Package 'Pi'

April 15, 2017

Type Package

Title Leveraging Genetic Evidence to Prioritise Drug Targets at the Gene, Pathway and Network Level

Version 1.2.1

Date 2017-2-13

Author Hai Fang, the ULTRA-DD Consortium, Julian C Knight

Maintainer Hai Fang <hfang@well.ox.ac.uk>

Depends XGR, igraph, dnet, ggplot2, graphics, stats

Imports Matrix, MASS, ggbio, GenomicRanges, GenomeInfoDb, supraHex, scales, grDevices, ggrepel, ROCR, randomForest

Suggests foreach, doParallel, BiocStyle, knitr, rmarkdown, png, GGally, gridExtra, fgsea, Gviz

Description Priority index or Pi is developed as a genomic-led target prioritisation system, with the focus on leveraging human genetic data to prioritise potential drug targets at the gene, pathway and network level. The long term goal is to use such information to enhance early-stage target validation. Based on evidence of disease association from genome-wide association studies (GWAS), this prioritisation system is able to generate evidence to support identification of the specific modulated genes (seed genes) that are responsible for the genetic association signal by utilising knowledge of linkage disequilibrium (co-inherited genetic variants), distance of associated variants from the gene, evidence of independent genetic association with gene expression in disease-relevant tissues, cell types and states, and evidence of physical interactions between disease-associated genetic variants and gene promoters based on genome-wide capture HiC-generated promoter interactomes in primary blood cell types. Seed genes are scored in an integrative way, quantifying the genetic influence. Scored seed genes are subsequently used as baits to rank seed genes plus additional (non-seed) genes; this is achieved by iteratively exploring the global connectivity of a gene interaction network. Genes with the highest priority are further used to identify/prioritise pathways that are significantly enriched with highly prioritised genes. Prioritised genes are also used to identify a gene network interconnecting highly prioritised genes and a minimal number of less prioritised genes (which act as linkers bringing together highly prioritised genes).

URL http://pi314.r-forge.r-project.org

BugReports https://github.com/hfang-bristol/Pi/issues

Collate 'xRWR.r' 'xPier.r' 'xPierGenes.r' 'xPierSNPs.r' 'xPierPathways.r' 'xPierManhattan.r' 'xPierSubnet.r' 'xPierMatrix.r' 'xSNPeqtl.r' 'xSNP2eGenes.r' 'xPierSNPsConsensus.r' 'xPredictROCR.r' 'xPredictCompare.r' 'xContour.r' 'xSNPhic.r' 'xPCHiCplot.r' 'xSNP2cGenes.r' 'xMLrandomforest.r' 'xPierSNPsAdv.r' 'xGSsimulator.r' 'xMLdotplot.r' 'xMLdensity.r' 'xPierTrack.r'

License GPL-3

VignetteBuilder knitr

biocViews Software, Genetics, GraphAndNetwork, Pathways, GeneExpression, GeneTarget, GenomeWideAssociation, LinkageDisequilibrium, Network, HiC

NeedsCompilation no

R topics documented:

`xContour` *Function to visualise a numeric matrix as a contour plot*

Description

`xContour` is supposed to visualise a numeric matrix as a contour plot.

Usage

```
xContour(data, main = "", xlab = "", ylab = "", key = "",
nlevels = 50, colormap = c("darkblue-lightblue-lightyellow-darkorange",
"bwr", "jet", "gbr", "wyr", "br", "yr", "rainbow", "wb"),
highlight = c("none", "min", "max"), highlight.col = "white",
x.label.cex = 0.95, x.label.srt = 30, signature = FALSE, ...)
```

Arguments

`data`	a numeric matrix for the contour plot
`main`	an overall title for the plot
`xlab`	a title for the x axis. If specified, it will override 'names(dimnames(data))[1]'
`ylab`	a title for the y axis. If specified, it will override 'names(dimnames(data))[2]'
`key`	a title for the key plot (on the right)
`nlevels`	the number of levels to partition the input matrix values. The same level has the same color mapped to
`colormap`	short name for the colormap. It can be one of "jet" (jet colormap), "bwr" (blue-white-red colormap), "gbr" (green-black-red colormap), "wyr" (white-yellow-red colormap), "br" (black-red colormap), "yr" (yellow-red colormap), "wb" (white-black colormap), and "rainbow" (rainbow colormap, that is, red-yellow-green-cyan-blue-magenta). Alternatively, any hyphen-separated HTML color names, e.g. "blue-black-yellow", "royalblue-white-sandybrown", "darkgreen-white-darkviolet". A list of standard color names can be found in `http://html-color-codes.info/color-names`
`highlight`	how to highlight the point. It can be 'none' for no highlight (by default), 'min' for highlighting the point with the minimum value of the matrix, and 'max' for highlighting the point with the maximum value of the matrix
`highlight.col`	the highlight colors
`x.label.cex`	the x-axis label size
`x.label.srt`	the x-axis label angle (in degree from horizontal)
`signature`	a logical to indicate whether the signature is assigned to the plot caption. By default, it sets FALSE
`...`	additional graphic parameters. For most parameters, please refer to `http://stat.ethz.ch/R-manual/R-devel/library/graphics/html/filled.contour.html`

Value

invisible

Note

none

See Also

`xContour`

Examples

```
x <- y <- seq(-4*pi, 4*pi, len=10)
r <- sqrt(outer(x^2, y^2, "+"))
data <- cos(r^2)*exp(-r/(2*pi))
xContour(data)
#xContour(data, signature=TRUE)
```

`xGSsimulator` *Function to simulate gold standard negatives (GSN) given gold standard positives (GSP) and a gene network*

Description

`xGSsimulator` is supposed to simulate gold standard negatives (GSN) given gold standard positives (GSP) and an input gene network. GSN targets are those after excluding GSP targets and their 1-order (by default) neighbors in the gene network.

Usage

```
xGSsimulator(GSP, population = NULL, network = c("STRING_medium",
"STRING_low", "STRING_high", "STRING_highest", "PCommonsUN_high",
"PCommonsUN_medium")[c(1, 6)], network.customised = NULL,
neighbor.order = 1, verbose = TRUE,
RData.location = "http://galahad.well.ox.ac.uk/bigdata")
```

Arguments

`GSP` a vector containing Gold Standard Positives (GSP)

`population` a vector containing population space in which gold standard negatives (GSN) will be considered. By default, it is NULL, meaning genes in the network will be used instead

`network` the built-in network. Currently two sources of network information are supported: the STRING database (version 10) and the Pathways Commons database (version 7). STRING is a meta-integration of undirect interactions from the functional aspect, while Pathways Commons mainly contains both undirect and direct interactions from the physical/pathway aspect. Both have scores to control the confidence of interactions. Therefore, the user can choose the different quality of the interactions. In STRING, "STRING_highest" indicates interactions with highest confidence (confidence scores>=900), "STRING_high" for interactions with high confidence (confidence scores>=700), "STRING_medium" for interactions with medium confidence (confidence scores>=400), and "STRING_low" for interactions with low confidence (confidence scores>=150). For undirect/physical interactions from Pathways Commons, "PCommonsUN_high" indicates undirect interactions with high confidence (supported with the PubMed references plus at least 2 different sources), "PCommonsUN_medium" for undirect interactions with medium confidence (supported with the PubMed references). By default, "STRING_medium" and "PCommonsUN_medium" are used

`network.customised`
an object of class "igraph". By default, it is NULL. It is designed to allow the user analysing their customised network data that are not listed in the above argument 'network'. This customisation (if provided) has the high priority over built-in network

`neighbor.order`	an integer giving the order of the neighborhood. By default, it is 1-order neighborhood
`verbose`	logical to indicate whether the messages will be displayed in the screen. By default, it sets to true for display
`RData.location`	the characters to tell the location of built-in RData files. See `xRDataLoader` for details

Value

a list with following components:

- `GSP`: a vector containing GSP after considering the population space
- `GSN`: a vector containing simulated GSN
- `g`: an "igraph" object
- `call`: the call that produced this result

Note

If multiple graphs are provided, they will be unionised for use.

See Also

`xRDataLoader`, `xPredictROCR`, `xMLrandomforest`

Examples

```
## Not run:
# Load the library
library(Pi)

## End(Not run)
RData.location <- "http://galahad.well.ox.ac.uk/bigdata_dev"
## Not run:
GS <- xGSsimulator(GSP, population,
network=c("STRING_medium","PCommonsUN_medium"),
RData.location=RData.location)

## End(Not run)
```

`xMLdensity` *Function to visualise machine learning results using density plot*

Description

`xMLdensity` is supposed to visualise machine learning results using density plot. It returns an object of class "ggplot".

Usage

```
xMLdensity(pTarget, displayBy = c("All", "GS", "GSN", "GSP",
"Putative"),
x.scale = c("sqrt", "normal"), signature = TRUE)
```

Arguments

`pTarget`	an object of class "pTarget"
`displayBy`	which targets will be used for displaying. It can be one of "GS" for gold standard targets, "GSN" for gold standard negatives, "GSP" for gold standard positives, "Putative" for putative targets (non-GS), "All" for all targets (by default)
`x.scale`	how to transform the x scale. It can be "normal" for no transformation, and "sqrt" for square root transformation (by default)
`signature`	logical to indicate whether the signature is assigned to the plot caption. By default, it sets TRUE showing which function is used to draw this graph

Value

an object of class "ggplot"

Note

none

See Also

`xMLrandomforest`

Examples

```
## Not run:
# Load the library
library(Pi)

## End(Not run)
RData.location <- "http://galahad.well.ox.ac.uk/bigdata_dev"
## Not run:
gp <- xMLdensity(pTarget, displayBy="All")
gp

## End(Not run)
```

`xMLdotplot` *Function to visualise machine learning results using dot plot*

Description

`xMLdotplot` is supposed to visualise machine learning results using dot plot. It returns an object of class "ggplot".

Usage

```
xMLdotplot(pTarget, displayBy = c("importance2fold", "roc2fold",
"fmax2fold",
"importance_accurancy", "importance_gini", "ROC", "Fmax"), signature =
TRUE)
```

Arguments

`pTarget`	an object of class "pTarget"
`displayBy`	which statistics will be used for displaying. It can be either statistics across folds ("importance2fold" for predictor importance, "roc2fold" for AUC in ROC, "fmax2fold" for F-max in Precision-Recall curve) or overall statistics ("importance_accurancy" for predictor importance measured by accuracy decrease, "importance_gini" for predictor importance measured by Gini decrease, "ROC" for AUC in ROC, "Fmax" for F-max in Precision-Recall curve)
`signature`	logical to indicate whether the signature is assigned to the plot caption. By default, it sets TRUE showing which function is used to draw this graph

Value

an object of class "ggplot"

Note

none

See Also

`xMLrandomforest`

Examples

```
## Not run:
# Load the library
library(Pi)

## End(Not run)
RData.location <- "http://galahad.well.ox.ac.uk/bigdata_dev"
## Not run:
gp <- xMLdotplot(pTarget, displayBy="importance_accurancy")
gp

## End(Not run)
```

`xMLrandomforest` *Function to integrate predictor matrix in a supervised manner via machine learning algorithm random forest.*

Description

`xMLrandomforest` is supposed to integrate predictor matrix in a supervised manner via machine learning algorithm random forest. It requires three inputs: 1) Gold Standard Positive (GSP) targets; 2) Gold Standard Negative (GSN) targets; 3) a predictor matrix containing genes in rows and predictors in columns, with their predictive scores inside it. It returns an object of class 'pTarget'.

Usage

```
xMLrandomforest(df_predictor, GSP, GSN, nfold = 3, mtry = NULL,
ntree = 2000, fold.aggregateBy = c("Ztransform", "logistic", "fishers",
"orderStatistic"), verbose = TRUE, ...)
```

Arguments

`df_predictor` a data frame containing genes (in rows) and predictors (in columns), with their predictive scores inside it. This data frame must has gene symbols as row names

`GSP` a vector containing Gold Standard Positive (GSP)

`GSN` a vector containing Gold Standard Negative (GSN)

`nfold` an integer specifying the number of folds for cross validataion

`mtry` an integer specifying the number of predictors randomly sampled as candidates at each split. If NULL, it will be tuned by ‘randomForest::tuneRF‘, with starting value as sqrt(p) where p is the number of predictors. The minimum value is 3

`ntree` an integer specifying the number of trees to grow. By default, it sets to 2000

`fold.aggregateBy`
the aggregate method used to aggregate results from k-fold cross validataion. It can be either "orderStatistic" for the method based on the order statistics of p-values, or "fishers" for Fisher’s method, "Ztransform" for Z-transform method, "logistic" for the logistic method. Without loss of generality, the Z-transform method does well in problems where evidence against the combined null is spread widely (equal footings) or when the total evidence is weak; Fisher’s method does best in problems where the evidence is concentrated in a relatively small fraction of the individual tests or when the evidence is at least moderately strong; the logistic method provides a compromise between these two. Notably, the aggregate methods ’Ztransform’ and ’logistic’ are preferred here

`verbose` logical to indicate whether the messages will be displayed in the screen. By default, it sets to TRUE for display

`...` additional graphic parameters. Please refer to ’randomForest::randomForest’ for the complete list.

Value

an object of class "pTarget", a list with following components:

- `model`: a list of models, results from per-fold train set
- `priority`: a data frame of nGene X 6 containing gene priority information, where nGene is the number of genes in the input data frame, and the 6 columns are "GS" (either ’GSP’, or ’GSN’, or ’Putative’), "name" (gene names), "rank" (ranks of the priority scores), "pvalue" (the cross-fold aggregated p-value of being GSP, per-fold p-value converted from empirical cumulative distribution of the probability of being GSP), "fdr" (fdr adjusted from the aggregated p-value), "priority" (-log10(fdr))
- `predictor`: a data frame, which is the same as the input data frame but inserting an additional column ’GS’ in the first column
- `pred2fold`: a list of data frame, results from per-fold test set
- `prob2fold`: a data frame of nGene X 2+nfold containing the probability of being GSP, where nGene is the number of genes in the input data frame, nfold is the number of folds for cross validataion, and the first two columns are "GS" (either ’GSP’, or ’GSN’, or ’Putative’), "name" (gene names), and the rest columns storing the per-fold probability of being GSP
- `importance2fold`: a data frame of nPredictor X 4+nfold containing the predictor importance info per fold, where nPredictor is the number of predictors, nfold is the number of folds for cross validataion, and the first 4 columns are "median" (the median of the importance across folds), "mad" (the median of absolute deviation of the importance across folds), "min" (the minimum of the importance across folds), "max" (the maximum of the importance across folds), and the rest columns storing the per-fold importance

- roc2fold: a data frame of 1+nPredictor X 4+nfold containing the supervised/predictor ROC info (AUC values), where nPredictor is the number of predictors, nfold is the number of folds for cross validataion, and the first 4 columns are "median" (the median of the AUC values across folds), "mad" (the median of absolute deviation of the AUC values across folds), "min" (the minimum of the AUC values across folds), "max" (the maximum of the AUC values across folds), and the rest columns storing the per-fold AUC values
- fmax2fold: a data frame of 1+nPredictor X 4+nfold containing the supervised/predictor PR info (F-max values), where nPredictor is the number of predictors, nfold is the number of folds for cross validataion, and the first 4 columns are "median" (the median of the F-max values across folds), "mad" (the median of absolute deviation of the F-max values across folds), "min" (the minimum of the F-max values across folds), "max" (the maximum of the F-max values across folds), and the rest columns storing the per-fold F-max values
- importance: a data frame of nPredictor X 2 containing the predictor importance info, where nPredictor is the number of predictors, two columns for two types ("MeanDecreaseAccuracy" and "MeanDecreaseGini") of predictor importance measures. "MeanDecreaseAccuracy" sees how worse the model performs without each predictor (a high decrease in accuracy would be expected for very informative predictors), while "MeanDecreaseGini" measures how pure the nodes are at the end of the tree (a high score means the predictor was important if each predictor is taken out)
- performance: a data frame of 1+nPredictor X 2 containing the supervised/predictor performance info predictor importance info, where nPredictor is the number of predictors, two columns are "ROC" (AUC values) and "Fmax" (F-max values)
- call: the call that produced this result

Note

none

Examples

```
## Not run:
# Load the library
library(Pi)

## End(Not run)
RData.location <- "http://galahad.well.ox.ac.uk/bigdata_dev"
## Not run:
pTarget <- xMLrandomforest(df_prediction, GSP, GSN)

## End(Not run)
```

---

xPCHiCplot *Function to visualise promoter capture HiC data using different network layouts*

Description

xPCHiCplot is supposed to visualise promoter capture HiC data using different network layouts.

Usage

```
xPCHiCplot(g, node.info = c("smart", "none", "GR", "GR_SNP",
"GR_SNP_target",
"SNP_target"), node.colors = c("skyblue", "pink1"), nodes.query = NULL,
newpage = TRUE, signature = TRUE, glayout = layout_with_kk,
vertex.frame.color = NA, vertex.size = NULL, vertex.color = NULL,
vertex.shape = "sphere", vertex.label = NULL, vertex.label.cex = NULL,
vertex.label.font = 2, vertex.label.dist = 0.3,
vertex.label.color = "black", edge.arrow.size = 0.5, edge.width = NULL,
edge.color = "grey", ...)
```

Arguments

`g` an object of both classes "igraph" and "PCHiC" (part of the results from `xSNPhic`)

`node.info` tells the information used to label nodes. It can be one of "none" for no node labeling, "GR" for only using genomic regions (GR), "GR_SNP" for using GR and SNP (if any), "GR_SNP_target" for using GR and SNP (if any) and target genes (if any), "SNP_target" for using SNP (if any) and target genes (if any), and "smart" (by default) for only using GR if both SNP and target genes are not available (otherwise GR will be hidden)

`node.colors` colors used to flag which nodes contain SNP or not. By default, a node harboring an SNP will be colored in 'skyblue' and the node without an SNP in 'pink'

`nodes.query` nodes in query for which edges attached to them will be displayed. By default, it sets to NULL meaning no such restriction

`newpage` logical to indicate whether to open a new page. By default, it sets to true for opening a new page

`signature` logical to indicate whether the signature is assigned to the plot caption. By default, it sets TRUE

`glayout` either a function or a numeric matrix configuring how the vertices will be placed on the plot. If layout is a function, this function will be called with the graph as the single parameter to determine the actual coordinates. This function can be one of "layout_nicely" (previously "layout.auto"), "layout_randomly" (previously "layout.random"), "layout_in_circle" (previously "layout.circle"), "layout_on_sphere" (previously "layout.sphere"), "layout_with_fr" (previously "layout.fruchterman.reingold"), "layout_with_kk" (previously "layout.kamada.kawai"), "layout_as_tree" (previously "layout.reingold.tilford"), "layout_with_lgl" (previously "layout.lgl"), "layout_with_graphopt" (previously "layout.graphopt"), "layout_with_sugiyama" (previously "layout.kamada.kawai"), "layout_with_dh" (previously "layout.davidson.harel"), "layout_with_drl" (previously "layout.drl"), "layout_with_gem" (previously "layout.gem"), "layout_with_mds". A full explanation of these layouts can be found in `http://igraph.org/r/doc/layout_nicely.html`

`vertex.frame.color`
the color of the frame of the vertices. If it is NA, then there is no frame

`vertex.size` the size of each vertex. If it is a vector, each vertex may differ in size

`vertex.color` the fill color of the vertices. If it is NA, then there is no fill color

`vertex.shape` the shape of each vertex. It can be one of "circle", "square", "csquare", "rectangle", "crectangle", "vrectangle", "pie" (`http://igraph.org/r/doc/vertex.shape.pie.html`), "sphere", and "none" `vertex.label` the label of the vertices. If it is NA, then there is no label. The default vertex labels are the name attribute of the nodes

`vertex.label.cex` the font size of vertex labels.

