alpha=0.05.
scale_threshold	threshold on the test scale (maximum interval length for the p-value adjustment) to be used in the adjusted p-value plot. Can be either a scalar (the same length for all features) or a vector (a length for each feature) or a list of vectors (a vector for each test). If NULL, the maximum possible interval length is used.
nlevel	number of desired color levels for the adjusted p-value heatmap.
type	type of plot to be drawn in addition to the adjusted p-value heatmap and plot. Possible types are: • "boxplot" for the pointwise quantiles curves (default), • "curves" for the (aligned) curves.
N_regions	number of regions to be randomly chosen (for each region dataset) in "curves" type plot. Default plots a maximum of 10 curves.
probs	probabilities corresponding to the quantiles to be drawn in "boxplot" type plot. Default plots quartile curves.
average	if TRUE, plot the mean curves.
size	if TRUE, plot the sample size in each position.
id_features_subset	vector with the identifiers of the features to be plotted.

col	vector of plotting colors for the different region datasets.
ask	if TRUE (default) the user is prompted before a new plot is drawn.
xlab	a title for the x axis.
ylim	the y limits of the feature measurement plot.
...	additional plot parameters.

Value

No value returned. The function produces a graphical output.

Author(s)

Marzia A Cremona, Alessia Pini, Francesca Chiaromonte, Simone Vantini

References

A Pini and S Vantini (2017). Interval-Wise Testing for functional data. *Journal of Nonparametric Statistics*.

See Also

IWTomicsData for "IWTomicsData" class, constructors, accessors and methods; IWTomicsTest function to perform the Interval-wise Testing; plotSummary to draw a summary plot of the test results.

Examples

```
data(regionsFeatures_center)

## One sample test for 'control' regions and feature 'ftr1'
regionsFeatures_test=IWTomicsTest(regionsFeatures_center,
                                  id_region1='control',id_features_subset='ftr1')

## Plotting the results of the one sample test
plotTest(regionsFeatures_test,col=5)

## Set the scale threshold to 25
plotTest(regionsFeatures_test,col=5,scale_threshold=25)

## Plot curves instead of boxplots, do not report sample size in each position
plotTest(regionsFeatures_test,col=5,type='curves',size=FALSE)


## Two sample test for 'elem1', 'elem2' and 'elem3' vs 'control' regions
regionsFeatures_test=IWTomicsTest(regionsFeatures_center,
                                  id_region1=c('elem1','elem2','elem3'),
                                  id_region2=c('control','control','control'))

## Plotting the results of the two sample test
plotTest(regionsFeatures_test)
```

```
## Set scale thresholds for the different features
plotTest(regionsFeatures_test,scale_threshold=c(25,30))

## Plot only results regarding feature 'ftr2', setting scale thresholds for each test
plotTest(regionsFeatures_test,id_features_subset='ftr2',
         scale_threshold=list(test1=10,test2=20,test3=30))

## Report also 0.05 and 0.95 quantiles in the boxplots
plotTest(regionsFeatures_test,id_features_subset='ftr2',
         scale_threshold=list(test1=10,test2=20,test3=30),
         probs=c(0.05,0.25,0.5,0.75,0.9))
```

`regionsFeatures_center` *Example of* `"IWTomicsData"` *object with center alignment*

Description

Example of `"IWTomicsData"` object with center alignment, used to illustrate usage of IWTomics package.

Usage

```
data(regionsFeatures_center)
```

Format

An object of class `"IWTomicsData"`, with four region datasets "Elements 1", "Elements2", "Elements 3" and "Controls" with center alignment and two features "Feature 1" and "Feature 2".

Value

`"IWTomicsData"` object.

Examples

```
data(regionsFeatures_center)
regionsFeatures_center
```

regionsFeatures_scale *Example of "IWTomicsData" object with scale alignment*

Description

Example of "IWTomicsData" object with scale alignment, used to illustrate usage of IWTomics package.

