Package ‘geneRxCluster’

April 25, 2017

Date 2013-02-13
Version 1.12.0
License GPL (>= 2)

Description Detect Differential Clustering of Genomic Sites such as gene therapy integrations. The package provides some functions for exploring genomic insertion sites originating from two different sources. Possibly, the two sources are two different gene therapy vectors. Vectors are preferred that target sensitive regions less frequently, motivating the search for localized clusters of insertions and comparison of the clusters formed by integration of different vectors. Scan statistics allow the discovery of spatial differences in clustering and calculation of False Discovery Rates (FDRs) providing statistical methods for comparing retroviral vectors. A scan statistic for comparing two vectors using multiple window widths to detect clustering differentials and compute FDRs is implemented here.

Title gRx Differential Clustering
Author Charles Berry
Maintainer Charles Berry <ccberry@ucsd.edu>
Depends GenomicRanges,IRanges
Suggests RUnit, BiocGenerics
biocViews Sequencing, Clustering, Genetics

R topics documented:
critVal.alpha .......................................................... 2
critVal.power ......................................................... 3
critVal.target ......................................................... 4
geneRxCluster ......................................................... 5
gRxCluster ........................................................... 5
gRxCluster-object ................................................. 6
gRxPlot .............................................................. 7
gRxPlotClumps ....................................................... 8
gRxSummary ......................................................... 9
critVal.alpha

Description
critical region cutpoints

Usage
critVal.alpha(k, p0, alpha, posdiff)

Arguments

k - window width(s)
p0 - length 2 probabilities
alpha - two tailed
posdiff - position difference matrix

Details
This version uses alpha and will find TFD

Value
list of cutoffs and attributes

Author(s)
Charles Berry

See Also
gRxCluster for how and why this function is used

Examples
# symmetric odds:
crit <- critVal.alpha(5:25,c(1,1)/2,alpha=0.05,
                     matrix(1,nr=50,nc=21))
crit[[1]]
sapply(crit,c)
# 5:1 odds
asymmetric.crit <- critVal.alpha(5:25,c(1,5)/6,alpha=0.05,
                                  matrix(1,nr=50,nc=21))
# show the critical regions
par(mfrow=c(1,2))
gRxPlot(crit,method="critical")
gRxPlot(asymmetric.crit,method="critical")
rm(crit,asymmetric.crit)
critVal.power

<table>
<thead>
<tr>
<th>Description</th>
<th>critical region cutpoints</th>
</tr>
</thead>
<tbody>
<tr>
<td>Usage</td>
<td><code>critVal.power(k, p0, target, pwr = 0.8, odds = 7)</code></td>
</tr>
<tr>
<td>Arguments</td>
<td>k - window width(s)</td>
</tr>
<tr>
<td></td>
<td>p0 - length 2 probabilities</td>
</tr>
<tr>
<td></td>
<td>target - false discoveries wanted</td>
</tr>
<tr>
<td></td>
<td>pwr - desired power</td>
</tr>
<tr>
<td></td>
<td>odds - alternative odds ratio</td>
</tr>
<tr>
<td>Details</td>
<td>This version uses power and TFD and will limit windows screened</td>
</tr>
<tr>
<td>Value</td>
<td>list of cutoffs and attributes</td>
</tr>
<tr>
<td>Author(s)</td>
<td>Charles Berry</td>
</tr>
<tr>
<td>See Also</td>
<td><code>gRxCluster</code> for how and why this function is used</td>
</tr>
<tr>
<td>Examples</td>
<td># symmetric odds:</td>
</tr>
<tr>
<td></td>
<td><code>crit &lt;-</code></td>
</tr>
<tr>
<td></td>
<td><code>critVal.power(5:25,c(1,1),5,pwr=0.8,odds=7)</code></td>
</tr>
<tr>
<td></td>
<td><code>crit[[1]]</code></td>
</tr>
<tr>
<td></td>
<td><code>sapply(crit,c)</code></td>
</tr>
<tr>
<td></td>
<td># 5:1 odds</td>
</tr>
<tr>
<td></td>
<td><code>asymmetric.crit &lt;-</code></td>
</tr>
<tr>
<td></td>
<td><code>critVal.power(5:25,c(1,5),5,pwr=0.8,odds=7)</code></td>
</tr>
<tr>
<td></td>
<td># show the critical regions</td>
</tr>
<tr>
<td></td>
<td><code>par(mfrow=c(1,2))</code></td>
</tr>
<tr>
<td></td>
<td><code>gRxPlot(crit,method=&quot;critical&quot;)</code></td>
</tr>
<tr>
<td></td>
<td><code>gRxPlot(asymmetric.crit,method=&quot;critical&quot;)</code></td>
</tr>
<tr>
<td></td>
<td><code>rm(crit,asymmetric.crit)</code></td>
</tr>
</tbody>
</table>
critVal.target  critical regions