`vertex.label.font` the font of vertex labels. It is interpreted the same way as the the 'font' graphical parameter: 1 is plain text, 2 is bold face, 3 is italic, 4 is bold and italic and 5 specifies the symbol font.

`vertex.label.dist` the distance of the label from the center of the vertex. If it is 0 then the label is centered on the vertex. If it is 1 then the label is displayed beside the vertex.

`vertex.label.color` the color of vertex labels.

`edge.arrow.size` the size of the arrows for the directed edge. The default value is 0.5.

`edge.width` the width of the directed edge. If NULL, the width edge is proportional to CHiCAGO scores (quantifying the strength of physical interactions).

`edge.color` the color of the directed edge. The default value is 'grey'.

`...` additional graphic parameters. See http://igraph.org/r/doc/plot.common.html for the complete list.

Value

an igraph object

Note

none

- `edge arrow`: interactions are represented as a direct graph (bait-prey)
- `edge thickness`: the thickness in an edge is proportional to the interaction strength
- `node color`: a node is colored in pink if it harbors SNPs in query; otherwise skyblue
- `node label`: a node is labelled with three pieces of information (if any): genomic regions, SNPs in query, genes associated (marked by an @ icon)

See Also

xSNPhic

Examples

```
## Not run:
# Load the library
library(Pi)

## End(Not run)

RData.location <- "http://galahad.well.ox.ac.uk/bigdata_dev"
# a) provide the SNPs with the significance info
## get lead SNPs reported in AS GWAS and their significance info (p-values)
#data.file <- "http://galahad.well.ox.ac.uk/bigdata/AS.txt"
#AS <- read.delim(data.file, header=TRUE, stringsAsFactors=FALSE)
```

```
ImmunoBase <- xRDataLoader(RData.customised='ImmunoBase',
RData.location=RData.location)
data <- names(ImmunoBase$AS$variants)

## Not run:
# b) extract HiC-gene pairs given a list of AS SNPs
PCHiC <- xSNPhic(data, include.HiC="Monocytes", GR.SNP="dbSNP_GWAS",
RData.location=RData.location)
head(PCHiC$df)

# c) visualise the interaction (a directed graph: bait->prey)
g <- PCHiC$ig
## a node with SNPs colored in 'skyblue' and the one without SNPs in 'pink'
## the width in an edge is proportional to the interaction strength
xPCHiCplot(g, vertex.shape="sphere")
xPCHiCplot(g, glayout=layout_in_circle, vertex.shape="sphere")

# d) control node labelling info
xPCHiCplot(g, node.info="GR_SNP_target")
xPCHiCplot(g, node.info="GR_SNP")
xPCHiCplot(g, node.info="SNP_target")
xPCHiCplot(g, node.info='SNP_target', vertex.label.cex=0.5)

## End(Not run)
```

xPier — *Function to do prioritisation through random walk techniques*

Description

xPier is supposed to prioritise nodes given an input graph and a list of seed nodes. It implements Random Walk with Restart (RWR) and calculates the affinity score of all nodes in the graph to the seeds. The priority score is the affinity score. Parallel computing is also supported for Linux-like or Windows operating systems. It returns an object of class "pNode".

Usage

```
xPier(seeds, g, seeds.inclusive = TRUE, normalise = c("laplacian",
"row",
"column", "none"), restart = 0.75, normalise.affinity.matrix =
c("none",
"quantile"), parallel = TRUE, multicores = NULL, verbose = TRUE)
```

Arguments

seeds	a named input vector containing a list of seed nodes. For this named vector, the element names are seed/node names (e.g. gene symbols), the element (non-zero) values used to weight the relative importance of seeds. Alternatively, it can be a matrix or data frame with two columns: 1st column for seed/node names, 2nd column for the weight values
g	an object of class "igraph" to represent network. It can be a weighted graph with the node attribute 'weight'

`seeds.inclusive`
: logical to indicate whether non-network seed genes are included for prioritisation. If TRUE (by default), these genes will be added to the netowrk

`normalise`
: the way to normalise the adjacency matrix of the input graph. It can be 'laplacian' for laplacian normalisation, 'row' for row-wise normalisation, 'column' for column-wise normalisation, or 'none'

`restart`
: the restart probability used for Random Walk with Restart (RWR). The restart probability takes the value from 0 to 1, controlling the range from the starting nodes/seeds that the walker will explore. The higher the value, the more likely the walker is to visit the nodes centered on the starting nodes. At the extreme when the restart probability is zero, the walker moves freely to the neighbors at each step without restarting from seeds, i.e., following a random walk (RW)

`normalise.affinity.matrix`
: the way to normalise the output affinity matrix. It can be 'none' for no normalisation, 'quantile' for quantile normalisation to ensure that columns (if multiple) of the output affinity matrix have the same quantiles

`parallel`
: logical to indicate whether parallel computation with multicores is used. By default, it sets to true, but not necessarily does so. Partly because parallel backends available will be system-specific (now only Linux or Mac OS). Also, it will depend on whether these two packages "foreach" and "doMC" have been installed. It can be installed via: `source("http://bioconductor.org/biocLite.R"); biocLite(c("foreach","doMC"))`. If not yet installed, this option will be disabled

`multicores`
: an integer to specify how many cores will be registered as the multicore parallel backend to the 'foreach' package. If NULL, it will use a half of cores available in a user's computer. This option only works when parallel computation is enabled

`verbose`
: logical to indicate whether the messages will be displayed in the screen. By default, it sets to true for display

Value

an object of class "pNode", a list with following components:

- `priority`: a matrix of nNode X 5 containing node priority information, where nNode is the number of nodes in the input graph, and the 5 columns are "name" (node names), "node" (1 for network genes, 0 for non-network seed genes), "seed" (1 for seeds, 0 for non-seeds), "weight" (weight values), "priority" (the priority scores that are rescaled to the range [0,1]), "rank" (ranks of the priority scores)
- `g`: an input "igraph" object
- `call`: the call that produced this result

Note

The input graph will treat as an unweighted graph if there is no 'weight' edge attribute associated with

See Also

`xRDataLoader`, `xRWR`, `xPierSNPs`, `xPierGenes`, `xPierPathways`

Examples

```
## Not run:
# Load the library
library(Pi)

## End(Not run)

RData.location <- "http://galahad.well.ox.ac.uk/bigdata_dev"
# a) provide the input nodes/genes with the significance info
sig <- rbeta(500, shape1=0.5, shape2=1)
## Not run:
## load human genes
org.Hs.eg <- xRDataLoader(RData='org.Hs.eg',
RData.location=RData.location)
data <- data.frame(symbols=org.Hs.eg$gene_info$Symbol[1:500], sig)

# b) provide the network
g <- xRDataLoader(RData.customised='org.Hs.PCommons_UN',
RData.location=RData.location)

# c) perform priority analysis
pNode <- xPier(seeds=data, g=g, restart=0.75)

## End(Not run)
```

---

xPierGenes — *Function to prioritise genes from an input network and the weight info imposed on its nodes*

Description

xPierGenes is supposed to prioritise genes given an input graph and a list of seed nodes. It implements Random Walk with Restart (RWR) and calculates the affinity score of all nodes in the graph to the seeds. The priority score is the affinity score. Parallel computing is also supported for Linux-like or Windows operating systems. It returns an object of class "pNode".

Usage

```
xPierGenes(data, network = c("STRING_highest", "STRING_high",
"STRING_medium",
"STRING_low", "PCommonsUN_high", "PCommonsUN_medium",
"PCommonsDN_high",
"PCommonsDN_medium", "PCommonsDN_Reactome", "PCommonsDN_KEGG",
"PCommonsDN_HumanCyc", "PCommonsDN_PID", "PCommonsDN_PANTHER",
"PCommonsDN_ReconX", "PCommonsDN_TRANSFAC", "PCommonsDN_PhosphoSite",
"PCommonsDN_CTD"), weighted = FALSE, network.customised = NULL,
seeds.inclusive = TRUE, normalise = c("laplacian", "row", "column",
"none"), restart = 0.75, normalise.affinity.matrix = c("none",
"quantile"), parallel = TRUE, multicores = NULL, verbose = TRUE,
RData.location = "http://galahad.well.ox.ac.uk/bigdata")
```

Arguments

`data` a named input vector containing a list of seed nodes (ie gene symbols). For this named vector, the element names are seed/node names (e.g. gene symbols), the element (non-zero) values used to weight the relative importance of seeds. Alternatively, it can be a matrix or data frame with two columns: 1st column for seed/node names, 2nd column for the weight values

`network` the built-in network. Currently two sources of network information are supported: the STRING database (version 10) and the Pathways Commons database (version 7). STRING is a meta-integration of undirect interactions from the functional aspect, while Pathways Commons mainly contains both undirect and direct interactions from the physical/pathway aspect. Both have scores to control the confidence of interactions. Therefore, the user can choose the different quality of the interactions. In STRING, "STRING_highest" indicates interactions with highest confidence (confidence scores>=900), "STRING_high" for interactions with high confidence (confidence scores>=700), "STRING_medium" for interactions with medium confidence (confidence scores>=400), and "STRING_low" for interactions with low confidence (confidence scores>=150). For undirect/physical interactions from Pathways Commons, "PCommonsUN_high" indicates undirect interactions with high confidence (supported with the PubMed references plus at least 2 different sources), "PCommonsUN_medium" for undirect interactions with medium confidence (supported with the PubMed references). For direct (pathway-merged) interactions from Pathways Commons, "PCommonsDN_high" indicates direct interactions with high confidence (supported with the PubMed references plus at least 2 different sources), and "PCommonsUN_medium" for direct interactions with medium confidence (supported with the PubMed references). In addtion to pooled version of pathways from all data sources, the user can also choose the pathway-merged network from individual sources, that is, "PCommonsDN_Reactome" for those from Reactome, "PCommonsDN_KEGG" for those from KEGG, "PCommonsDN_HumanCyc" for those from HumanCyc, "PCommonsDN_PID" for those froom PID, "PCommonsDN_PANTHER" for those from PANTHER, "PCommonsDN_ReconX" for those from ReconX, "PCommonsDN_TRANSFAC" for those from TRANSFAC, "PCommonsDN_PhosphoSite" for those from PhosphoSite, and "PCommonsDN_CTD" for those from CTD

`weighted` logical to indicate whether edge weights should be considered. By default, it sets to false. If true, it only works for the network from the STRING database

`network.customised`
an object of class "igraph". By default, it is NULL. It is designed to allow the user analysing their customised network data that are not listed in the above argument 'network'. This customisation (if provided) has the high priority over built-in network. If the user provides the "igraph" object with the "weight" edge attribute, RWR will assume to walk on the weighted network

`seeds.inclusive`
logical to indicate whether non-network seed genes are included for prioritisation. If TRUE (by default), these genes will be added to the netowrk

`normalise` the way to normalise the adjacency matrix of the input graph. It can be 'laplacian' for laplacian normalisation, 'row' for row-wise normalisation, 'column' for column-wise normalisation, or 'none'

`restart` the restart probability used for Random Walk with Restart (RWR). The restart probability takes the value from 0 to 1, controlling the range from the starting nodes/seeds that the walker will explore. The higher the value, the more likely the walker is to visit the nodes centered on the starting nodes. At the extreme when the restart probability is zero, the walker moves freely to the neighbors at each step without restarting from seeds, i.e., following a random walk (RW)

`normalise.affinity.matrix`
: the way to normalise the output affinity matrix. It can be 'none' for no normalisation, 'quantile' for quantile normalisation to ensure that columns (if multiple) of the output affinity matrix have the same quantiles

`parallel`
: logical to indicate whether parallel computation with multicores is used. By default, it sets to true, but not necessarily does so. Partly because parallel backends available will be system-specific (now only Linux or Mac OS). Also, it will depend on whether these two packages "foreach" and "doMC" have been installed. It can be installed via: `source("http://bioconductor.org/biocLite.R"); biocLite(c("foreach","doMC"))`. If not yet installed, this option will be disabled

`multicores`
: an integer to specify how many cores will be registered as the multicore parallel backend to the 'foreach' package. If NULL, it will use a half of cores available in a user's computer. This option only works when parallel computation is enabled

`verbose`
: logical to indicate whether the messages will be displayed in the screen. By default, it sets to true for display

`RData.location`
: the characters to tell the location of built-in RData files. See `xRDataLoader` for details

Value

an object of class "pNode", a list with following components:

- `priority`: a matrix of nNode X 6 containing node priority information, where nNode is the number of nodes in the input graph, and the 5 columns are "name" (node names), "node" (1 for network genes, 0 for non-network seed genes), "seed" (1 for seeds, 0 for non-seeds), "weight" (weight values), "priority" (the priority scores that are rescaled to the range [0,1]), "rank" (ranks of the priority scores), "description" (node description)
- `g`: an input "igraph" object
- `call`: the call that produced this result

Note

The input graph will treat as an unweighted graph if there is no 'weight' edge attribute associated with

See Also

`xRDataLoader`, `xPierSNPs`, `xPier`, `xPierPathways`

Examples

```
## Not run:
# Load the library
library(Pi)

## End(Not run)
```

```
RData.location <- "http://galahad.well.ox.ac.uk/bigdata_dev"
# a) provide the seed nodes/genes with the weight info
## load ImmunoBase
ImmunoBase <- xRDataLoader(RData.customised='ImmunoBase',
RData.location=RData.location)
## get genes within 500kb away from AS GWAS lead SNPs
seeds.genes <- ImmunoBase$AS$genes_variants
## seeds weighted according to distance away from lead SNPs
data <- 1- seeds.genes/500000

## Not run:
# b) perform priority analysis
pNode <- xPierGenes(data=data, network="PCommonsDN_medium",restart=0.7,
RData.location=RData.location)

# c) save to the file called 'Genes_priority.txt'
write.table(pNode$priority, file="Genes_priority.txt", sep="\t",
row.names=FALSE)

## End(Not run)
```

xPierManhattan *Function to visualise prioritised genes using manhattan plot*

Description

xPierManhattan is supposed to visualise prioritised genes using manhattan plot. Genes with the top priority are highlighed. It returns an object of class "ggplot".

Usage

```
xPierManhattan(pNode, color = c("darkred", "darkgreen"), top = 50,
top.label.type = c("box", "text"), top.label.size = 2,
top.label.col = "darkblue", top.label.query = NULL,
label.query.only = FALSE, y.scale = c("normal", "sqrt"),
GR.Gene = c("UCSC_knownGene", "UCSC_knownCanonical"), signature = TRUE,
verbose = TRUE, RData.location =
"http://galahad.well.ox.ac.uk/bigdata")
```

Arguments

pNode	an object of class "pNode" (or "pTarget" or "dTarget")
color	a character vector for colors to alternate chromosome colorings. If NULL, ggplot2 default colors will be used. If a single character is provided, it can be "jet" (jet colormap) or "rainbow" (rainbow colormap, that is, red-yellow-green-cyan-blue-magenta)
top	the number of the top targets to be labelled/highlighted
top.label.type	how to label the top targets. It can be "box" drawing a box around the labels , and "text" for the text only
top.label.size	the highlight label size
top.label.col	the highlight label color

`top.label.query`
: which top genes in query will be labelled. By default, it sets to NULL meaning all top genes will be displayed. If labels in query can not be found, then all will be displayed

`label.query.only`
: logical to indicate whether only genes in query will be displayed. By default, it sets to FALSE. It only works when labels in query are enabled/found

`y.scale`
: how to transform the y scale. It can be "normal" for no transformation, and "sqrt" for square root transformation

`GR.Gene`
: the genomic regions of genes. By default, it is 'UCSC_knownGene', that is, UCSC known genes (together with genomic locations) based on human genome assembly hg19. It can be 'UCSC_knownCanonical', that is, UCSC known canonical genes (together with genomic locations) based on human genome assembly hg19. Alternatively, the user can specify the customised input. To do so, first save your RData file (containing an GR object) into your local computer, and make sure the GR object content names refer to Gene Symbols. Then, tell "GR.Gene" with your RData file name (with or without extension), plus specify your file RData path in "RData.location"

`signature`
: logical to indicate whether the signature is assigned to the plot caption. By default, it sets TRUE

`verbose`
: logical to indicate whether the messages will be displayed in the screen. By default, it sets to false for no display

`RData.location`
: the characters to tell the location of built-in RData files. See `xRDataLoader` for details

Value

an object of class "ggplot", appended by an GR object called 'gr'

Note

none

See Also

`xRDataLoader`, `xPier`, `xPierSNPs`, `xPierGenes`, `xPierPathways`

Examples

```
## Not run:
# Load the library
library(Pi)

## End(Not run)

RData.location <- "http://galahad.well.ox.ac.uk/bigdata_dev"
# a) provide the SNPs with the significance info
## get lead SNPs reported in AS GWAS and their significance info (p-values)
#data.file <- "http://galahad.well.ox.ac.uk/bigdata/AS.txt"
#AS <- read.delim(data.file, header=TRUE, stringsAsFactors=FALSE)
ImmunoBase <- xRDataLoader(RData.customised='ImmunoBase',
RData.location=RData.location)
gr <- ImmunoBase$AS$variants
```

```
AS <- as.data.frame(GenomicRanges::mcols(gr)[, c('Variant','Pvalue')])

## Not run:
# b) perform priority analysis
pNode <- xPierSNPs(data=AS, include.eQTL="JKng_mono",
include.HiC='Monocytes', network="PCommonsUN_medium", restart=0.7,
RData.location=RData.location)

# c) manhattan plot
## default plot
mp <- xPierManhattan(pNode, RData.location=RData.location)
#pdf(file="Gene_manhattan.pdf", height=6, width=12, compress=TRUE)
print(mp)
#dev.off()
mp$gr
## control visuals
mp <- xPierManhattan(pNode, color='ggplot2', top=50,
top.label.col="black", y.scale="sqrt", RData.location=RData.location)
mp
## control labels
# only IL genes will be labelled
ind <- grep('^IL', rownames(pNode$priority))
top.label.query <- rownames(pNode$priority)[ind]
mp <- xPierManhattan(pNode, top.label.query=top.label.query,
RData.location=RData.location)
mp
# only IL genes will be displayed
mp <- xPierManhattan(pNode, top.label.query=top.label.query,
label.query.only=TRUE, RData.location=RData.location)
mp

## End(Not run)
```

xPierMatrix *Function to extract priority matrix from a list of pNode objects*

Description

xPierMatrix is supposed to extract priority matrix from a list of pNode objects. Also supported is the aggregation of priority matrix (similar to the meta-analysis) generating the priority results; we view this functionality as the discovery mode of the prioritisation.