Usage

```
data(regionsFeatures_scale)
```

Format

An object of class "IWTomicsData", with four region datasets "Elements 1", "Elements2", "Elements 3" and "Controls" with scale alignment and two features "Feature 1" and "Feature 2".

Value

"IWTomicsData" object.

Examples

```
data(regionsFeatures_scale)
regionsFeatures_scale
```

regions_example *Example of regions*

Description

Example of regions used to illustrate usage of "IWTomicsData" object constructor. It contains four different region datasets: "elem1", "elem2", "elem3" and "control".

Usage

```
data(regions_example)
```

Format

A "GRangesList" object with genomic regions of each region dataset: "elem1", "elem2", "elem3" and "control".

Value

"GRangesList" object.

Examples

```
data(regions_example)
```

---

smooth-IWTomicsData *Smooth curves of a* "IWTomicsData" *object*

Description

The function allows to smooth the curves in a "IWTomicsData" object, to fill in gaps in the measurements, to change measurement resolutions and, when region alignment is "scale", to measure the features in the different regions on the same grid.

Usage

```
## S4 method for signature 'IWTomicsData'
smooth(x, type="locpoly",
  id_regions_subset=idRegions(x), id_features_subset=idFeatures(x),
  resolution_new=resolution(x)[id_features_subset],
  scale_grid=unlist(lapply(lengthFeatures(x)[id_features_subset],
    function(length) max(unlist(length[id_regions_subset])))),
  fill_gaps=TRUE, bandwidth=5, degree=3, dist_knots=10, parallel=FALSE)
```

Arguments

x	"IWTomicsData" object.
type	type of smoothing used. Possible types are: • "locpoly" for local polynomials (see help of locpoly function), • "kernel" for Nadaraya-Watson kernel smoothing with Gaussian kernel (see help of ksmooth function), • "splines" for regression b-splines (see create.bspline.basis and smooth.basis). Note that "splines" is very computational expensive when regions have different length and/or gaps.
id_regions_subset	vector with the identifiers of the region datasets to be smoothed.
id_features_subset	vector with the identifiers of the features to be smoothed.
resolution_new	resolution used to evaluate smoothed curves. As default, measurement resolution is mainteined after smoothing and smoothed curves are evaluated on the same grid as the raw curves. Can be either a constant (the same resolution for all features) or a vector of constants (a resolution for each feature). Not used if region alignment is "scale".
scale_grid	number of equally-spaced grid points over which the smoothed curves are evaluated. Can be either a constant (the same grid for all features) or a vector of constants (a grid for each feature). Only used if region alignment is "scale".

fill_gaps	if TRUE (default), smoothing is used to fill gaps (NA measurements). Can be set to FALSE only if region alignment is not "scale" and measurement resolution is mainteined.
bandwidth	bandwidth smoothing parameter used in "locpoly" or "kernel" type smoothing.
degree	degree of local polynomial or b-splines used in "locpoly" or "splines" type smoothing.
dist_knots	(approximate) distance between knots defining the b-splines used in "splines" type smoothing.
parallel	if TRUE smoothing is run in parallel on the different curves.

Value

smooth returns a "IWTomicsData" object with smoothed curves and updated resolutions and lengths (without the optional slot test).

Note

If the regions have different lengths and/or gaps are present in the feature measurements (NA measurements), "splines" type smoothing is very computational expensive. In this case we suggest to use "locpoly" or "kernel" type smoothing to smooth a large number of curves.

Author(s)

Marzia A Cremona, Alessia Pini, Francesca Chiaromonte, Simone Vantini

See Also

IWTomicsData for "IWTomicsData" class, constructors, accessors and methods; plot method to plot "IWTomicsData" objects; IWTomicsTest for the Interval-Wise Testing.