Description

critical region cutpoints

Usage

critVal.target(k, p0, target, posdiff = NULL, ns)

Arguments

  k  window width(s)
  p0  length 2 probabilities
  target  - two tailed
  posdiff  - position difference matrix
  ns  the number of windows passing filter at each k

Details

  This version uses TFD and will find alpha implicitly

Value

  list of cutoffs and attributes

Author(s)

  Charles Berry

See Also

  gRxCluster for how and why this function is used

Examples

  # symmetric odds:
  crit <- critVal.target(5:25,c(1,1),1,ns=rep(10,21))
  crit[[1]]
  sapply(crit,c)
  # 5:1 odds
  asymmetric.crit <- critVal.target(5:25,c(1,5),1,ns=rep(10,21))
  # show the critical regions
  par(mfrow=c(1,2))
  gRxPlot(crit,method="critical")
  gRxPlot(asymmetric.crit,method="critical")
  rm(crit,asymmetric.crit)
geneRxCluster  

**Description**

geneRxCluster provides the function \texttt{gRxCluster} and friends.

**Details**

Windows defined by \(k\) consecutive integration sites are scanned. A two class indicator is tallied to determine whether one class dominates. If one does, a flag is set and the window is retained. Various values of \(k\) are used. Conflicts between overlapping windows with the same value of \(k\) can occur — two windows are dominated by the two different classes. In that case, the sites of overlap are marked and neither window is retained. Conflicts can also arise between windows differing in their values of \(k\). In that case, the window having the smaller value of \(k\) is retained and the other is discarded.

Permutation tests and permutation based false discovery rates are available. Filtering of windows is allowed so that regions which are sparsely populated need not be studied.

gRxCluster

**Description**

cluster integration sites - optionally perform the permutations needed to estimate the discoveries expected under a null hypothesis

**Usage**

\[
gRxCluster(\texttt{object, starts, group, kvals, nperm = 0L,}
  \texttt{pruneFun = prune.loglik, ..., cutpt.filter.expr, cutpt.tail.expr, tmp.env,}
  \texttt{sample.id, sample.tab})
\]

**Arguments**

- \texttt{object}\hspace{1em} chromosome names or other grouping of starts
- \texttt{starts}\hspace{1em} ordered chromosome position or ordered integer vector
- \texttt{group}\hspace{1em} logical vector separating two groups
- \texttt{kvals}\hspace{1em} integer vector of window widths
- \texttt{nperm}\hspace{1em} number of permutations for FDR calculation
- \texttt{pruneFun}\hspace{1em} a function like \texttt{prune.loglik}.
- \texttt{\ldots}\hspace{1em} other args
- \texttt{cutpt.filter.expr}\hspace{1em} (optional) \texttt{R} object or call (or variable naming a call) with (optional) \texttt{var x (window widths in base pairs) to filter windows. It must evaluate to mode "double". If not specified, as.double(apply(x,2,median,na.rm=TRUE)) is used. If an atomic vector of length one is supplied it is expanded to the proper length and coerced to double. If this arg is the name of a variable provided in \texttt{tmp.env}, it must be protected with \texttt{quote(...)}.}
gRxCluster-object

cutpt.tail.expr
R object or call (or variable naming a call) with (optional) vars: k, n, and x (as above). Returns list like critVal.target. k is a vector of the number of sites in a collection of windows, and n is a vector of counts or proportions for the two classes of insertion. If not supplied, critVal.target(k, n, target=5, posdiff=x) is used. If this arg is the name of a variable provided in tmp.env, it must be protected with quote(...).

tmp.env
(optional) environment to use in evaluation of cutpt.* expressions. This is usually needed for critVal.power, which is first calculated and placed in the environment, and the supplied object is used in the expression for cutpt.filter.expr.

sample.id
(optional) integer vector indexing cells in sample.tab to be looked up to determine group under permutation. A factor can be used, too, but will be coerced to integer.

sample.tab
(optional) integer vector containing 0 or 1 in each cell. Its length is the same as max(sample.id). Both or neither sample.id and sample.tab should be supplied. When supplied sample.tab[sample.id] must equal group. If the arguments are supplied, permutations are of the form sample(sample.tab)[sample.id]. Otherwise they are of the form sample(group).