Usage

```
xPierMatrix(list_pNode, displayBy = c("score", "rank", "pvalue"),
combineBy = c("intersect", "union"), aggregateBy = c("none", "fishers",
"logistic", "Ztransform", "orderStatistic"), verbose = TRUE)
```

Arguments

list_pNode a list of "pNode" objects

displayBy which priority will be extracted. It can be "score" for priority score (by default), "rank" for priority rank, "pvalue" for priority p-value `combineBy` how to resolve nodes/targets from a list of "pNode" objects. It can be "intersect" for intersecting nodes (by default), "union" for unionising nodes

`aggregateBy` the aggregate method used. It can be either "none" for no aggregation, or "orderStatistic" for the method based on the order statistics of p-values, "fishers" for Fisher's method, "Ztransform" for Z-transform method, "logistic" for the logistic method. Without loss of generality, the Z-transform method does well in problems where evidence against the combined null is spread widely (equal footings) or when the total evidence is weak; Fisher's method does best in problems where the evidence is concentrated in a relatively small fraction of the individual tests or when the evidence is at least moderately strong; the logistic method provides a compromise between these two. Notably, the aggregate methods 'fishers' and 'logistic' are preferred here

`verbose` logical to indicate whether the messages will be displayed in the screen. By default, it sets to true for display

Value

If aggregateBy is 'none' (by default), a data frame containing priority matrix, with each column/predictor for either priority score, or priorty rank or priority p-value. If aggregateBy is not 'none', an object of the class "dTarget", a list with following components:

- `priority`: a data frame of nGene X 5 containing gene priority (aggregated) information, where nGene is the number of genes, and the 5 columns are "name" (gene names), "rank" (ranks of the priority scores), "pvalue" (the aggregated p-value, converted from empirical cumulative distribution of the probability of being GSP), "fdr" (fdr adjusted from the aggregated p-value), "priority" (-log10(pvalue) but rescaled into the 0-10 range)
- `predictor`: a data frame containing priority matrix, with each column/predictor for either priority score, or priorty rank or priority p-value
- `call`: the call that produced this result

Note

none

See Also

`xPierSNPs`

Examples

```
## Not run:
# Load the library
library(Pi)

## End(Not run)
RData.location <- "http://galahad.well.ox.ac.uk/bigdata_dev"
## Not run:
df_score <- xPierMatrix(ls_pNode)

## End(Not run)
```

xPierPathways — *Function to prioritise pathways based on enrichment analysis of top prioritised genes*

Description

xPierPathways is supposed to prioritise pathways given prioritised genes and the ontology in query. It returns an object of class "eTerm". It is done via enrichment analysis.

Usage

```
xPierPathways(pNode, priority.top = 100, background = NULL,
ontology = c("GOBP", "GOMF", "GOCC", "PS", "PS2", "SF", "Pfam", "DO",
"HPPA", "HPMI", "HPCM", "HPMA", "MP", "MsigdbH", "MsigdbC1",
"MsigdbC2CGP",
"MsigdbC2CPall", "MsigdbC2CP", "MsigdbC2KEGG", "MsigdbC2REACTOME",
"MsigdbC2BIOCARTA", "MsigdbC3TFT", "MsigdbC3MIR", "MsigdbC4CGN",
"MsigdbC4CM",
"MsigdbC5BP", "MsigdbC5MF", "MsigdbC5CC", "MsigdbC6", "MsigdbC7",
"DGIdb",
"GTExV4", "GTExV6", "CreedsDisease", "CreedsDiseaseUP",
"CreedsDiseaseDN",
"CreedsDrug", "CreedsDrugUP", "CreedsDrugDN", "CreedsGene",
"CreedsGeneUP",
"CreedsGeneDN"), size.range = c(10, 2000), min.overlap = 3,
which.distance = NULL, test = c("hypergeo", "fisher", "binomial"),
p.adjust.method = c("BH", "BY", "bonferroni", "holm", "hochberg",
"hommel"),
ontology.algorithm = c("none", "pc", "elim", "lea"), elim.pvalue =
0.01,
lea.depth = 2, path.mode = c("all_paths", "shortest_paths",
"all_shortest_paths"), true.path.rule = FALSE, verbose = TRUE,
RData.location = "http://galahad.well.ox.ac.uk/bigdata")
```

Arguments

pNode	an object of class "pNode" (or "pTarget" or "dTarget")
priority.top	the number of the top targets used for enrichment analysis. By default, it sets to 100
background	a background vector. It contains a list of Gene Symbols as the test background. If NULL, by default all annotatable are used as background
ontology	the ontology supported currently. It can be "GOBP" for Gene Ontology Biological Process, "GOMF" for Gene Ontology Molecular Function, "GOCC" for Gene Ontology Cellular Component, "PS" for phylostratific age information, "PS2" for the collapsed PS version (inferred ancestors being collapsed into one with the known taxonomy information), "SF" for SCOP domain superfamilies, "Pfam" for Pfam domain families, "DO" for Disease Ontology, "HPPA" for Human Phenotype Phenotypic Abnormality, "HPMI" for Human Phenotype Mode of Inheritance, "HPCM" for Human Phenotype Clinical Modifier, "HPMA" for

Human Phenotype Mortality Aging, "MP" for Mammalian Phenotype, Drug-Gene Interaction database ("DGIdb") for drugable categories, tissue-specific eQTL-containing genes from GTEx ("GTExV4" and "GTExV6"), crowd extracted expression of differential signatures from CREEDS ("CreedsDisease", "CreedsDiseaseUP", "CreedsDiseaseDN", "CreedsDrug", "CreedsDrugUP", "CreedsDrugDN", "CreedsGene", "CreedsGeneUP" and "CreedsGeneDN"), and the molecular signatures database (Msigdb, including "MsigdbH", "MsigdbC1", "MsigdbC2CGP", "MsigdbC2CPall", "MsigdbC2CP", "MsigdbC2KEGG", "MsigdbC2REACTOME", "MsigdbC2BIOCARTA", "MsigdbC3TFT", "MsigdbC3MIR", "MsigdbC4CGN", "MsigdbC4CM", "MsigdbC5BP", "MsigdbC5MF", "MsigdbC5CC", "MsigdbC6", "MsigdbC7")

`size.range` the minimum and maximum size of members of each term in consideration. By default, it sets to a minimum of 10 but no more than 2000

`min.overlap` the minimum number of overlaps. Only those terms with members that overlap with input data at least min.overlap (3 by default) will be processed

`which.distance` which terms with the distance away from the ontology root (if any) is used to restrict terms in consideration. By default, it sets to 'NULL' to consider all distances

`test` the statistic test used. It can be "fisher" for using fisher's exact test, "hypergeo" for using hypergeometric test, or "binomial" for using binomial test. Fisher's exact test is to test the independence between gene group (genes belonging to a group or not) and gene annotation (genes annotated by a term or not), and thus compare sampling to the left part of background (after sampling without replacement). Hypergeometric test is to sample at random (without replacement) from the background containing annotated and non-annotated genes, and thus compare sampling to background. Unlike hypergeometric test, binomial test is to sample at random (with replacement) from the background with the constant probability. In terms of the ease of finding the significance, they are in order: hypergeometric test > binomial test > fisher's exact test. In other words, in terms of the calculated p-value, hypergeometric test < binomial test < fisher's exact test

`p.adjust.method`

the method used to adjust p-values. It can be one of "BH", "BY", "bonferroni", "holm", "hochberg" and "hommel". The first two methods "BH" (widely used) and "BY" control the false discovery rate (FDR: the expected proportion of false discoveries amongst the rejected hypotheses); the last four methods "bonferroni", "holm", "hochberg" and "hommel" are designed to give strong control of the family-wise error rate (FWER). Notes: FDR is a less stringent condition than FWER

`ontology.algorithm`

the algorithm used to account for the hierarchy of the ontology. It can be one of "none", "pc", "elim" and "lea". For details, please see 'Note' below

`elim.pvalue` the parameter only used when "ontology.algorithm" is "elim". It is used to control how to declare a signficantly enriched term (and subsequently all genes in this term are eliminated from all its ancestors)

`lea.depth` the parameter only used when "ontology.algorithm" is "lea". It is used to control how many maximum depth is used to consider the children of a term (and subsequently all genes in these children term are eliminated from the use for the recalculation of the signifance at this term)

`path.mode` the mode of paths induced by vertices/nodes with input annotation data. It can be "all_paths" for all possible paths to the root, "shortest_paths" for only one path to the root (for each node in query), "all_shortest_paths" for all shortest paths to the root (i.e. for each node, find all shortest paths with the equal lengths)

`true.path.rule` logical to indicate whether the true-path rule should be applied to propagate annotations. By default, it sets to false

`verbose` logical to indicate whether the messages will be displayed in the screen. By default, it sets to false for no display

`RData.location` the characters to tell the location of built-in RData files. See `xRDataLoader` for details

Value

an object of class "eTerm", a list with following components:

- `term_info`: a matrix of nTerm X 4 containing snp/gene set information, where nTerm is the number of terms, and the 4 columns are "id" (i.e. "Term ID"), "name" (i.e. "Term Name"), "namespace" and "distance"
- `annotation`: a list of terms containing annotations, each term storing its annotations. Always, terms are identified by "id"
- `data`: a vector containing input data in consideration. It is not always the same as the input data as only those mappable are retained
- `background`: a vector containing the background data. It is not always the same as the input data as only those mappable are retained
- `overlap`: a list of overlapped snp/gene sets, each storing snps overlapped between a snp/gene set and the given input data (i.e. the snps of interest). Always, gene sets are identified by "id"
- `zscore`: a vector containing z-scores
- `pvalue`: a vector containing p-values
- `adjp`: a vector containing adjusted p-values. It is the p value but after being adjusted for multiple comparisons
- `call`: the call that produced this result

Note

The interpretation of the algorithms used to account for the hierarchy of the ontology is:

- "none": does not consider the ontology hierarchy at all.
- "lea": computers the significance of a term in terms of the significance of its children at the maximum depth (e.g. 2). Precisely, once snps are already annotated to any children terms with a more signficance than itself, then all these snps are eliminated from the use for the recalculation of the signifance at that term. The final p-values takes the maximum of the original p-value and the recalculated p-value.
- "elim": computers the significance of a term in terms of the significance of its all children. Precisely, once snps are already annotated to a signficantly enriched term under the cutoff of e.g. pvalue<1e-2, all these snps are eliminated from the ancestors of that term).
- "pc": requires the significance of a term not only using the whole snps as background but also using snps annotated to all its direct parents/ancestors as background. The final p-value takes the maximum of both p-values in these two calculations.
- "Notes": the order of the number of significant terms is: "none" > "lea" > "elim" > "pc".

See Also

xRDataLoader, xEnricher

Examples

```
## Not run:
# Load the library
library(Pi)

## End(Not run)

RData.location <- "http://galahad.well.ox.ac.uk/bigdata_dev"
# a) provide the SNPs with the significance info
## get lead SNPs reported in AS GWAS and their significance info (p-values)
#data.file <- "http://galahad.well.ox.ac.uk/bigdata/AS.txt"
#AS <- read.delim(data.file, header=TRUE, stringsAsFactors=FALSE)
ImmunoBase <- xRDataLoader(RData.customised='ImmunoBase')
gr <- ImmunoBase$AS$variants
AS <- as.data.frame(GenomicRanges::mcols(gr)[, c('Variant','Pvalue')])

## Not run:
# b) perform priority analysis
pNode <- xPierSNPs(data=AS, include.eQTL="JKng_mono",
include.HiC='Monocytes', network="PCommonsUN_medium", restart=0.7,
RData.location=RData.location)

# c) derive pathway-level priority
eTerm <- xPierPathways(pNode=pNode, priority.top=100,
ontology="MsigdbC2CP", RData.location=RData.location)

# d) view enrichment results for the top significant terms
xEnrichViewer(eTerm)

# e) save enrichment results to the file called 'Pathways_priority.txt'
res <- xEnrichViewer(eTerm, top_num=length(eTerm$adjp), sortBy="adjp",
details=TRUE)
output <- data.frame(term=rownames(res), res)
utils::write.table(output, file="Pathways_priority.txt", sep="\t",
row.names=FALSE)

## End(Not run)
```

---

xPierSNPs *Function to prioritise genes given a list of seed SNPs together with the significance level (e.g. GWAS reported p-values)*

Description

xPierSNPs is supposed to prioritise genes given a list of seed SNPs together with the significance level. To prioritise genes, it first defines and scores seed genes: nearby genes and eQTL genes. With seed genes and their scores, it then uses Random Walk with Restart (RWR) to calculate the affinity score of all nodes in the input graph to the seed genes. The priority score is the affinity score. Parallel computing is also supported for Linux-like or Windows operating systems. It returns an object of class "pNode".

Usage

```
xPierSNPs(data, include.LD = NA, LD.customised = NULL, LD.r2 = 0.8,
significance.threshold = 5e-05, score.cap = 10, distance.max = 2e+05,
decay.kernel = c("rapid", "slow", "linear", "constant"),
decay.exponent = 2, GR.SNP = c("dbSNP_GWAS", "dbSNP_Common"),
GR.Gene = c("UCSC_knownGene", "UCSC_knownCanonical"), include.eQTL =
c(NA,
"JKscience_TS2A", "JKscience_TS2A_CD14", "JKscience_TS2A_LPS2",
"JKscience_TS2A_LPS24", "JKscience_TS2A_IFN", "JKscience_TS2B",
"JKscience_TS2B_CD14", "JKscience_TS2B_LPS2", "JKscience_TS2B_LPS24",
"JKscience_TS2B_IFN", "JKscience_TS3A", "JKng_bcell", "JKng_bcell_cis",
"JKng_bcell_trans", "JKng_mono", "JKng_mono_cis", "JKng_mono_trans",
"JKnc_neutro", "JKnc_neutro_cis", "JKnc_neutro_trans", "JK_nk",
"GTEx_V4_Adipose_Subcutaneous", "GTEx_V4_Artery_Aorta",
"GTEx_V4_Artery_Tibial", "GTEx_V4_Esophagus_Mucosa",
"GTEx_V4_Esophagus_Muscularis", "GTEx_V4_Heart_Left_Ventricle",
"GTEx_V4_Lung", "GTEx_V4_Muscle_Skeletal", "GTEx_V4_Nerve_Tibial",
"GTEx_V4_Skin_Sun_Exposed_Lower_leg", "GTEx_V4_Stomach",
"GTEx_V4_Thyroid",
"GTEx_V4_Whole_Blood", "eQTLdb_NK", "eQTLdb_CD14", "eQTLdb_LPS2",
"eQTLdb_LPS24", "eQTLdb_IFN"), eQTL.customised = NULL, include.HiC =
c(NA,
"Monocytes", "Macrophages_M0", "Macrophages_M1", "Macrophages_M2",
"Neutrophils", "Megakaryocytes", "Endothelial_precursors",
"Erythroblasts",
"Fetal_thymus", "Naive_CD4_T_cells", "Total_CD4_T_cells",
"Activated_total_CD4_T_cells", "Nonactivated_total_CD4_T_cells",
"Naive_CD8_T_cells", "Total_CD8_T_cells", "Naive_B_cells",
"Total_B_cells",
"PE.Monocytes", "PE.Macrophages_M0", "PE.Macrophages_M1",
"PE.Macrophages_M2",
"PE.Neutrophils", "PE.Megakaryocytes", "PE.Erythroblasts",
"PE.Naive_CD4_T_cells", "PE.Naive_CD8_T_cells"),
cdf.function = c("empirical", "exponential"), relative.importance =
c(1/3,
1/3, 1/3), scoring.scheme = c("max", "sum", "sequential"),
network = c("STRING_highest", "STRING_high", "STRING_medium",
"STRING_low",
"PCommonsUN_high", "PCommonsUN_medium", "PCommonsDN_high",
"PCommonsDN_medium", "PCommonsDN_Reactome", "PCommonsDN_KEGG",
"PCommonsDN_HumanCyc", "PCommonsDN_PID", "PCommonsDN_PANTHER",
"PCommonsDN_ReconX", "PCommonsDN_TRANSFAC", "PCommonsDN_PhosphoSite",
"PCommonsDN_CTD"), weighted = FALSE, network.customised = NULL,
seeds.inclusive = TRUE, normalise = c("laplacian", "row", "column",
"none"), restart = 0.75, normalise.affinity.matrix = c("none",
"quantile"), parallel = TRUE, multicores = NULL, verbose = TRUE,
RData.location = "http://galahad.well.ox.ac.uk/bigdata")
```

Arguments

`data` a named input vector containing the sinificance level for nodes (dbSNP). For this named vector, the element names are dbSNP ID (or in the format such as

’chr16:28525386’), the element values for the significance level (measured as p-value or fdr). Alternatively, it can be a matrix or data frame with two columns: 1st column for dbSNP, 2nd column for the significance level