Examples

```
## ------------------------------------------------------------------------------------------
## ------------------------------------------------------------------------------------------
## CURVE ALIGNMENT CENTER
## ------------------------------------------------------------------------------------------
## ------------------------------------------------------------------------------------------
data(regionsFeatures_center)
dev.new()
plot(regionsFeatures_center,type='curves',N_regions=lengthRegions(regionsFeatures_center))

## Smooth all the curves with local polynomials
regionsFeatures_smooth=smooth(regionsFeatures_center,type='locpoly')
plot(regionsFeatures_smooth,type='curves',N_regions=lengthRegions(regionsFeatures_smooth))

## Smooth only feature 'ftr1', only region datasets 'elem3' and 'control'
regionsFeatures_smooth=smooth(regionsFeatures_center,type='locpoly',id_features_subset='ftr1',
                              id_regions_subset=c('elem3','control'))
plot(regionsFeatures_smooth,type='curves',N_regions=lengthRegions(regionsFeatures_smooth))
```

```
## Smooth only feature 'ftr1' and change its resolution
regionsFeatures_smooth=smooth(regionsFeatures_center,type='locpoly',id_features_subset='ftr1',
                              resolution_new=4000)
plot(regionsFeatures_smooth,type='curves',N_regions=lengthRegions(regionsFeatures_smooth))

## Smooth without filling gaps
regionsFeatures_smooth=smooth(regionsFeatures_center,type='locpoly',fill_gaps=FALSE)
plot(regionsFeatures_smooth,type='curves',N_regions=lengthRegions(regionsFeatures_smooth))


## ------------------------------------------------------------------------------------------
## ------------------------------------------------------------------------------------------
## CURVE ALIGNMENT SCALE
## ------------------------------------------------------------------------------------------
## ------------------------------------------------------------------------------------------
data(regionsFeatures_scale)
dev.new()
plot(regionsFeatures_scale,type='curves',N_regions=lengthRegions(regionsFeatures_scale),
                           average=FALSE,size=FALSE)

## Smooth all the curves with local polynomials
regionsFeatures_smooth=smooth(regionsFeatures_scale,type='locpoly')

## Plot of the smoothed curves
plot(regionsFeatures_smooth,type='curves',N_regions=lengthRegions(regionsFeatures_smooth),
     average=FALSE)
```

Index

∗ **classes**
IWTomicsData-class, 6
∗ **datasets**
ETn_example, 4
features_example, 5
regions_example, 24
regionsFeatures_center, 23
regionsFeatures_scale, 24
∗ **internal**
IWTomics-internal, 6
∗ **package**
IWTomics-package, 2
[,IWTomicsData,ANY,ANY,ANY-method
(IWTomicsData-class), 6

adjusted_pval *(IWTomicsData-class)*, 6
adjusted_pval,IWTomicsData,missing,character,missing-method
(IWTomics-internal), 6
adjusted_pval,IWTomicsData,missing,character,vector-method
(IWTomics-internal), 6
adjusted_pval,IWTomicsData,missing,missing,missing-method
(IWTomics-internal), 6
adjusted_pval,IWTomicsData,missing,missing,vector-method
(IWTomics-internal), 6
adjusted_pval,IWTomicsData,vector,character,missing-method
(IWTomics-internal), 6
adjusted_pval,IWTomicsData,vector,character,vector-method
(IWTomics-internal), 6
adjusted_pval,IWTomicsData,vector,missing,missing-method
(IWTomics-internal), 6
adjusted_pval,IWTomicsData,vector,missing,vector-method
(IWTomics-internal), 6
adjusted_pval,IWTomicsData-method
(IWTomics-internal), 6
alignment *(IWTomicsData-class)*, 6
alignment,IWTomicsData-method
(IWTomics-internal), 6

c,IWTomicsData-method
(IWTomicsData-class), 6
cbind,IWTomicsData-method
(IWTomicsData-class), 6
cor, *16*
create.bspline.basis, *25*

dim,IWTomicsData-method
(IWTomics-internal), 6

ETn_example, 4

features *(IWTomicsData-class)*, 6
features,IWTomicsData-method
(IWTomics-internal), 6
features_example, 5