Value
a GRanges object with a special metadata slot, see gRxCluster-object

Author(s)
Charles Berry example inst/ex-gRxCluster.R

---

gRxCluster-object gRxCluster object

Description
Overview of the result of gRxCluster(...)

Details
The object returned is a GRanges object. If the object is x, seqnames(x) and ranges(x) slots demarcate the clusters discovered. There will be one element for each cluster (aka 'clump') discovered. Using the default argument pruneFun=prune.loglik or pruneFun=noprune, mcols(x) will have these columns:

value1 and value2 are the counts of the two classes of insertion sites for the clusters of object x clump.id numbers each cluster.

If the user supplies a custom pruneFun, it should return a GRanges with those columns and one element for each unique clump.id. The column target.min has the smallest nominal False Discoveries Expected for each cluster and is added to (or replaces) the mcols(x) produced by the argument supplied as pruneFun.

metadata(x) will include these components:
**criticalValues** A list object such as supplied by `critVal.target` whose elements each give the cutpoints to be used for a window with k sites. `attributes(metadata(object)$criticalValues[[i]])` will contain elements

- `fdr` with dimension c(k+1,4) of target false discovery expectations and and the one-sided p-values
- `target` the target for false discovery which sometimes is specified a priori and sometimes results from calculation
- `n` an upper bound on the number of windows to screen, if this number is needed.

In some cases, an attribute is attached to `metadata(object)$criticalValues`, see `critVal.power` for an example.

**kvals** the number of sites, k, to include in a window

**perm_cluster_best** a list whose canonical element is a vector of values like `x$target.min` obtained from a permutation of the class indicators

**summary_matrix** a matrix giving the start, end, depth, and counts in each class for every cluster and depth in sequential order

**call** the call invoking `gRxCluster` which may include some arguments added by default.

**Author(s)**

Charles Berry <ccberry@ucsd.edu>

---

**Description**

Plot Clumps and/or Critical Regions

**Usage**

```r
gRxPlot(object, pi.0 = NULL, method = c("odds", "criticalRegions"),
xlim = NULL, main = NULL, xlab = "log odds ratio", breaks = "Sturges",
kvals = NULL, ...)
```

**Arguments**

- `object` either the results of `gRxCluster` or a list containing cutpoints for critical regions.
- `pi.0` the background proportion for vector 2
- `method` character vector of “odds” and/or “criticalRegions”
- `xlim` limits of the log odds histogram
- `main` a title for the panel(s)
- `xlab` label fgor the x-axis of the log odds plot
- `breaks` see `hist`
- `kvals` values to use in selecting a subset of the critical regions to display
- `...` other args to pass to the plotting routine(s)
The results of a call to \texttt{gRxCluster} are plotted. Optionally, with method="criticalRegions" only the critical regions are plotted or with method="odds" the log odds only are plotted.

**Value**

see \texttt{hist}

**Author(s)**

Charles Berry

**See Also**

\texttt{gRxPlotClumps} for a more fine grained display

**Examples**

```r
x.seqnames <- rep(letters[1:3],each=500)
x.starts <- c(seq(1,length=500),seq(1,by=2,length=500),seq(1,by=3,length=500))
x.lens <- rep(c(5,10,15,20,25),each=20)
x.group <- rep(rep(c(TRUE,FALSE),length=length(x.lens)),x.lens)
## add a bit of fuzz:
x.group <- !==rbinom(length(x.group),1,pr=ifelse(x.group,.8,.2))
x.kvals <- as.integer(sort(unique(x.lens)))
x.res <- gRxCluster(x.seqnames,x.starts,x.group,x.kvals)
gRxPlot(x.res)
rm( x.seqnames, x.starts, x.lens, x.group, x.kvals, x.res)
```

```
\texttt{gRxPlotClumps} \hspace{1cm} \texttt{gRxPlotClumps}
```