`include.LD` additional SNPs in LD with Lead SNPs are also included. By default, it is ’NA’ to disable this option. Otherwise, LD SNPs will be included based on one or more of 26 populations and 5 super populations from 1000 Genomics Project data (phase 3). The population can be one of 5 super populations ("AFR", "AMR", "EAS", "EUR", "SAS"), or one of 26 populations ("ACB", "ASW", "BEB", "CDX", "CEU", "CHB", "CHS", "CLM", "ESN", "FIN", "GBR", "GIH", "GWD", "IBS", "ITU", "JPT", "KHV", "LWK", "MSL", "MXL", "PEL", "PJL", "PUR", "STU", "TSI", "YRI"). Explanations for population code can be found at `http://www.1000genomes.org/faq/which-populations-are-part-your-study`

`LD.customised` a user-input matrix or data frame with 3 columns: 1st column for Lead SNPs, 2nd column for LD SNPs, and 3rd for LD r2 value. It is designed to allow the user analysing their pre-calculated LD info. This customisation (if provided) has the high priority over built-in LD SNPs

`LD.r2` the LD r2 value. By default, it is 0.8, meaning that SNPs in LD (r2>=0.8) with input SNPs will be considered as LD SNPs. It can be any value from 0.8 to 1

`significance.threshold`
the given significance threshold. By default, it is set to NULL, meaning there is no constraint on the significance level when transforming the significance level of SNPs into scores. If given, those SNPs below this are considered significant and thus scored positively. Instead, those above this are considered insigificant and thus receive no score

`score.cap` the maximum score being capped. By default, it is set to 10. If NULL, no capping is applied

`distance.max` the maximum distance between genes and SNPs. Only those genes no far way from this distance will be considered as seed genes. This parameter will influence the distance-component weights calculated for nearby SNPs per gene

`decay.kernel` a character specifying a decay kernel function. It can be one of ’slow’ for slow decay, ’linear’ for linear decay, and ’rapid’ for rapid decay. If no distance weight is used, please select ’constant’

`decay.exponent` an integer specifying a decay exponent. By default, it sets to 2

`GR.SNP` the genomic regions of SNPs. By default, it is ’dbSNP_GWAS’, that is, SNPs from dbSNP (version 146) restricted to GWAS SNPs and their LD SNPs (hg19). It can be ’dbSNP_Common’, that is, Common SNPs from dbSNP (version 146) plus GWAS SNPs and their LD SNPs (hg19). Alternatively, the user can specify the customised input. To do so, first save your RData file (containing an GR object) into your local computer, and make sure the GR object content names refer to dbSNP IDs. Then, tell "GR.SNP" with your RData file name (with or without extension), plus specify your file RData path in "RData.location". Note: you can also load your customised GR object directly

`GR.Gene` the genomic regions of genes. By default, it is ’UCSC_knownGene’, that is, UCSC known genes (together with genomic locations) based on human genome assembly hg19. It can be ’UCSC_knownCanonical’, that is, UCSC known canonical genes (together with genomic locations) based on human genome assembly hg19. Alternatively, the user can specify the customised input. To do so, first save your RData file (containing an GR object) into your local computer, and make sure the GR object content names refer to Gene Symbols. Then, tell "GR.Gene" with your RData file name (with or without extension), plus specify your file RData path in "RData.location". Note: you can also load your customised GR object directly

`include.eQTL` genes modulated by eQTL (also Lead SNPs or in LD with Lead SNPs) are also included. By default, it is 'NA' to disable this option. Otherwise, those genes modulated by eQTL will be included. Pre-built eQTL datasets are detailed in the section 'Note'

`eQTL.customised`
a user-input matrix or data frame with 3 columns: 1st column for SNPs/eQTLs, 2nd column for Genes, and 3rd for eQTL mapping significance level (p-values or FDR). It is designed to allow the user analysing their eQTL data. This customisation (if provided) has the high priority over built-in eQTL data

`include.HiC` genes linked to input SNPs are also included. By default, it is 'NA' to disable this option. Otherwise, those genes linked to SNPs will be included according to Promoter Capture HiC (PCHiC) datasets. Pre-built HiC datasets are detailed in the section 'Note'

`cdf.function` a character specifying a Cumulative Distribution Function (cdf). It can be one of 'exponential' based on exponential cdf, 'empirical' for empirical cdf

`relative.importance`
a vector specifying the relative importance of nearby genes, eQTL genes and HiC genes. By default, it sets c(1/3, 1/3, 1/3)

`scoring.scheme` the method used to calculate seed gene scores under a set of SNPs. It can be one of "sum" for adding up, "max" for the maximum, and "sequential" for the sequential weighting. The sequential weighting is done via: $\sum_{i=1} \frac{R_i}{i}$, where R_i is the i^{th} rank (in a descreasing order)

`network` the built-in network. Currently two sources of network information are supported: the STRING database (version 10) and the Pathways Commons database (version 7). STRING is a meta-integration of undirect interactions from the functional aspect, while Pathways Commons mainly contains both undirect and direct interactions from the physical/pathway aspect. Both have scores to control the confidence of interactions. Therefore, the user can choose the different quality of the interactions. In STRING, "STRING_highest" indicates interactions with highest confidence (confidence scores>=900), "STRING_high" for interactions with high confidence (confidence scores>=700), "STRING_medium" for interactions with medium confidence (confidence scores>=400), and "STRING_low" for interactions with low confidence (confidence scores>=150). For undirect/physical interactions from Pathways Commons, "PCommonsUN_high" indicates undirect interactions with high confidence (supported with the PubMed references plus at least 2 different sources), "PCommonsUN_medium" for undirect interactions with medium confidence (supported with the PubMed references). For direct (pathway-merged) interactions from Pathways Commons, "PCommonsDN_high" indicates direct interactions with high confidence (supported with the PubMed references plus at least 2 different sources), and "PCommonsUN_medium" for direct interactions with medium confidence (supported with the PubMed references). In addtion to pooled version of pathways from all data sources, the user can also choose the pathway-merged network from individual sources, that is, "PCommonsDN_Reactome" for those from Reactome, "PCommonsDN_KEGG" for those from KEGG, "PCommonsDN_HumanCyc" for those from HumanCyc, "PCommonsDN_PID" for those froom PID, "PCommonsDN_PANTHER" for those from PANTHER, "PCommonsDN_ReconX" for those from ReconX, "PCommonsDN_TRANSFAC" for those from TRANSFAC, "PCommonsDN_PhosphoSite" for those from PhosphoSite, and "PCommonsDN_CTD" for those from CTD `weighted` logical to indicate whether edge weights should be considered. By default, it sets to false. If true, it only works for the network from the STRING database

`network.customised`
an object of class "igraph". By default, it is NULL. It is designed to allow the user analysing their customised network data that are not listed in the above argument 'network'. This customisation (if provided) has the high priority over built-in network. If the user provides the "igraph" object with the "weight" edge attribute, RWR will assume to walk on the weighted network

`seeds.inclusive`
logical to indicate whether non-network seed genes are included for prioritisation. If TRUE (by default), these genes will be added to the netowrk

`normalise` the way to normalise the adjacency matrix of the input graph. It can be 'laplacian' for laplacian normalisation, 'row' for row-wise normalisation, 'column' for column-wise normalisation, or 'none'

`restart` the restart probability used for Random Walk with Restart (RWR). The restart probability takes the value from 0 to 1, controlling the range from the starting nodes/seeds that the walker will explore. The higher the value, the more likely the walker is to visit the nodes centered on the starting nodes. At the extreme when the restart probability is zero, the walker moves freely to the neighbors at each step without restarting from seeds, i.e., following a random walk (RW)

`normalise.affinity.matrix`
the way to normalise the output affinity matrix. It can be 'none' for no normalisation, 'quantile' for quantile normalisation to ensure that columns (if multiple) of the output affinity matrix have the same quantiles

`parallel` logical to indicate whether parallel computation with multicores is used. By default, it sets to true, but not necessarily does so. Partly because parallel backends available will be system-specific (now only Linux or Mac OS). Also, it will depend on whether these two packages "foreach" and "doMC" have been installed. It can be installed via: `source("http://bioconductor.org/biocLite.R"); biocLite(c("foreach","doMC"))`. If not yet installed, this option will be disabled

`multicores` an integer to specify how many cores will be registered as the multicore parallel backend to the 'foreach' package. If NULL, it will use a half of cores available in a user's computer. This option only works when parallel computation is enabled

`verbose` logical to indicate whether the messages will be displayed in the screen. By default, it sets to true for display

`RData.location` the characters to tell the location of built-in RData files. See `xRDataLoader` for details

Value

an object of class "pNode", a list with following components:

- `priority`: a matrix of nNode X 6 containing node priority information, where nNode is the number of nodes in the input graph, and the 6 columns are "name" (node names), "node" (1 for network genes, 0 for non-network seed genes), "seed" (1 for seeds, 0 for non-seeds), "weight" (weight values), "priority" (the priority scores that are rescaled to the range [0,1]), "rank" (ranks of the priority scores), "description" (node description)
- `g`: an input "igraph" object

- SNP: a data frame of nSNP X 4 containing input SNPs and/or LD SNPs info, where nSNP is the number of input SNPs and/or LD SNPs, and the 4 columns are "SNP" (dbSNP), "Score" (the SNP score), "Pval" (the SNP p-value), "Flag" (indicative of Lead SNPs or LD SNPs)
- Gene2SNP: a data frame of nPair X 3 containing Gene-SNP pair info, where nPair is the number of Gene-SNP pairs, and the 3 columns are "Gene" (seed genes), "SNP" (dbSNP), "Score" (an SNP's genetic influential score on a seed gene), "Pval" (the SNP p-value)
- nGenes: if not NULL, it is a data frame containing nGene-SNP pair info
- eGenes: if not NULL, it is a data frame containing eGene-SNP pair info per context
- cGenes: if not NULL, it is a data frame containing cGene-SNP pair info per context
- call: the call that produced this result

Note

The prioritisation procedure (from SNPs to target genes) consists of following steps:

- i) xSNPscores used to calculate the SNP score.
- ii) xSNP2nGenes used to define and score the nearby genes.
- iii) xSNP2eGenes used to define and score the eQTL genes.
- iv) xSNP2cGenes used to define and score the HiC genes.
- v) define seed genes as the nearby genes in ii) and the eQTL genes in iii) and the HiC genes in iv), which are then scored in an integrative manner.
- vi) xPierGenes used to prioritise genes using an input graph and a list of seed genes and their scores from v). The priority score is the affinity score estimated by Random Walk with Restart (RWR), measured as the affinity of all nodes in the graph to the seeds.

Pre-built eQTL datasets are described below according to the data sources.
1. Context-specific eQTLs in monocytes: resting and activating states. Sourced from Science 2014, 343(6175):1246949

- JKscience_TS2A: cis-eQTLs in either state (based on 228 individuals with expression data available for all experimental conditions).
- JKscience_TS2A_CD14: cis-eQTLs only in the resting/CD14+ state (based on 228 individuals).
- JKscience_TS2A_LPS2: cis-eQTLs only in the activating state induced by 2-hour LPS (based on 228 individuals).
- JKscience_TS2A_LPS24: cis-eQTLs only in the activating state induced by 24-hour LPS (based on 228 individuals).
- JKscience_TS2A_IFN: cis-eQTLs only in the activating state induced by 24-hour interferon-gamma (based on 228 individuals).
- JKscience_TS2B: cis-eQTLs in either state (based on 432 individuals).
- JKscience_TS2B_CD14: cis-eQTLs only in the resting/CD14+ state (based on 432 individuals).
- JKscience_TS2B_LPS2: cis-eQTLs only in the activating state induced by 2-hour LPS (based on 432 individuals).
- JKscience_TS2B_LPS24: cis-eQTLs only in the activating state induced by 24-hour LPS (based on 432 individuals).
- JKscience_TS2B_IFN: cis-eQTLs only in the activating state induced by 24-hour interferon-gamma (based on 432 individuals).

- JKscience_TS3A: trans-eQTLs in either state.

2. eQTLs in B cells. Sourced from Nature Genetics 2012, 44(5):502-510

- JKng_bcell: cis- and trans-eQTLs.
- JKng_bcell_cis: cis-eQTLs only.
- JKng_bcell_trans: trans-eQTLs only.

3. eQTLs in monocytes. Sourced from Nature Genetics 2012, 44(5):502-510

- JKng_mono: cis- and trans-eQTLs.
- JKng_mono_cis: cis-eQTLs only.
- JKng_mono_trans: trans-eQTLs only.

4. eQTLs in neutrophils. Sourced from Nature Communications 2015, 7(6):7545

- JKnc_neutro: cis- and trans-eQTLs.
- JKnc_neutro_cis: cis-eQTLs only.
- JKnc_neutro_trans: trans-eQTLs only.

5. eQTLs in NK cells. Unpublished

- JK_nk: cis-eQTLs.

6. Tissue-specific eQTLs from GTEx (version 4; incuding 13 tissues). Sourced from Science 2015, 348(6235):648-60

- GTEx_V4_Adipose_Subcutaneous: cis-eQTLs in tissue 'Adipose Subcutaneous'.
- GTEx_V4_Artery_Aorta: cis-eQTLs in tissue 'Artery Aorta'.
- GTEx_V4_Artery_Tibial: cis-eQTLs in tissue 'Artery Tibial'.
- GTEx_V4_Esophagus_Mucosa: cis-eQTLs in tissue 'Esophagus Mucosa'.
- GTEx_V4_Esophagus_Muscularis: cis-eQTLs in tissue 'Esophagus Muscularis'.
- GTEx_V4_Heart_Left_Ventricle: cis-eQTLs in tissue 'Heart Left Ventricle'.
- GTEx_V4_Lung: cis-eQTLs in tissue 'Lung'.
- GTEx_V4_Muscle_Skeletal: cis-eQTLs in tissue 'Muscle Skeletal'.
- GTEx_V4_Nerve_Tibial: cis-eQTLs in tissue 'Nerve Tibial'.
- GTEx_V4_Skin_Sun_Exposed_Lower_leg: cis-eQTLs in tissue 'Skin Sun Exposed Lower leg'.
- GTEx_V4_Stomach: cis-eQTLs in tissue 'Stomach'.
- GTEx_V4_Thyroid: cis-eQTLs in tissue 'Thyroid'.
- GTEx_V4_Whole_Blood: cis-eQTLs in tissue 'Whole Blood'.

Pre-built HiC datasets are described below according to the data sources.
1. Promoter Capture HiC datasets in 17 primary blood cell types. Sourced from Cell 2016, 167(5):1369-1384.e19

- Monocytes: physical interactions (CHiCAGO score >=5) of promoters (baits) with the other end (preys) in Monocytes.
- Macrophages_M0: promoter interactomes in Macrophages M0.
- Macrophages_M1: promoter interactomes in Macrophages M1.

- Macrophages_M2: promoter interactomes in Macrophages M2.
- Neutrophils: promoter interactomes in Neutrophils.
- Megakaryocytes: promoter interactomes in Megakaryocytes.
- Endothelial_precursors: promoter interactomes in Endothelial precursors.
- Fetal_thymus: promoter interactomes in Fetal thymus.
- Naive_CD4_T_cells: promoter interactomes in Naive CD4+ T cells.
- Total_CD4_T_cells: promoter interactomes in Total CD4+ T cells.
- Activated_total_CD4_T_cells: promoter interactomes in Activated total CD4+ T cells.
- Nonactivated_total_CD4_T_cells: promoter interactomes in Nonactivated total CD4+ T cells.
- Naive_CD8_T_cells: promoter interactomes in Naive CD8+ T cells.
- Total_CD8_T_cells: promoter interactomes in Total CD8+ T cells.
- Naive_B_cells: promoter interactomes in Naive B cells.
- Total_B_cells: promoter interactomes in Total B cells.

2. Promoter Capture HiC datasets (involving active promoters and enhancers) in 9 primary blood cell types. Sourced from Cell 2016, 167(5):1369-1384.e19

- PE.Monocytes: physical interactions (CHiCAGO score >=5) of promoters (baits) with the other end (enhancers as preys) in Monocytes.
- PE.Macrophages_M0: promoter-enhancer interactomes in Macrophages M0.
- PE.Macrophages_M1: promoter-enhancer interactomes in Macrophages M1.
- PE.Macrophages_M2: promoter-enhancer interactomes in Macrophages M2.
- PE.Neutrophils: promoter-enhancer interactomes in Neutrophils.
- PE.Megakaryocytes: promoter-enhancer interactomes in Megakaryocytes.
- PE.Erythroblasts: promoter-enhancer interactomes in Erythroblasts.
- PE.Naive_CD4_T_cells: promoter-enhancer interactomes in Naive CD4+ T cells.
- PE.Naive_CD8_T_cells: promoter-enhancer interactomes in Naive CD8+ T cells.

See Also

xSNPscores, xSNP2nGenes, xSNP2eGenes, xSNP2cGenes, xSparseMatrix, xSM2DF, xPier, xPierGenes, xPierPathways

Examples

```
## Not run:
# Load the library
library(Pi)

## End(Not run)

RData.location <- "http://galahad.well.ox.ac.uk/bigdata_dev"
# a) provide the SNPs with the significance info
## get lead SNPs reported in AS GWAS and their significance info (p-values)
#data.file <- "http://galahad.well.ox.ac.uk/bigdata/AS.txt"
#AS <- read.delim(data.file, header=TRUE, stringsAsFactors=FALSE)
ImmunoBase <- xRDataLoader(RData.customised='ImmunoBase',
RData.location=RData.location)
```

```
gr <- ImmunoBase$AS$variants
AS <- as.data.frame(GenomicRanges::mcols(gr)[, c('Variant','Pvalue')])

## Not run:
# b) perform priority analysis
pNode <- xPierSNPs(data=AS, include.eQTL="JKng_mono",
include.HiC='Monocytes', network="PCommonsUN_medium", restart=0.7,
RData.location=RData.location)

# c) save to the file called 'SNPs_priority.txt'
write.table(pNode$priority, file="SNPs_priority.txt", sep="\t",
row.names=FALSE)

# d) manhattan plot
mp <- xPierManhattan(pNode, top=20, top.label.size=1.5, y.scale="sqrt",
RData.location=RData.location)
#pdf(file="Gene_manhattan.pdf", height=6, width=12, compress=TRUE)
print(mp)
#dev.off()

## End(Not run)
```

xPierSNPsAdv

Function to prepare genetic predictors given a list of seed SNPs together with the significance level (e.g. GWAS reported p-values)

Description

xPierSNPsAdv is supposed to prepare genetic predictors given a list of seed SNPs together with the significance level (e.g. GWAS reported p-values). Internally it calls xPierSNPs to prepare the distance predictor, the eQTL predictors (if required) and the HiC predictors (if required). It returns a list of class "pNode" objects.