GRangesList, *6*, *8*, *24*

idFeatures *(IWTomicsData-class)*, 6
idFeatures,IWTomicsData-method
(IWTomics-internal), 6
idFeaturesTest *(IWTomicsData-class)*, 6
idFeaturesTest,IWTomicsData-method
(IWTomics-internal), 6
idRegions *(IWTomicsData-class)*, 6
idRegions,IWTomicsData-method
(IWTomics-internal), 6
idRegionsTest *(IWTomicsData-class)*, 6
idRegionsTest,IWTomicsData,missing-method
(IWTomics-internal), 6
idRegionsTest,IWTomicsData,vector-method
(IWTomics-internal), 6
idRegionsTest,IWTomicsData-method
(IWTomics-internal), 6
IWTomics *(IWTomics-package)*, 2
IWTomics-internal, 6
IWTomics-package, 2
IWTomicsData, *5*, *12*, *13*, *15*, *17*, *19*, *21–26*
IWTomicsData *(IWTomicsData-class)*, 6
IWTomicsData,character,character-method
(IWTomicsData-class), 6

IWTomicsData,character,data.frame-method
(IWTomicsData-class), 6
IWTomicsData,character,matrix-method
(IWTomicsData-class), 6
IWTomicsData,GRangesList,list-method
(IWTomicsData-class), 6
IWTomicsData-class, 6
IWTomicsTest, *7*, *9*, *10*, 11, *17*, *19*, *22*, *26*

ksmooth, *25*

lengthFeatures (IWTomicsData-class), 6
lengthFeatures,IWTomicsData-method
(IWTomics-internal), 6
lengthRegions (IWTomicsData-class), 6
lengthRegions,IWTomicsData-method
(IWTomics-internal), 6
locpoly, *25*

merge,IWTomicsData,IWTomicsData-method
(IWTomicsData-class), 6
metadata (IWTomicsData-class), 6
metadata,IWTomicsData-method
(IWTomics-internal), 6

nameFeatures (IWTomicsData-class), 6
nameFeatures,IWTomicsData-method
(IWTomics-internal), 6
nameRegions (IWTomicsData-class), 6
nameRegions,IWTomicsData-method
(IWTomics-internal), 6
nFeatures (IWTomicsData-class), 6
nFeatures,IWTomicsData-method
(IWTomics-internal), 6
nRegions (IWTomicsData-class), 6
nRegions,IWTomicsData-method
(IWTomics-internal), 6
nTests (IWTomicsData-class), 6
nTests,IWTomicsData-method
(IWTomics-internal), 6

plot, *10*, *13*, *26*
plot (plot-IWTomicsData), 15
plot,IWTomicsData,ANY-method
(IWTomics-internal), 6
plot,IWTomicsData,missing-method
(IWTomics-internal), 6
plot,IWTomicsData-method
(plot-IWTomicsData), 15
plot-IWTomicsData, 15
plot.IWTomicsData (IWTomics-internal), 6
plotSummary, *7*, *13*, 18, *22*
plotTest, *13*, *19*, 21

rbind,IWTomicsData-method
(IWTomicsData-class), 6
read.delim, *8*
regions (IWTomicsData-class), 6
regions,IWTomicsData-method
(IWTomics-internal), 6
regions_example, 24
regionsFeatures_center, 23
regionsFeatures_scale, 24
resolution (IWTomicsData-class), 6
resolution,IWTomicsData-method
(IWTomics-internal), 6

show,IWTomicsData-method
(IWTomicsData-class), 6
smooth, *10*, *13*, *17*
smooth (smooth-IWTomicsData), 25
smooth,IWTomicsData-method
(smooth-IWTomicsData), 25
smooth-IWTomicsData, 25
smooth.basis, *25*

testInput (IWTomicsData-class), 6
testInput,IWTomicsData-method
(IWTomics-internal), 6