**Description**

Plot gRxCluster object clumps

**Usage**

\texttt{gRxPlotClumps(object, data, seqlens, panelExpr = quote(grid()))}

**Arguments**

- **object**: result of gRxCluster
- **data**: (optional) GRanges like that from which args to gRxCluster were derived
- **seqlens**: (optional) seqlengths(data) or similar. Can be given if data is missing
- **panelExpr**: - an expression to evaluate after drawing each panel

**Details**

Plot Relative Frequencies of the two classes according to region. Regions typically alternate between clusters and non-clusters on each chromosome.
Author(s)

Charles Berry

Examples

```r
x.seqnames <- rep(letters[1:3], each=50)
x.starts <- c(seq(1, length=50), seq(1, by=2, length=50), seq(1, by=3, length=50))
x.lens <- rep(c(5, 10, 15, 20, 25), each=2)
x.group <- rep(rep(c(TRUE, FALSE), length=length(x.lens)), x.lens)
## add a bit of fuzz:
x.group <- 1==rbinom(length(x.group), 1, pr=ifelse(x.group,.8,.2))
x.kvals <- as.integer(sort(unique(x.lens)))
x.res <- gRxCluster(x.seqnames, x.starts, x.group, x.kvals)
gRxPlotClumps(x.res)
rm(x.seqnames, x.starts, x.lens, x.group, x.kvals, x.res)
```

Description

Summarize gRxCluster Results

Usage

```r
gRxSummary(object, targetFD = NULL)
```

Arguments

- `object` the result of `gRxCluster`
- `targetFD` the critical value target in each tail

Details

Get the FDR and related data for a run of gRxCluster. By selecting a value for `targetFD` that is smaller than what was used in constructing the object, fewer clumps will be included in the computation for the False Discovery Rate - akin to what would have been obtained from the object if it had been constructed using that value.

Value

- a list containing the summarized results

Author(s)

Charles Berry
Examples

x.seqnames <- rep(letters[1:3], each=50)
x.starts <- c(seq(1, length=50), seq(1, by=2, length=50), seq(1, by=3, length=50))
x.lens <- rep(c(5, 10, 15, 20, 25), each=2)
x.group <- rep(rep(c(TRUE, FALSE), length=length(x.lens)), x.lens)
x.kvals <- as.integer(sort(unique(x.lens)))
x.res <- gRxCluster(x.seqnames, x.starts, x.group, x.kvals, nperm=100L)
gRxSummary(x.res)
rm( x.seqnames, x.starts, x.lens, x.group, x.kvals, x.res)

Description

join contiguous windows

Usage

noprune(x, ...)

Arguments

  x          a GRanges object
  ...        currently unused

Details

return all the candidate sites in a clump without pruning
this is to be used as the pruneFun are of gRxCluster

Value

same as gRxCluster less the metadata

Author(s)

Charles Berry

See Also

gRxCluster-object for more details on what this function returns.
Description

Plot a set of cutpoints - Utility

Usage

```r
## S3 method for class 'cutpoints'
plot(crit, pi.0 = NULL, kvals = NULL, ...)
```

Arguments

- `crit` - a cutpoint object see `gRxCluster`
- `pi.0` - optional null value to plot
- `kvals` - which cutpoints to include in the plot
- `...` - passed to `barplot`

Details

NOT FOR USERS. Not exported.

Value

list with components of “bar.x” (the value of `hist()`), “kvals” (window widths plotted), and “pi.0” (the input value of `pi.0`)

Author(s)

Charles Berry

---

Description

best contiguous region

Usage

```r
prune.loglik(x, p.null = 0.5)
```

Arguments

- `x` - a GRanges object
- `p.null` - the probability of category 1 (FALSE)
Details
prune each end of the region using loglik criterion
this is to be used as the pruneFun are of gRxCluster

Author(s)
Charles Berry

See Also
`gRxCluster-object` for details on what this function returns.
Index

*Topic **cluster**
  gRxCluster-object, 6

critVal.alpha, 2
critVal.power, 3, 6, 7
critVal.target, 4, 7

geneRxCluster, 5
geneRxCluster-package (geneRxCluster), 5
GRanges, 6
gRxCluster, 2–5, 6, 7, 8, 10, 11
gRxCluster-object, 6
gRxPlot, 7
gRxPlotClumps, 8, 8
gRxSummary, 9

hist, 7, 8

noprunen, 10

plot.cutpoints, 11
prune.loglik, 11