Usage

```
xPierSNPsAdv(data, include.LD = NA, LD.customised = NULL, LD.r2 = 0.8,
significance.threshold = 5e-05, score.cap = 10, distance.max = 1000,
decay.kernel = c("constant", "slow", "linear", "rapid"),
decay.exponent = 2, GR.SNP = c("dbSNP_GWAS", "dbSNP_Common"),
GR.Gene = c("UCSC_knownGene", "UCSC_knownCanonical"), include.eQTL =
c(NA,
"JKscience_TS2A", "JKscience_TS2A_CD14", "JKscience_TS2A_LPS2",
"JKscience_TS2A_LPS24", "JKscience_TS2A_IFN", "JKscience_TS2B",
"JKscience_TS2B_CD14", "JKscience_TS2B_LPS2", "JKscience_TS2B_LPS24",
"JKscience_TS2B_IFN", "JKscience_TS3A", "JKng_bcell", "JKng_bcell_cis",
"JKng_bcell_trans", "JKng_mono", "JKng_mono_cis", "JKng_mono_trans",
"JKnc_neutro", "JKnc_neutro_cis", "JKnc_neutro_trans", "JK_nk",
"GTEx_V4_Adipose_Subcutaneous", "GTEx_V4_Artery_Aorta",
"GTEx_V4_Artery_Tibial", "GTEx_V4_Esophagus_Mucosa",
"GTEx_V4_Esophagus_Muscularis", "GTEx_V4_Heart_Left_Ventricle",
"GTEx_V4_Lung", "GTEx_V4_Muscle_Skeletal", "GTEx_V4_Nerve_Tibial",
"GTEx_V4_Skin_Sun_Exposed_Lower_leg", "GTEx_V4_Stomach",
"GTEx_V4_Thyroid",
"GTEx_V4_Whole_Blood", "eQTLdb_NK", "eQTLdb_CD14", "eQTLdb_LPS2",
"eQTLdb_LPS24", "eQTLdb_IFN"), eQTL.customised = NULL, include.HiC =
c(NA,
"Monocytes", "Macrophages_M0", "Macrophages_M1", "Macrophages_M2",
"Neutrophils", "Megakaryocytes", "Endothelial_precursors",
"Erythroblasts",
"Fetal_thymus", "Naive_CD4_T_cells", "Total_CD4_T_cells",
"Activated_total_CD4_T_cells", "Nonactivated_total_CD4_T_cells",
"Naive_CD8_T_cells", "Total_CD8_T_cells", "Naive_B_cells",
"Total_B_cells",
"PE.Monocytes", "PE.Macrophages_M0", "PE.Macrophages_M1",
"PE.Macrophages_M2",
"PE.Neutrophils", "PE.Megakaryocytes", "PE.Erythroblasts",
"PE.Naive_CD4_T_cells", "PE.Naive_CD8_T_cells"),
cdf.function = c("empirical", "exponential"), scoring.scheme = c("max",
"sum", "sequential"), network = c("STRING_highest", "STRING_high",
"STRING_medium", "STRING_low", "PCommonsUN_high", "PCommonsUN_medium",
"PCommonsDN_high", "PCommonsDN_medium", "PCommonsDN_Reactome",
"PCommonsDN_KEGG", "PCommonsDN_HumanCyc", "PCommonsDN_PID",
"PCommonsDN_PANTHER", "PCommonsDN_ReconX", "PCommonsDN_TRANSFAC",
"PCommonsDN_PhosphoSite", "PCommonsDN_CTD"), weighted = FALSE,
network.customised = NULL, seeds.inclusive = TRUE,
normalise = c("laplacian", "row", "column", "none"), restart = 0.75,
normalise.affinity.matrix = c("none", "quantile"), parallel = TRUE,
multicores = NULL, verbose = TRUE, verbose.details = FALSE,
RData.location = "http://galahad.well.ox.ac.uk/bigdata")
```

Arguments

`data` a named input vector containing the sinificance level for nodes (dbSNP). For this named vector, the element names are dbSNP ID (or in the format such as 'chr16:28525386'), the element values for the significance level (measured as p-value or fdr). Alternatively, it can be a matrix or data frame with two columns: 1st column for dbSNP, 2nd column for the significance level

`include.LD` additional SNPs in LD with Lead SNPs are also included. By default, it is 'NA' to disable this option. Otherwise, LD SNPs will be included based on one or more of 5 super-populations from 1000 Genomics Project data (phase 3). They are "AFR", "AMR", "EAS", "EUR", and "SAS". Explanations for population code can be found at http://www.1000genomes.org/faq/which-populations-are-part-your-s

`LD.customised` a user-input matrix or data frame with 3 columns: 1st column for Lead SNPs, 2nd column for LD SNPs, and 3rd for LD r2 value. It is designed to allow the user analysing their pre-calculated LD info. This customisation (if provided) has the high priority over built-in LD SNPs

`LD.r2` the LD r2 value. By default, it is 0.8, meaning that SNPs in LD (r2>=0.8) with input SNPs will be considered as LD SNPs. It can be any value from 0.8 to 1

`significance.threshold`
the given significance threshold. By default, it is set to NULL, meaning there is no constraint on the significance level when transforming the significance level of SNPs into scores. If given, those SNPs below this are considered significant and thus scored positively. Instead, those above this are considered insigificant and thus receive no score

`score.cap` the maximum score being capped. By default, it is set to 10. If NULL, no capping is applied

`distance.max` the maximum distance between genes and SNPs. Only those genes no far way from this distance will be considered as seed genes. This parameter will influence the distance-component weights calculated for nearby SNPs per gene

`decay.kernel` a character specifying a decay kernel function. It can be one of 'slow' for slow decay, 'linear' for linear decay, and 'rapid' for rapid decay. If no distance weight is used, please select 'constant'

`decay.exponent` an integer specifying a decay exponent. By default, it sets to 2

`GR.SNP` the genomic regions of SNPs. By default, it is 'dbSNP_GWAS', that is, SNPs from dbSNP (version 146) restricted to GWAS SNPs and their LD SNPs (hg19). It can be 'dbSNP_Common', that is, Common SNPs from dbSNP (version 146) plus GWAS SNPs and their LD SNPs (hg19). Alternatively, the user can specify the customised input. To do so, first save your RData file (containing an GR object) into your local computer, and make sure the GR object content names refer to dbSNP IDs. Then, tell "GR.SNP" with your RData file name (with or without extension), plus specify your file RData path in "RData.location". Note: you can also load your customised GR object directly

`GR.Gene` the genomic regions of genes. By default, it is 'UCSC_knownGene', that is, UCSC known genes (together with genomic locations) based on human genome assembly hg19. It can be 'UCSC_knownCanonical', that is, UCSC known canonical genes (together with genomic locations) based on human genome assembly hg19. Alternatively, the user can specify the customised input. To do so, first save your RData file (containing an GR object) into your local computer, and make sure the GR object content names refer to Gene Symbols. Then, tell "GR.Gene" with your RData file name (with or without extension), plus specify your file RData path in "RData.location". Note: you can also load your customised GR object directly

`include.eQTL` genes modulated by eQTL (also Lead SNPs or in LD with Lead SNPs) are also included. By default, it is 'NA' to disable this option. Otherwise, those genes modulated by eQTL will be included. Pre-built eQTL datasets are detailed in the section 'Note'

`eQTL.customised`

a user-input matrix or data frame with 3 columns: 1st column for SNPs/eQTLs, 2nd column for Genes, and 3rd for eQTL mapping significance level (p-values or FDR). It is designed to allow the user analysing their eQTL data. This customisation (if provided) has the high priority over built-in eQTL data

`include.HiC` genes linked to input SNPs are also included. By default, it is 'NA' to disable this option. Otherwise, those genes linked to SNPs will be included according to Promoter Capture HiC (PCHiC) datasets. Pre-built HiC datasets are detailed in the section 'Note'

`cdf.function` a character specifying a Cumulative Distribution Function (cdf). It can be one of 'exponential' based on exponential cdf, 'empirical' for empirical cdf

`scoring.scheme` the method used to calculate seed gene scores under a set of SNPs. It can be one of "sum" for adding up, "max" for the maximum, and "sequential" for the sequential weighting. The sequential weighting is done via: $\sum_{i=1} \frac{R_i}{i}$, where R_i is the i^{th} rank (in a descreasing order)

`network` the built-in network. Currently two sources of network information are supported: the STRING database (version 10) and the Pathways Commons database (version 7). STRING is a meta-integration of undirect interactions from the functional aspect, while Pathways Commons mainly contains both undirect and direct interactions from the physical/pathway aspect. Both have scores to control the confidence of interactions. Therefore, the user can choose the different quality of the interactions. In STRING, "STRING_highest" indicates interactions with highest confidence (confidence scores>=900), "STRING_high" for interactions with high confidence (confidence scores>=700), "STRING_medium" for interactions with medium confidence (confidence scores>=400), and "STRING_low" for interactions with low confidence (confidence scores>=150). For undirect/physical interactions from Pathways Commons, "PCommonsUN_high" indicates undirect interactions with high confidence (supported with the PubMed references plus at least 2 different sources), "PCommonsUN_medium" for undirect interactions with medium confidence (supported with the PubMed references). For direct (pathway-merged) interactions from Pathways Commons, "PCommonsDN_high" indicates direct interactions with high confidence (supported with the PubMed references plus at least 2 different sources), and "PCommonsUN_medium" for direct interactions with medium confidence (supported with the PubMed references). In addtion to pooled version of pathways from all data sources, the user can also choose the pathway-merged network from individual sources, that is, "PCommonsDN_Reactome" for those from Reactome, "PCommonsDN_KEGG" for those from KEGG, "PCommonsDN_HumanCyc" for those from HumanCyc, "PCommonsDN_PID" for those froom PID, "PCommonsDN_PANTHER" for those from PANTHER, "PCommonsDN_ReconX" for those from ReconX, "PCommonsDN_TRANSFAC" for those from TRANSFAC, "PCommonsDN_PhosphoSite" for those from PhosphoSite, and "PCommonsDN_CTD" for those from CTD

`weighted` logical to indicate whether edge weights should be considered. By default, it sets to false. If true, it only works for the network from the STRING database

`network.customised`

an object of class "igraph". By default, it is NULL. It is designed to allow the user analysing their customised network data that are not listed in the above argument 'network'. This customisation (if provided) has the high priority over built-in network. If the user provides the "igraph" object with the "weight" edge attribute, RWR will assume to walk on the weighted network

`seeds.inclusive`

logical to indicate whether non-network seed genes are included for prioritisation. If TRUE (by default), these genes will be added to the netowrk

`normalise` the way to normalise the adjacency matrix of the input graph. It can be 'laplacian' for laplacian normalisation, 'row' for row-wise normalisation, 'column' for column-wise normalisation, or 'none'

`restart` the restart probability used for Random Walk with Restart (RWR). The restart probability takes the value from 0 to 1, controlling the range from the starting nodes/seeds that the walker will explore. The higher the value, the more likely the walker is to visit the nodes centered on the starting nodes. At the extreme when the restart probability is zero, the walker moves freely to the neighbors at each step without restarting from seeds, i.e., following a random walk (RW)

`normalise.affinity.matrix`

the way to normalise the output affinity matrix. It can be 'none' for no normalisation, 'quantile' for quantile normalisation to ensure that columns (if multiple) of the output affinity matrix have the same quantiles `parallel` logical to indicate whether parallel computation with multicores is used. By default, it sets to true, but not necessarily does so. Partly because parallel backends available will be system-specific (now only Linux or Mac OS). Also, it will depend on whether these two packages "foreach" and "doMC" have been installed. It can be installed via: `source("http://bioconductor.org/biocLite.R"); biocLite(c("foreach","doMC"))`. If not yet installed, this option will be disabled

`multicores` an integer to specify how many cores will be registered as the multicore parallel backend to the 'foreach' package. If NULL, it will use a half of cores available in a user's computer. This option only works when parallel computation is enabled

`verbose` logical to indicate whether the messages will be displayed in the screen. By default, it sets to true for display

`verbose.details` logical to indicate whether the detailed messages from being-called functions will be displayed in the screen. By default, it sets to FALSE enabling messages

`RData.location` the characters to tell the location of built-in RData files. See `xRDataLoader` for details

Value

A list of class "pNode" objects, each object having a list with following components:

- `priority`: a matrix of nNode X 6 containing node priority information, where nNode is the number of nodes in the input graph, and the 6 columns are "name" (node names), "node" (1 for network genes, 0 for non-network seed genes), "seed" (1 for seeds, 0 for non-seeds), "weight" (weight values), "priority" (the priority scores that are rescaled to the range [0,1]), "rank" (ranks of the priority scores), "description" (node description)
- `g`: an input "igraph" object
- `SNP`: a data frame of nSNP X 4 containing input SNPs and/or LD SNPs info, where nSNP is the number of input SNPs and/or LD SNPs, and the 4 columns are "SNP" (dbSNP), "Score" (the SNP score), "Pval" (the SNP p-value), "Flag" (indicative of Lead SNPs or LD SNPs)
- `Gene2SNP`: a data frame of nPair X 3 containing Gene-SNP pair info, where nPair is the number of Gene-SNP pairs, and the 3 columns are "Gene" (seed genes), "SNP" (dbSNP), "Score" (an SNP's genetic influential score on a seed gene), "Pval" (the SNP p-value)
- `nGenes`: if not NULL, it is a data frame containing nGene-SNP pair info
- `eGenes`: if not NULL, it is a data frame containing eGene-SNP pair info per context
- `cGenes`: if not NULL, it is a data frame containing cGene-SNP pair info per context
- `call`: the call that produced this result

Note

This function calls `xPierSNPs` in a loop way generating the distance predictor, the eQTL predictors (if required) and the HiC predictors (if required). Pre-built eQTL datasets are described below according to the data sources.
1. Context-specific eQTLs in monocytes: resting and activating states. Sourced from Science 2014, 343(6175):1246949

- `JKscience_TS2A`: cis-eQTLs in either state (based on 228 individuals with expression data available for all experimental conditions).

- JKscience_TS2A_CD14: cis-eQTLs only in the resting/CD14+ state (based on 228 individuals).
- JKscience_TS2A_LPS2: cis-eQTLs only in the activating state induced by 2-hour LPS (based on 228 individuals).
- JKscience_TS2A_LPS24: cis-eQTLs only in the activating state induced by 24-hour LPS (based on 228 individuals).
- JKscience_TS2A_IFN: cis-eQTLs only in the activating state induced by 24-hour interferon-gamma (based on 228 individuals).
- JKscience_TS2B: cis-eQTLs in either state (based on 432 individuals).
- JKscience_TS2B_CD14: cis-eQTLs only in the resting/CD14+ state (based on 432 individuals).
- JKscience_TS2B_LPS2: cis-eQTLs only in the activating state induced by 2-hour LPS (based on 432 individuals).
- JKscience_TS2B_LPS24: cis-eQTLs only in the activating state induced by 24-hour LPS (based on 432 individuals).
- JKscience_TS2B_IFN: cis-eQTLs only in the activating state induced by 24-hour interferon-gamma (based on 432 individuals).
- JKscience_TS3A: trans-eQTLs in either state.

2. eQTLs in B cells. Sourced from Nature Genetics 2012, 44(5):502-510

- JKng_bcell: cis- and trans-eQTLs.
- JKng_bcell_cis: cis-eQTLs only.
- JKng_bcell_trans: trans-eQTLs only.

3. eQTLs in monocytes. Sourced from Nature Genetics 2012, 44(5):502-510

- JKng_mono: cis- and trans-eQTLs.
- JKng_mono_cis: cis-eQTLs only.
- JKng_mono_trans: trans-eQTLs only.

4. eQTLs in neutrophils. Sourced from Nature Communications 2015, 7(6):7545

- JKnc_neutro: cis- and trans-eQTLs.
- JKnc_neutro_cis: cis-eQTLs only.
- JKnc_neutro_trans: trans-eQTLs only.

5. eQTLs in NK cells. Unpublished

- JK_nk: cis-eQTLs.

6. Tissue-specific eQTLs from GTEx (version 4; incuding 13 tissues). Sourced from Science 2015, 348(6235):648-60

- GTEx_V4_Adipose_Subcutaneous: cis-eQTLs in tissue 'Adipose Subcutaneous'.
- GTEx_V4_Artery_Aorta: cis-eQTLs in tissue 'Artery Aorta'.
- GTEx_V4_Artery_Tibial: cis-eQTLs in tissue 'Artery Tibial'.
- GTEx_V4_Esophagus_Mucosa: cis-eQTLs in tissue 'Esophagus Mucosa'.
- GTEx_V4_Esophagus_Muscularis: cis-eQTLs in tissue 'Esophagus Muscularis'.
- GTEx_V4_Heart_Left_Ventricle: cis-eQTLs in tissue 'Heart Left Ventricle'.

- GTEx_V4_Lung: cis-eQTLs in tissue 'Lung'.
- GTEx_V4_Muscle_Skeletal: cis-eQTLs in tissue 'Muscle Skeletal'.
- GTEx_V4_Nerve_Tibial: cis-eQTLs in tissue 'Nerve Tibial'.
- GTEx_V4_Skin_Sun_Exposed_Lower_leg: cis-eQTLs in tissue 'Skin Sun Exposed Lower leg'.
- GTEx_V4_Stomach: cis-eQTLs in tissue 'Stomach'.
- GTEx_V4_Thyroid: cis-eQTLs in tissue 'Thyroid'.
- GTEx_V4_Whole_Blood: cis-eQTLs in tissue 'Whole Blood'.

Pre-built HiC datasets are described below according to the data sources.
1. Promoter Capture HiC datasets in 17 primary blood cell types. Sourced from Cell 2016, 167(5):1369-1384.e19

- Monocytes: physical interactions (CHiCAGO score >=5) of promoters (baits) with the other end (preys) in Monocytes.
- Macrophages_M0: promoter interactomes in Macrophages M0.
- Macrophages_M1: promoter interactomes in Macrophages M1.
- Macrophages_M2: promoter interactomes in Macrophages M2.
- Neutrophils: promoter interactomes in Neutrophils.
- Megakaryocytes: promoter interactomes in Megakaryocytes.
- Endothelial_precursors: promoter interactomes in Endothelial precursors.
- Fetal_thymus: promoter interactomes in Fetal thymus.
- Naive_CD4_T_cells: promoter interactomes in Naive CD4+ T cells.
- Total_CD4_T_cells: promoter interactomes in Total CD4+ T cells.
- Activated_total_CD4_T_cells: promoter interactomes in Activated total CD4+ T cells.
- Nonactivated_total_CD4_T_cells: promoter interactomes in Nonactivated total CD4+ T cells.
- Naive_CD8_T_cells: promoter interactomes in Naive CD8+ T cells.
- Total_CD8_T_cells: promoter interactomes in Total CD8+ T cells.
- Naive_B_cells: promoter interactomes in Naive B cells.
- Total_B_cells: promoter interactomes in Total B cells.

2. Promoter Capture HiC datasets (involving active promoters and enhancers) in 9 primary blood cell types. Sourced from Cell 2016, 167(5):1369-1384.e19

- PE.Monocytes: physical interactions (CHiCAGO score >=5) of promoters (baits) with the other end (enhancers as preys) in Monocytes.
- PE.Macrophages_M0: promoter-enhancer interactomes in Macrophages M0.
- PE.Macrophages_M1: promoter-enhancer interactomes in Macrophages M1.
- PE.Macrophages_M2: promoter-enhancer interactomes in Macrophages M2.
- PE.Neutrophils: promoter-enhancer interactomes in Neutrophils.
- PE.Megakaryocytes: promoter-enhancer interactomes in Megakaryocytes.
- PE.Erythroblasts: promoter-enhancer interactomes in Erythroblasts.
- PE.Naive_CD4_T_cells: promoter-enhancer interactomes in Naive CD4+ T cells.
- PE.Naive_CD8_T_cells: promoter-enhancer interactomes in Naive CD8+ T cells.

See Also

xPierSNPs, xPierMatrix

Examples

```
## Not run:
# Load the library
library(Pi)

## End(Not run)

RData.location <- "http://galahad.well.ox.ac.uk/bigdata_dev"
# a) provide the SNPs with the significance info
## get lead SNPs reported in AS GWAS and their significance info (p-values)
#data.file <- "http://galahad.well.ox.ac.uk/bigdata/AS.txt"
#AS <- read.delim(data.file, header=TRUE, stringsAsFactors=FALSE)
ImmunoBase <- xRDataLoader(RData.customised='ImmunoBase',
RData.location=RData.location)
gr <- ImmunoBase$AS$variants
AS <- as.data.frame(GenomicRanges::mcols(gr)[, c('Variant','Pvalue')])

## Not run:
# b) perform priority analysis
ls_pNode <- xPierSNPsAdv(data=AS, include.eQTL="JKng_mono",
include.HiC='Monocytes', network="PCommonsUN_medium", restart=0.7,
RData.location=RData.location)

## End(Not run)
```

---

xPierSNPsConsensus *Function to resolve relative importance of distance weight and eQTL weight priorising consensus gene ranks given a list of seed SNPs together with the significance level (e.g. GWAS reported p-values)*

Description

xPierSNPsConsensus is supposed to priorise genes given a list of seed SNPs together with the significance level. It is a parameter-free version of xPierSNPs identifying the consensus rank (less sensitive to the relative importance of the distance weight and eQTL weight). It returns an object of class "pNode" but appended with components on optimal distance weight and consensus info

Usage

```
xPierSNPsConsensus(data, include.LD = NA, LD.customised = NULL,
LD.r2 = 0.8, significance.threshold = 5e-05, distance.max = 2e+05,
decay.kernel = c("rapid", "slow", "linear"), decay.exponent = 2,
GR.SNP = c("dbSNP_GWAS", "dbSNP_Common"), GR.Gene = c("UCSC_knownGene",
"UCSC_knownCanonical"), include.eQTL = c(NA, "JKscience_TS2A",
"JKscience_TS2B", "JKscience_TS3A", "JKng_bcell", "JKng_mono",
"JKnc_neutro",
"JK_nk", "GTEx_V4_Adipose_Subcutaneous", "GTEx_V4_Artery_Aorta",
"GTEx_V4_Artery_Tibial", "GTEx_V4_Esophagus_Mucosa",
"GTEx_V4_Esophagus_Muscularis", "GTEx_V4_Heart_Left_Ventricle",
"GTEx_V4_Lung", "GTEx_V4_Muscle_Skeletal", "GTEx_V4_Nerve_Tibial",
"GTEx_V4_Skin_Sun_Exposed_Lower_leg", "GTEx_V4_Stomach",
"GTEx_V4_Thyroid",
"GTEx_V4_Whole_Blood", "eQTLdb_NK", "eQTLdb_CD14", "eQTLdb_LPS2",
"eQTLdb_LPS24", "eQTLdb_IFN"), eQTL.customised = NULL,
cdf.function = c("empirical", "exponential"), scoring.scheme = c("max",
"sum", "sequential"), network = c("STRING_highest", "STRING_high",
"STRING_medium", "STRING_low", "PCommonsUN_high", "PCommonsUN_medium",
"PCommonsDN_high", "PCommonsDN_medium", "PCommonsDN_Reactome",
"PCommonsDN_KEGG", "PCommonsDN_HumanCyc", "PCommonsDN_PID",
"PCommonsDN_PANTHER", "PCommonsDN_ReconX", "PCommonsDN_TRANSFAC",
"PCommonsDN_PhosphoSite", "PCommonsDN_CTD"), weighted = FALSE,
network.customised = NULL, normalise = c("laplacian", "row", "column",
"none"), restart = 0.75, normalise.affinity.matrix = c("none",
"quantile"), parallel = TRUE, multicores = NULL, verbose = TRUE,
RData.location = "http://galahad.well.ox.ac.uk/bigdata")
```

Arguments

`data` a named input vector containing the sinificance level for nodes (dbSNP). For this named vector, the element names are dbSNP ID (or in the format such as ’chr16:28525386’), the element values for the significance level (measured as p-value or fdr). Alternatively, it can be a matrix or data frame with two columns: 1st column for dbSNP, 2nd column for the significance level

`include.LD` additional SNPs in LD with Lead SNPs are also included. By default, it is ’NA’ to disable this option. Otherwise, LD SNPs will be included based on one or more of 26 populations and 5 super populations from 1000 Genomics Project data (phase 3). The population can be one of 5 super populations ("AFR", "AMR", "EAS", "EUR", "SAS"), or one of 26 populations ("ACB", "ASW", "BEB", "CDX", "CEU", "CHB", "CHS", "CLM", "ESN", "FIN", "GBR", "GIH", "GWD", "IBS", "ITU", "JPT", "KHV", "LWK", "MSL", "MXL", "PEL", "PJL", "PUR", "STU", "TSI", "YRI"). Explanations for population code can be found at http://www.1000genomes.org/faq/which-populations-are-part-your-study

`LD.customised` a user-input matrix or data frame with 3 columns: 1st column for Lead SNPs, 2nd column for LD SNPs, and 3rd for LD r2 value. It is designed to allow the user analysing their pre-calculated LD info. This customisation (if provided) has the high priority over built-in LD SNPs

`LD.r2` the LD r2 value. By default, it is 0.8, meaning that SNPs in LD (r2>=0.8) with input SNPs will be considered as LD SNPs. It can be any value from 0.8 to 1

`significance.threshold`
the given significance threshold. By default, it is set to NULL, meaning there is no constraint on the significance level when transforming the significance level of SNPs into scores. If given, those SNPs below this are considered significant and thus scored positively. Instead, those above this are considered insigificant and thus receive no score

`distance.max` the maximum distance between genes and SNPs. Only those genes no far way from this distance will be considered as seed genes. This parameter will influence the distance-component weights calculated for nearby SNPs per gene

`decay.kernel` a character specifying a decay kernel function. It can be one of ’slow’ for slow decay, ’linear’ for linear decay, and ’rapid’ for rapid decay

`decay.exponent` an integer specifying a decay exponent. By default, it sets to 2

`GR.SNP` the genomic regions of SNPs. By default, it is 'dbSNP_GWAS', that is, SNPs from dbSNP (version 146) restricted to GWAS SNPs and their LD SNPs (hg19). It can be 'dbSNP_Common', that is, Common SNPs from dbSNP (version 146) plus GWAS SNPs and their LD SNPs (hg19). Alternatively, the user can specify the customised input. To do so, first save your RData file (containing an GR object) into your local computer, and make sure the GR object content names refer to dbSNP IDs. Then, tell "GR.SNP" with your RData file name (with or without extension), plus specify your file RData path in "RData.location"

`GR.Gene` the genomic regions of genes. By default, it is 'UCSC_knownGene', that is, UCSC known genes (together with genomic locations) based on human genome assembly hg19. It can be 'UCSC_knownCanonical', that is, UCSC known canonical genes (together with genomic locations) based on human genome assembly hg19. Alternatively, the user can specify the customised input. To do so, first save your RData file (containing an GR object) into your local computer, and make sure the GR object content names refer to Gene Symbols. Then, tell "GR.Gene" with your RData file name (with or without extension), plus specify your file RData path in "RData.location"

`include.eQTL` genes modulated by eQTL (also Lead SNPs or in LD with Lead SNPs) are also included. By default, it is 'NA' to disable this option. Otherwise, those genes modulated by eQTL will be included: immune stimulation in monocytes ('JKscience_TS1A' and 'JKscience_TS2B' for cis-eQTLs or 'JKscience_TS3A' for trans-eQTLs) from Science 2014, 343(6175):1246949; cis- and trans-eQTLs in B cells ('JKng_bcell') and in monocytes ('JKng_mono') from Nature Genetics 2012, 44(5):502-510; cis- and trans-eQTLs in neutrophils ('JKnc_neutro') from Nature Communications 2015, 7(6):7545; cis-eQTLs in NK cells ('JK_nk') which is unpublished. Also supported are GTEx cis-eQTLs from Science 2015, 348(6235):648-60, including 13 tissues: 'GTEx_Adipose_Subcutaneous','GTEx_Artery_Aorta','GT

`eQTL.customised`
a user-input matrix or data frame with 3 columns: 1st column for SNPs/eQTLs, 2nd column for Genes, and 3rd for eQTL mapping significance level (p-values or FDR). It is designed to allow the user analysing their eQTL data. This customisation (if provided) has the high priority over built-in eQTL data.

`cdf.function` a character specifying a Cumulative Distribution Function (cdf). It can be one of 'exponential' based on exponential cdf, 'empirical' for empirical cdf

`scoring.scheme` the method used to calculate seed gene scores under a set of SNPs. It can be one of "sum" for adding up, "max" for the maximum, and "sequential" for the sequential weighting. The sequential weighting is done via: $\sum_{i=1} \frac{R_i}{i}$, where R_i is the i^{th} rank (in a descreasing order)

`network` the built-in network. Currently two sources of network information are supported: the STRING database (version 10) and the Pathways Commons database (version 7). STRING is a meta-integration of undirect interactions from the functional aspect, while Pathways Commons mainly contains both undirect and direct interactions from the physical/pathway aspect. Both have scores to control the confidence of interactions. Therefore, the user can choose the different quality of the interactions. In STRING, "STRING_highest" indicates interactions with highest confidence (confidence scores>=900), "STRING_high" for interactions with high confidence (confidence scores>=700), "STRING_medium" for interactions with medium confidence (confidence scores>=400), and "STRING_low" for interactions with low confidence (confidence scores>=150). For undirect/physical interactions from Pathways Commons, "PCommonsUN_high" indicates undirect interactions with high confidence (supported with the PubMed references plus at least 2 different sources), "PCommonsUN_medium" for undirect interactions with medium confidence (supported with the PubMed references). For direct (pathway-merged) interactions from Pathways Commons, "PCommonsDN_high" indicates direct interactions with high confidence (supported with the PubMed references plus at least 2 different sources), and "PCommonsUN_medium" for direct interactions with medium confidence (supported with the PubMed references). In addtion to pooled version of pathways from all data sources, the user can also choose the pathway-merged network from individual sources, that is, "PCommonsDN_Reactome" for those from Reactome, "PCommonsDN_KEGG" for those from KEGG, "PCommonsDN_HumanCyc" for those from HumanCyc, "PCommonsDN_PID" for those froom PID, "PCommonsDN_PANTHER" for those from PANTHER, "PCommonsDN_ReconX" for those from ReconX, "PCommonsDN_TRANSFAC" for those from TRANSFAC, "PCommonsDN_PhosphoSite" for those from PhosphoSite, and "PCommonsDN_CTD" for those from CTD

`weighted` logical to indicate whether edge weights should be considered. By default, it sets to false. If true, it only works for the network from the STRING database

`network.customised` an object of class "igraph". By default, it is NULL. It is designed to allow the user analysing their customised network data that are not listed in the above argument 'network'. This customisation (if provided) has the high priority over built-in network. If the user provides the "igraph" object with the "weight" edge attribute, RWR will assume to walk on the weighted network

`normalise` the way to normalise the adjacency matrix of the input graph. It can be 'laplacian' for laplacian normalisation, 'row' for row-wise normalisation, 'column' for column-wise normalisation, or 'none'

`restart` the restart probability used for Random Walk with Restart (RWR). The restart probability takes the value from 0 to 1, controlling the range from the starting nodes/seeds that the walker will explore. The higher the value, the more likely the walker is to visit the nodes centered on the starting nodes. At the extreme when the restart probability is zero, the walker moves freely to the neighbors at each step without restarting from seeds, i.e., following a random walk (RW)

`normalise.affinity.matrix` the way to normalise the output affinity matrix. It can be 'none' for no normalisation, 'quantile' for quantile normalisation to ensure that columns (if multiple) of the output affinity matrix have the same quantiles

`parallel` logical to indicate whether parallel computation with multicores is used. By default, it sets to true, but not necessarily does so. Partly because parallel backends available will be system-specific (now only Linux or Mac OS). Also, it will depend on whether these two packages "foreach" and "doMC" have been installed. It can be installed via: `source("http://bioconductor.org/biocLite.R"); biocLite(c("foreach","doMC"))`. If not yet installed, this option will be disabled

`multicores` an integer to specify how many cores will be registered as the multicore parallel backend to the 'foreach' package. If NULL, it will use a half of cores available in a user's computer. This option only works when parallel computation is enabled

`verbose` logical to indicate whether the messages will be displayed in the screen. By default, it sets to true for display

`RData.location` the characters to tell the location of built-in RData files. See `xRDataLoader` for details

Value

an object of class "pNode", a list with following components:

- `priority`: a matrix of nNode X 4 containing node priority information, where nNode is the number of nodes in the input graph, and the 4 columns are "name" (node names), "seed" (1 for seeds, 0 for non-seeds), "weight" (weight/score values for seed genes), "priority" (the priority scores that are rescaled to the range [0,1]), "rank" (ranks of the priority scores), and two additional columns: 'driver' telling who drives the prioritisation ('nGenes','eGenes' or'both'), and 'consensus_rank'
- `g`: an input "igraph" object
- `SNP`: a data frame of nSNP X 3 containing input SNPs and/or LD SNPs info, where nSNP is the number of input SNPs and/or LD SNPs, and the 3 columns are "SNP" (dbSNP), "Score" (the SNP score), "Pval" (the SNP p-value)
- `Gene2SNP`: a matrix of Genes X SNPs, each non-zero cell telling an SNP's genetic influential score on a seed gene
- `nGenes`: the relative weight for nearby genes
- `consensus`: a matrix containing details on rank results by decreasing the relative importance of nGenes. In addition to rank matrix, it has columns 'rank_median' for median rank excluding two extremes 'n_1' (nGenes only) and 'n_0' (eGenes only), 'rank_MAD' for median absolute deviation, 'driver' telling who drives the prioritisation ('nGenes','eGenes' or'both'), 'consensus_rank' for the rank of the median rank list
- `call`: the call that produced this result

Note

none

See Also

`xPierSNPs`

Examples

```
## Not run:
# Load the library
library(Pi)

## End(Not run)

# a) provide the SNPs with the significance info
## get lead SNPs reported in AS GWAS and their significance info (p-values)
#data.file <- "http://galahad.well.ox.ac.uk/bigdata/AS.txt"
#AS <- read.delim(data.file, header=TRUE, stringsAsFactors=FALSE)
ImmunoBase <- xRDataLoader(RData.customised='ImmunoBase')
gr <- ImmunoBase$AS$variants
AS <- as.data.frame(GenomicRanges::mcols(gr)[, c('Variant','Pvalue')])

## Not run:
# b) perform priority analysis
pNode <- xPierSNPsConsensus(data=AS, include.LD="EUR",
include.eQTL=c("JKscience_TS2A","JKscience_TS3A"),
network="PCommonsUN_medium", restart=0.7)
```

```
# c) save to the file called 'SNPs_priority.consensus.txt'
write.table(pNode$priority, file="SNPs_priority.consensus.txt",
sep="\t", row.names=FALSE)

# d) manhattan plot
mp <- xPierManhattan(pNode, highlight.top=10)
#pdf(file="Gene_manhattan.pdf", height=6, width=12, compress=TRUE)
print(mp)
#dev.off()

## End(Not run)
```

xPierSubnet *Function to identify a gene network from top prioritised genes*

Description

xPierSubnet is supposed to identify maximum-scoring gene subnetwork from a graph with the node information on priority scores, both are part of an object of class "pNode". It returns an object of class "igraph".

Usage

```
xPierSubnet(pNode, priority.quantite = 0.1, network = c(NA,
"STRING_highest", "STRING_high", "STRING_medium", "STRING_low",
"PCommonsUN_high", "PCommonsUN_medium", "PCommonsDN_high",
"PCommonsDN_medium", "PCommonsDN_Reactome", "PCommonsDN_KEGG",
"PCommonsDN_HumanCyc", "PCommonsDN_PID", "PCommonsDN_PANTHER",
"PCommonsDN_ReconX", "PCommonsDN_TRANSFAC", "PCommonsDN_PhosphoSite",
"PCommonsDN_CTD"), network.customised = NULL, subnet.significance =
0.01,
subnet.size = NULL, verbose = TRUE,
RData.location = "http://galahad.well.ox.ac.uk/bigdata")
```

Arguments

pNode an object of class "pNode" (or "pTarget" or "dTarget")

priority.quantite
the quantite of the top priority genes. By default, 10 analysis. If NULL or NA, all prioritised genes will be used

network the built-in network. If NA, the network used for prioritisation will be used, which is part of the object of class "pNode". Otherwise, choose the other network of interest. Currently two sources of network information are supported: the STRING database (version 10) and the Pathways Commons database (version 7). STRING is a meta-integration of undirect interactions from the functional aspect, while Pathways Commons mainly contains both undirect and direct interactions from the physical/pathway aspect. Both have scores to control the confidence of interactions. Therefore, the user can choose the different quality of the interactions. In STRING, "STRING_highest" indicates interactions with highest confidence (confidence scores>=900), "STRING_high" for interactions with high confidence (confidence scores>=700), "STRING_medium" for interactions with medium confidence (confidence scores>=400), and "STRING_low" for interactions with low confidence (confidence scores>=150). For undirect/physical interactions from Pathways Commons, "PCommonsUN_high" indicates undirect interactions with high confidence (supported with the PubMed references plus at least 2 different sources), "PCommonsUN_medium" for undirect interactions with medium confidence (supported with the PubMed references). For direct (pathway-merged) interactions from Pathways Commons, "PCommonsDN_high" indicates direct interactions with high confidence (supported with the PubMed references plus at least 2 different sources), and "PCommonsUN_medium" for direct interactions with medium confidence (supported with the PubMed references). In addtion to pooled version of pathways from all data sources, the user can also choose the pathway-merged network from individual sources, that is, "PCommonsDN_Reactome" for those from Reactome, "PCommonsDN_KEGG" for those from KEGG, "PCommonsDN_HumanCyc" for those from HumanCyc, "PCommonsDN_PID" for those froom PID, "PCommonsDN_PANTHER" for those from PANTHER, "PCommonsDN_ReconX" for those from ReconX, "PCommonsDN_TRANSFAC" for those from TRANSFAC, "PCommonsDN_PhosphoSite" for those from PhosphoSite, and "PCommonsDN_CTD" for those from CTD

`network.customised`
: an object of class "igraph". By default, it is NULL. It is designed to allow the user analysing their customised network data that are not listed in the above argument 'network'. This customisation (if provided) has the high priority over built-in network

`subnet.significance`
: the given significance threshold. By default, it is set to NULL, meaning there is no constraint on nodes/genes. If given, those nodes/genes with p-values below this are considered significant and thus scored positively. Instead, those p-values above this given significance threshold are considered insigificant and thus scored negatively

`subnet.size`
: the desired number of nodes constrained to the resulting subnet. It is not nulll, a wide range of significance thresholds will be scanned to find the optimal significance threshold leading to the desired number of nodes in the resulting subnet. Notably, the given significance threshold will be overwritten by this option

`verbose`
: logical to indicate whether the messages will be displayed in the screen. By default, it sets to true for display

`RData.location`
: the characters to tell the location of built-in RData files. See `xRDataLoader` for details

Value

a subgraph with a maximum score, an object of class "igraph". It has ndoe attributes: signficance, score, priority (part of the "pNode" object)

Note

The priority score will be first scaled to the range x=[0 100] and then is converted to pvalue-like significant level: 10^(-x). Next, `xSubneterGenes` is used to identify a maximum-scoring gene subnetwork that contains as many highly prioritised genes as possible but a few lowly prioritised genes as linkers. An iterative procedure of scanning different priority thresholds is also used to identify the network with a desired number of nodes/genes. Notably, the preferential use of the same network as used in gene-level prioritisation is due to the fact that gene-level affinity/priority scores are smoothly distributed over the network after being walked. In other words, the chance of identifying such a gene network enriched with top prioritised genes is much higher.

See Also

xSubneterGenes

Examples

```
## Not run:
# Load the library
library(Pi)

## End(Not run)

RData.location <- "http://galahad.well.ox.ac.uk/bigdata_dev"
# a) provide the SNPs with the significance info
## get lead SNPs reported in AS GWAS and their significance info (p-values)
#data.file <- "http://galahad.well.ox.ac.uk/bigdata/AS.txt"
#AS <- read.delim(data.file, header=TRUE, stringsAsFactors=FALSE)
ImmunoBase <- xRDataLoader(RData.customised='ImmunoBase',
RData.location=RData.location)
gr <- ImmunoBase$AS$variants
AS <- as.data.frame(GenomicRanges::mcols(gr)[, c('Variant','Pvalue')])

## Not run:
# b) perform priority analysis
pNode <- xPierSNPs(data=AS, include.eQTL="JKng_mono",
include.HiC='Monocytes', network="PCommonsUN_medium", restart=0.7,
RData.location=RData.location)

# c) perform network analysis
# find maximum-scoring subnet with the desired node number=50
subnet <- xPierSubnet(pNode, priority.quantite=0.1, subnet.size=50,
RData.location=RData.location)

# d) save subnet results to the files called 'subnet_edges.txt' and 'subnet_nodes.txt'
output <- igraph::get.data.frame(subnet, what="edges")
utils::write.table(output, file="subnet_edges.txt", sep="\t",
row.names=FALSE)
output <- igraph::get.data.frame(subnet, what="vertices")
utils::write.table(output, file="subnet_nodes.txt", sep="\t",
row.names=FALSE)

# e) visualise the identified subnet
## do visualisation with nodes colored according to the priority
xVisNet(g=subnet, pattern=V(subnet)$priority, vertex.shape="sphere")
## do visualisation with nodes colored according to pvalue-like signficance
xVisNet(g=subnet, pattern=-log10(as.numeric(V(subnet)$significance)),
vertex.shape="sphere", colormap="wyr")

# f) visualise the identified subnet as a circos plot
library(RCircos)
xCircos(g=subnet, entity="Gene", RData.location=RData.location)

## End(Not run)
```

xPierTrack *Function to visualise prioritised genes using track plot*

Description

xPierTrack is supposed to visualise prioritised genes using track plot. Priority for the gene in query is displayed on the data track and nearby genes on the annotation track. Genomic locations on the X-axis are indicated on the X-axis, and the gene in query is highlighted.

Usage

```
xPierTrack(pNode, priority.top = NULL, target.query = NULL,
window = 1e+06, nearby = NULL, query.highlight = TRUE,
GR.Gene = c("UCSC_knownGene", "UCSC_knownCanonical"), verbose = TRUE,
RData.location = "http://galahad.well.ox.ac.uk/bigdata")
```

Arguments

pNode	an object of class "pNode" (or "pTarget" or "dTarget")
priority.top	the number of the top targets used for track plot. By default, it is NULL meaning all targets are used
target.query	which gene in query will be visualised. If NULL, the target gene with the top priority will be displayed
window	the maximum distance defining nearby genes around the target gene in query. By default it is 1e6
nearby	the maximum number defining nearby genes around the target gene in query. By default it is NULL. If not NULL, it will overwrite the parameter 'window'
query.highlight	logical to indicate whether the gene in query will be highlighted
GR.Gene	the genomic regions of genes. By default, it is 'UCSC_knownGene', that is, UCSC known genes (together with genomic locations) based on human genome assembly hg19. It can be 'UCSC_knownCanonical', that is, UCSC known canonical genes (together with genomic locations) based on human genome assembly hg19. Alternatively, the user can specify the customised input. To do so, first save your RData file (containing an GR object) into your local computer, and make sure the GR object content names refer to Gene Symbols. Then, tell "GR.Gene" with your RData file name (with or without extension), plus specify your file RData path in "RData.location"
verbose	logical to indicate whether the messages will be displayed in the screen. By default, it sets to false for no display
RData.location	the characters to tell the location of built-in RData files. See xRDataLoader for details

Value

an object of class "ggplot", appended by an GR object called 'gr'

Note

none

See Also

xMLrandomforest

Examples

```
## Not run:
# Load the library
library(Pi)

## End(Not run)

RData.location <- "http://galahad.well.ox.ac.uk/bigdata_dev"
# a) provide the SNPs with the significance info
## get lead SNPs reported in AS GWAS and their significance info (p-values)
#data.file <- "http://galahad.well.ox.ac.uk/bigdata/AS.txt"
#AS <- read.delim(data.file, header=TRUE, stringsAsFactors=FALSE)
ImmunoBase <- xRDataLoader(RData.customised='ImmunoBase',
RData.location=RData.location)
gr <- ImmunoBase$AS$variants
AS <- as.data.frame(GenomicRanges::mcols(gr)[, c('Variant','Pvalue')])

## Not run:
# b) perform priority analysis
pNode <- xPierSNPs(data=AS, include.eQTL="JKng_mono",
include.HiC='Monocytes', network="PCommonsUN_medium", restart=0.7,
RData.location=RData.location)

# c) track plot
library(Gviz)
#pdf(file="Gene_tracks.pdf", height=4, width=10, compress=TRUE)
xPierTrack(pNode, RData.location=RData.location)
#dev.off()
xPierTrack(pNode, priority.top=1000, nearby=20,
RData.location=RData.location)

## End(Not run)
```

xPredictCompare *Function to compare prediction performance results*

Description

xPredictCompare is supposed to compare prediction performance results. It returns an object of class "ggplot".

Usage

```
xPredictCompare(list_pPerf, displayBy = c("ROC", "PR"), type =
c("curve",
"bar"), sort = TRUE, detail = TRUE, facet = FALSE, signature = TRUE)
```

Arguments

`list_pPerf` a list of "pPerf" objects

`displayBy` which performance will be used for comparison. It can be "ROC" for ROC curve (by default), "PR" for PR curve

`type` the type of plot to draw. It can be "curve" for curve plot (by default), "bar" for bar plot

`sort` logical to indicate whether to sort methods according to performance. By default, it sets TRUE

`detail` logical to indicate whether to label methods along with performance. By default, it sets TRUE

`facet` logical to indicate whether to facet/wrap a 1d of panels into 2d. By default, it sets FALSE

`signature` a logical to indicate whether the signature is assigned to the plot caption. By default, it sets TRUE showing which function is used to draw this graph

Value

an object of class "ggplot" or NULL (if all input pPerf objects are NULL)

Note

none

See Also

xPredictROCR

Examples

```
# Load the library
## Not run:
# Load the library
library(Pi)

## End(Not run)
RData.location <- "http://galahad.well.ox.ac.uk/bigdata_dev"
## Not run:
bp <- xPredictCompare(ls_pPerf, displayBy="ROC")
print(bp)
## modify legend position
bp + theme(legend.position=c(0.75,0.25))

## End(Not run)
```

xPredictROCR *Function to assess the prediction performance via ROC and Precision-Recall (PR) analysis*

Description

`xPredictROCR` is supposed to assess the prediction performance via Receiver Operating Characteristic (ROC) and Precision-Recall (PR) analysis. It requires three inputs: 1) Gold Standard Positive (GSP) targets; 2) Gold Standard Negative (GSN) targets; 3) prediction containing predicted targets and predictive scores.

Usage

```
xPredictROCR(prediction, GSP, GSN, rescale = TRUE, plot = c("none",
"ROC",
"PR"), verbose = TRUE, signature = TRUE)
```

Arguments

`prediction`	a data frame containing predictions along with predictive scores. It has two columns: 1st column for target, 2nd column for predictive scores (the higher the better). Alternatively, it can be an object of class "pNode" (or "pTarget" or "dTarget") from which a data frame is extracted
`GSP`	a vector containing Gold Standard Positives (GSP)
`GSN`	a vector containing Gold Standard Negatives (GSN)
`rescale`	logical to indicate whether to linearly rescale predictive scores for GSP/GSN targets to the range [0,1]. By default, it sets to TRUE
`plot`	the way to plot performance curve. It can be 'none' for no curve returned, 'ROC' for ROC curve, and 'PR' for PR curve.
`verbose`	logical to indicate whether the messages will be displayed in the screen. By default, it sets to TRUE for display
`signature`	a logical to indicate whether the signature is assigned to the plot caption. By default, it sets TRUE showing which function is used to draw this graph

Value

If plot is 'none' (by default), an object of class "pPerf", a list with following components:

- `PRS`: a data frame with 3 columns ('Precision', 'Recall' and 'Specificity')
- `AUROC`: a scalar value for ROC AUC
- `Fmax`: a scalar value for maximum F-measure
- `ROC_perf`: a ROCR performance-class object for ROC curve
- `PR_perf`: a ROCR performance-class object for PR curve
- `Pred_obj`: a ROCR prediction-class object (potentially used for calculating other performance measures)
- `call`: the call that produced this result

If plot is 'ROC' or 'PR', it will return a ggplot object after being appended with the same components as mentioned above. If no GSP and/or GSN is predicted, it will return NULL

Note

AUC: the area under ROC F-measure: the maximum of a harmonic mean between precision and recall along PR curve

Examples

```
## Not run:
# Load the library
library(Pi)

## End(Not run)
RData.location <- "http://galahad.well.ox.ac.uk/bigdata_dev"
## Not run:
pPerf <- xPredictROCR(prediction, GSP, GSN)

## End(Not run)
```

---

xRWR   *Function to implement Random Walk with Restart (RWR) on the input graph*

Description

xRWR is supposed to implement Random Walk with Restart (RWR) on the input graph. If the seeds (i.e. a set of starting nodes) are given, it intends to calculate the affinity score of all nodes in the graph to the seeds. If the seeds are not given, it will pre-compute affinity matrix for nodes in the input graph with respect to each starting node (as a seed) by looping over every node in the graph. Parallel computing is also supported.

Usage

```
xRWR(g, normalise = c("laplacian", "row", "column", "none"),
setSeeds = NULL, restart = 0.75, normalise.affinity.matrix = c("none",
"quantile"), parallel = TRUE, multicores = NULL, verbose = TRUE)
```

Arguments

g	an object of class "igraph" or "graphNEL"
normalise	the way to normalise the adjacency matrix of the input graph. It can be 'laplacian' for laplacian normalisation, 'row' for row-wise normalisation, 'column' for column-wise normalisation, or 'none'
setSeeds	an input matrix used to define sets of starting seeds. One column corresponds to one set of seeds that a walker starts with. The input matrix must have row names, coming from node names of input graph, i.e. V(g)$name, since there is a mapping operation. The non-zero entries mean that the corresonding rows (i.e. the gene/row names) are used as the seeds, and non-zero values can be viewed as how to weight the relative importance of seeds. By default, this option sets to "NULL", suggesting each node in the graph will be used as a set of the seed to pre-compute affinity matrix for the input graph. This default does not scale for large input graphs since it will loop over every node in the graph; however, the pre-computed affinity matrix can be extensively reused for obtaining affinity

scores between any combinations of nodes/seeds, allows for some flexibility in the downstream use, in particular when sampling a large number of random node combinations for statistical testing

`restart` the restart probability used for RWR. The restart probability takes the value from 0 to 1, controlling the range from the starting nodes/seeds that the walker will explore. The higher the value, the more likely the walker is to visit the nodes centered on the starting nodes. At the extreme when the restart probability is zero, the walker moves freely to the neighbors at each step without restarting from seeds, i.e., following a random walk (RW)

`normalise.affinity.matrix`
the way to normalise the output affinity matrix. It can be 'none' for no normalisation, 'quantile' for quantile normalisation to ensure that columns (if multiple) of the output affinity matrix have the same quantiles

`parallel` logical to indicate whether parallel computation with multicores is used. By default, it sets to true, but not necessarily does so. It will depend on whether these two packages "foreach" and "doParallel" have been installed. It can be installed via: `source("http://bioconductor.org/biocLite.R"); biocLite(c("foreach","doParalle`
If not yet installed, this option will be disabled

`multicores` an integer to specify how many cores will be registered as the multicore parallel backend to the 'foreach' package. If NULL, it will use a half of cores available in a user's computer. This option only works when parallel computation is enabled

`verbose` logical to indicate whether the messages will be displayed in the screen. By default, it sets to true for display

Value

It returns a sparse matrix, called 'PTmatrix':

- When the seeds are NOT given: a pre-computated affinity matrix with the dimension of n X n, where n is the number of nodes in the input graph. Columns stand for starting nodes walking from, and rows for ending nodes walking to. Therefore, a column for a starting node represents a steady-state affinity vector that the starting node will visit all the ending nodes in the graph
- When the seeds are given: an affinity matrix with the dimension of n X nset, where n is the number of nodes in the input graph, and nset for the number of the sets of seeds (i.e. the number of columns in setSeeds). Each column stands for the steady probability vector, storing the affinity score of all nodes in the graph to the starting nodes/seeds. This steady probability vector can be viewed as the "influential impact" over the graph imposed by the starting nodes/seeds.

Note

The input graph will treat as an unweighted graph if there is no 'weight' edge attribute associated with

See Also

`xPier`

Examples

```
# 1) generate a random graph according to the ER model
set.seed(123)
g <- erdos.renyi.game(10, 1/10)

## Not run:
# 2) produce the induced subgraph only based on the nodes in query
subg <- dNetInduce(g, V(g), knn=0)
V(subg)$name <- 1:vcount(subg)

# 3) obtain the pre-computated affinity matrix
PTmatrix <- xRWR(g=subg, normalise="laplacian", restart=0.75,
parallel=FALSE)
# visualise affinity matrix
visHeatmapAdv(as.matrix(PTmatrix), Rowv=FALSE, Colv=FALSE,
colormap="wyr", KeyValueName="Affinity")

# 4) obtain affinity matrix given sets of seeds
# define sets of seeds
# each seed with equal weight (i.e. all non-zero entries are '1')
aSeeds <- c(1,0,1,0,1)
bSeeds <- c(0,0,1,0,1)
setSeeds <- data.frame(aSeeds,bSeeds)
rownames(setSeeds) <- 1:5
# calcualte affinity matrix
PTmatrix <- xRWR(g=subg, normalise="laplacian", setSeeds=setSeeds,
restart=0.75, parallel=FALSE)
PTmatrix

## End(Not run)
```

---

xSNP2cGenes *Function to define HiC genes given a list of SNPs*

Description

xSNP2cGenes is supposed to define HiC genes given a list of SNPs. The HiC weight is calcualted as Cumulative Distribution Function of HiC interaction scores.

Usage

```
xSNP2cGenes(data, entity = c("SNP", "chr:start-end", "data.frame",
"bed",
"GRanges"), include.HiC = c(NA, "Monocytes", "Macrophages_M0",
"Macrophages_M1", "Macrophages_M2", "Neutrophils", "Megakaryocytes",
"Endothelial_precursors", "Erythroblasts", "Fetal_thymus",
"Naive_CD4_T_cells", "Total_CD4_T_cells",
"Activated_total_CD4_T_cells",
"Nonactivated_total_CD4_T_cells", "Naive_CD8_T_cells",
"Total_CD8_T_cells",
"Naive_B_cells", "Total_B_cells", "PE.Monocytes", "PE.Macrophages_M0",
"PE.Macrophages_M1", "PE.Macrophages_M2", "PE.Neutrophils",
"PE.Megakaryocytes", "PE.Erythroblasts", "PE.Naive_CD4_T_cells",
"PE.Naive_CD8_T_cells"), GR.SNP = c("dbSNP_GWAS", "dbSNP_Common"),
cdf.function = c("empirical", "exponential"), plot = FALSE,
verbose = TRUE, RData.location =
"http://galahad.well.ox.ac.uk/bigdata")
```

Arguments

`data` NULL or a input vector containing SNPs. If NULL, all SNPs will be considered. If a input vector containing SNPs, SNPs should be provided as dbSNP ID (ie starting with rs) or in the format of 'chrN:xxx', where N is either 1-22 or X, xxx is number; for example, 'chr16:28525386'. Alternatively, it can be other formats/entities (see the next parameter 'entity')

`entity` the data entity. By default, it is "SNP". For general use, it can also be one of "chr:start-end", "data.frame", "bed" or "GRanges"

`include.HiC` genes linked to input SNPs are also included. By default, it is 'NA' to disable this option. Otherwise, those genes linked to SNPs will be included according to Promoter Capture HiC (PCHiC) datasets. Pre-built HiC datasets are detailed in the section 'Note'

`GR.SNP` the genomic regions of SNPs. By default, it is 'dbSNP_GWAS', that is, SNPs from dbSNP (version 146) restricted to GWAS SNPs and their LD SNPs (hg19). It can be 'dbSNP_Common', that is, Common SNPs from dbSNP (version 146) plus GWAS SNPs and their LD SNPs (hg19). Alternatively, the user can specify the customised input. To do so, first save your RData file (containing an GR object) into your local computer, and make sure the GR object content names refer to dbSNP IDs. Then, tell "GR.SNP" with your RData file name (with or without extension), plus specify your file RData path in "RData.location". Note: you can also load your customised GR object directly

`cdf.function` a character specifying a Cumulative Distribution Function (cdf). It can be one of 'exponential' based on exponential cdf, 'empirical' for empirical cdf

`plot` logical to indicate whether the histogram plot (plus density or CDF plot) should be drawn. By default, it sets to false for no plotting

`verbose` logical to indicate whether the messages will be displayed in the screen. By default, it sets to true for display

`RData.location` the characters to tell the location of built-in RData files. See `xRDataLoader` for details

Value

a data frame with following columns:

- `Gene`: SNP-interacting genes caputured by HiC
- `SNP`: SNPs
- `Sig`: the interaction score (the higher stronger)
- `Weight`: the HiC weight

Note

Pre-built HiC datasets are described below according to the data sources.
1. Promoter Capture HiC datasets in 17 primary blood cell types. Sourced from Cell 2016, 167(5):1369-1384.e19

- Monocytes: physical interactions (CHiCAGO score >=5) of promoters (baits) with the other end (preys) in Monocytes.
- Macrophages_M0: promoter interactomes in Macrophages M0.
- Macrophages_M1: promoter interactomes in Macrophages M1.
- Macrophages_M2: promoter interactomes in Macrophages M2.
- Neutrophils: promoter interactomes in Neutrophils.
- Megakaryocytes: promoter interactomes in Megakaryocytes.
- Endothelial_precursors: promoter interactomes in Endothelial precursors.
- Fetal_thymus: promoter interactomes in Fetal thymus.
- Naive_CD4_T_cells: promoter interactomes in Naive CD4+ T cells.
- Total_CD4_T_cells: promoter interactomes in Total CD4+ T cells.
- Activated_total_CD4_T_cells: promoter interactomes in Activated total CD4+ T cells.
- Nonactivated_total_CD4_T_cells: promoter interactomes in Nonactivated total CD4+ T cells.
- Naive_CD8_T_cells: promoter interactomes in Naive CD8+ T cells.
- Total_CD8_T_cells: promoter interactomes in Total CD8+ T cells.
- Naive_B_cells: promoter interactomes in Naive B cells.
- Total_B_cells: promoter interactomes in Total B cells.

2. Promoter Capture HiC datasets (involving active promoters and enhancers) in 9 primary blood cell types. Sourced from Cell 2016, 167(5):1369-1384.e19

- PE.Monocytes: physical interactions (CHiCAGO score >=5) of promoters (baits) with the other end (enhancers as preys) in Monocytes.
- PE.Macrophages_M0: promoter-enhancer interactomes in Macrophages M0.
- PE.Macrophages_M1: promoter-enhancer interactomes in Macrophages M1.
- PE.Macrophages_M2: promoter-enhancer interactomes in Macrophages M2.
- PE.Neutrophils: promoter-enhancer interactomes in Neutrophils.
- PE.Megakaryocytes: promoter-enhancer interactomes in Megakaryocytes.
- PE.Erythroblasts: promoter-enhancer interactomes in Erythroblasts.
- PE.Naive_CD4_T_cells: promoter-enhancer interactomes in Naive CD4+ T cells.
- PE.Naive_CD8_T_cells: promoter-enhancer interactomes in Naive CD8+ T cells.

See Also

xSNPhic

Examples

```
## Not run:
# Load the library
library(Pi)

## End(Not run)

RData.location <- "http://galahad.well.ox.ac.uk/bigdata_dev"
# a) provide the SNPs with the significance info
```

```
## get lead SNPs reported in AS GWAS and their significance info (p-values)
#data.file <- "http://galahad.well.ox.ac.uk/bigdata/AS.txt"
#AS <- read.delim(data.file, header=TRUE, stringsAsFactors=FALSE)
ImmunoBase <- xRDataLoader(RData.customised='ImmunoBase',
RData.location=RData.location)
data <- names(ImmunoBase$AS$variants)

## Not run:
# b) define HiC genes
df_cGenes <- xSNP2cGenes(data, include.HiC="Monocytes",
RData.location=RData.location)

## End(Not run)
```

xSNP2eGenes *Function to define eQTL genes given a list of SNPs or a customised eQTL mapping data*

Description

xSNP2eGenes is supposed to define eQTL genes given a list of SNPs or a customised eQTL mapping data. The eQTL weight is calcualted as Cumulative Distribution Function of negative log-transformed eQTL-reported signficance level.

Usage

```
xSNP2eGenes(data, include.eQTL = c(NA, "JKscience_TS2A",
"JKscience_TS2A_CD14", "JKscience_TS2A_LPS2", "JKscience_TS2A_LPS24",
"JKscience_TS2A_IFN", "JKscience_TS2B", "JKscience_TS2B_CD14",
"JKscience_TS2B_LPS2", "JKscience_TS2B_LPS24", "JKscience_TS2B_IFN",
"JKscience_TS3A", "JKng_bcell", "JKng_bcell_cis", "JKng_bcell_trans",
"JKng_mono", "JKng_mono_cis", "JKng_mono_trans", "JKnc_neutro",
"JKnc_neutro_cis", "JKnc_neutro_trans", "JK_nk",
"GTEx_V4_Adipose_Subcutaneous", "GTEx_V4_Artery_Aorta",
"GTEx_V4_Artery_Tibial", "GTEx_V4_Esophagus_Mucosa",
"GTEx_V4_Esophagus_Muscularis", "GTEx_V4_Heart_Left_Ventricle",
"GTEx_V4_Lung", "GTEx_V4_Muscle_Skeletal", "GTEx_V4_Nerve_Tibial",
"GTEx_V4_Skin_Sun_Exposed_Lower_leg", "GTEx_V4_Stomach",
"GTEx_V4_Thyroid",
"GTEx_V4_Whole_Blood", "eQTLdb_NK", "eQTLdb_CD14", "eQTLdb_LPS2",
"eQTLdb_LPS24", "eQTLdb_IFN"), eQTL.customised = NULL,
cdf.function = c("empirical", "exponential"), plot = FALSE,
verbose = TRUE, RData.location =
"http://galahad.well.ox.ac.uk/bigdata")
```

Arguments

data a input vector containing SNPs. SNPs should be provided as dbSNP ID (ie starting with rs). Alternatively, they can be in the format of 'chrN:xxx', where N is either 1-22 or X, xxx is number; for example, 'chr16:28525386' `include.eQTL` genes modulated by eQTL (also Lead SNPs or in LD with Lead SNPs) are also included. By default, it is 'NA' to disable this option. Otherwise, those genes modulated by eQTL will be included. Pre-built eQTL datasets are detailed in the section 'Note'

`eQTL.customised` a user-input matrix or data frame with 3 columns: 1st column for SNPs/eQTLs, 2nd column for Genes, and 3rd for eQTL mapping significance level (p-values or FDR). It is designed to allow the user analysing their eQTL data. This customisation (if provided) has the high priority over built-in eQTL data.

`cdf.function` a character specifying a Cumulative Distribution Function (cdf). It can be one of 'exponential' based on exponential cdf, 'empirical' for empirical cdf

`plot` logical to indicate whether the histogram plot (plus density or CDF plot) should be drawn. By default, it sets to false for no plotting

`verbose` logical to indicate whether the messages will be displayed in the screen. By default, it sets to true for display

`RData.location` the characters to tell the location of built-in RData files. See `xRDataLoader` for details

Value

a data frame with following columns:

- `Gene`: eQTL-containing genes
- `SNP`: eQTLs
- `Sig`: the eQTL mapping significant level (the best/minimum)
- `Weight`: the eQTL weight

Note

Pre-built eQTL datasets are described below according to the data sources.
1. Context-specific eQTLs in monocytes: resting and activating states. Sourced from Science 2014, 343(6175):1246949

- `JKscience_TS2A`: cis-eQTLs in either state (based on 228 individuals with expression data available for all experimental conditions).
- `JKscience_TS2A_CD14`: cis-eQTLs only in the resting/CD14+ state (based on 228 individuals).
- `JKscience_TS2A_LPS2`: cis-eQTLs only in the activating state induced by 2-hour LPS (based on 228 individuals).
- `JKscience_TS2A_LPS24`: cis-eQTLs only in the activating state induced by 24-hour LPS (based on 228 individuals).
- `JKscience_TS2A_IFN`: cis-eQTLs only in the activating state induced by 24-hour interferon-gamma (based on 228 individuals).
- `JKscience_TS2B`: cis-eQTLs in either state (based on 432 individuals).
- `JKscience_TS2B_CD14`: cis-eQTLs only in the resting/CD14+ state (based on 432 individuals).
- `JKscience_TS2B_LPS2`: cis-eQTLs only in the activating state induced by 2-hour LPS (based on 432 individuals).

- JKscience_TS2B_LPS24: cis-eQTLs only in the activating state induced by 24-hour LPS (based on 432 individuals).
- JKscience_TS2B_IFN: cis-eQTLs only in the activating state induced by 24-hour interferon-gamma (based on 432 individuals).
- JKscience_TS3A: trans-eQTLs in either state.

2. eQTLs in B cells. Sourced from Nature Genetics 2012, 44(5):502-510

   - JKng_bcell: cis- and trans-eQTLs.
   - JKng_bcell_cis: cis-eQTLs only.
   - JKng_bcell_trans: trans-eQTLs only.

3. eQTLs in monocytes. Sourced from Nature Genetics 2012, 44(5):502-510

   - JKng_mono: cis- and trans-eQTLs.
   - JKng_mono_cis: cis-eQTLs only.
   - JKng_mono_trans: trans-eQTLs only.

4. eQTLs in neutrophils. Sourced from Nature Communications 2015, 7(6):7545

   - JKnc_neutro: cis- and trans-eQTLs.
   - JKnc_neutro_cis: cis-eQTLs only.
   - JKnc_neutro_trans: trans-eQTLs only.

5. eQTLs in NK cells. Unpublished

   - JK_nk: cis-eQTLs.

6. Tissue-specific eQTLs from GTEx (version 4; incuding 13 tissues). Sourced from Science 2015, 348(6235):648-60

   - GTEx_V4_Adipose_Subcutaneous: cis-eQTLs in tissue 'Adipose Subcutaneous'.
   - GTEx_V4_Artery_Aorta: cis-eQTLs in tissue 'Artery Aorta'.
   - GTEx_V4_Artery_Tibial: cis-eQTLs in tissue 'Artery Tibial'.
   - GTEx_V4_Esophagus_Mucosa: cis-eQTLs in tissue 'Esophagus Mucosa'.
   - GTEx_V4_Esophagus_Muscularis: cis-eQTLs in tissue 'Esophagus Muscularis'.
   - GTEx_V4_Heart_Left_Ventricle: cis-eQTLs in tissue 'Heart Left Ventricle'.
   - GTEx_V4_Lung: cis-eQTLs in tissue 'Lung'.
   - GTEx_V4_Muscle_Skeletal: cis-eQTLs in tissue 'Muscle Skeletal'.
   - GTEx_V4_Nerve_Tibial: cis-eQTLs in tissue 'Nerve Tibial'.
   - GTEx_V4_Skin_Sun_Exposed_Lower_leg: cis-eQTLs in tissue 'Skin Sun Exposed Lower leg'.
   - GTEx_V4_Stomach: cis-eQTLs in tissue 'Stomach'.
   - GTEx_V4_Thyroid: cis-eQTLs in tissue 'Thyroid'.
   - GTEx_V4_Whole_Blood: cis-eQTLs in tissue 'Whole Blood'.

See Also

xRDataLoader

Examples

```
## Not run:
# Load the library
library(Pi)

## End(Not run)

RData.location <- "http://galahad.well.ox.ac.uk/bigdata_dev"
# a) provide the SNPs with the significance info
## get lead SNPs reported in AS GWAS and their significance info (p-values)
#data.file <- "http://galahad.well.ox.ac.uk/bigdata/AS.txt"
#AS <- read.delim(data.file, header=TRUE, stringsAsFactors=FALSE)
ImmunoBase <- xRDataLoader(RData.customised='ImmunoBase',
RData.location=RData.location)
gr <- ImmunoBase$AS$variants
AS <- as.data.frame(GenomicRanges::mcols(gr)[, c('Variant','Pvalue')])

## Not run:
# b) define eQTL genes
df_eGenes <- xSNP2eGenes(data=AS[,1], include.eQTL="JKscience_TS2A",
RData.location=RData.location)

## End(Not run)
```

---

xSNPeqtl — *Function to extract eQTL-gene pairs given a list of SNPs or a customised eQTL mapping data*

Description

xSNPeqtl is supposed to extract eQTL-gene pairs given a list of SNPs or a customised eQTL mapping data.

Usage

```
xSNPeqtl(data = NULL, include.eQTL = c(NA, "JKscience_TS2A",
"JKscience_TS2A_CD14", "JKscience_TS2A_LPS2", "JKscience_TS2A_LPS24",
"JKscience_TS2A_IFN", "JKscience_TS2B", "JKscience_TS2B_CD14",
"JKscience_TS2B_LPS2", "JKscience_TS2B_LPS24", "JKscience_TS2B_IFN",
"JKscience_TS3A", "JKng_bcell", "JKng_bcell_cis", "JKng_bcell_trans",
"JKng_mono", "JKng_mono_cis", "JKng_mono_trans", "JKnc_neutro",
"JKnc_neutro_cis", "JKnc_neutro_trans", "JK_nk",
"GTEx_V4_Adipose_Subcutaneous", "GTEx_V4_Artery_Aorta",
"GTEx_V4_Artery_Tibial", "GTEx_V4_Esophagus_Mucosa",
"GTEx_V4_Esophagus_Muscularis", "GTEx_V4_Heart_Left_Ventricle",
"GTEx_V4_Lung", "GTEx_V4_Muscle_Skeletal", "GTEx_V4_Nerve_Tibial",
"GTEx_V4_Skin_Sun_Exposed_Lower_leg", "GTEx_V4_Stomach",
"GTEx_V4_Thyroid",
"GTEx_V4_Whole_Blood", "eQTLdb_NK", "eQTLdb_CD14", "eQTLdb_LPS2",
"eQTLdb_LPS24", "eQTLdb_IFN"), eQTL.customised = NULL, verbose = TRUE,
RData.location = "http://galahad.well.ox.ac.uk/bigdata")
```

Arguments

`data` NULL or a input vector containing SNPs. If NULL, all SNPs will be considered. If a input vector containing SNPs, SNPs should be provided as dbSNP ID (ie starting with rs). Alternatively, they can be in the format of 'chrN:xxx', where N is either 1-22 or X, xxx is number; for example, 'chr16:28525386'

`include.eQTL` genes modulated by eQTL (also Lead SNPs or in LD with Lead SNPs) are also included. By default, it is 'NA' to disable this option. Otherwise, those genes modulated by eQTL will be included. Pre-built eQTL datasets are detailed in the section 'Note'

`eQTL.customised` a user-input matrix or data frame with 3 columns: 1st column for SNPs/eQTLs, 2nd column for Genes, and 3rd for eQTL mapping significance level (p-values or FDR). It is designed to allow the user analysing their eQTL data. This customisation (if provided) has the high priority over built-in eQTL data.

`verbose` logical to indicate whether the messages will be displayed in the screen. By default, it sets to true for display

`RData.location` the characters to tell the location of built-in RData files. See `xRDataLoader` for details

Value

a data frame with following columns:

- `SNP`: eQTLs
- `Gene`: eQTL-containing genes
- `Sig`: the eQTL mapping significant level
- `Context`: the context in which eQTL data was generated

Note

Pre-built eQTL datasets are described below according to the data sources.
1. Context-specific eQTLs in monocytes: resting and activating states. Sourced from Science 2014, 343(6175):1246949

- `JKscience_TS2A`: cis-eQTLs in either state (based on 228 individuals with expression data available for all experimental conditions).
- `JKscience_TS2A_CD14`: cis-eQTLs only in the resting/CD14+ state (based on 228 individuals).
- `JKscience_TS2A_LPS2`: cis-eQTLs only in the activating state induced by 2-hour LPS (based on 228 individuals).
- `JKscience_TS2A_LPS24`: cis-eQTLs only in the activating state induced by 24-hour LPS (based on 228 individuals).
- `JKscience_TS2A_IFN`: cis-eQTLs only in the activating state induced by 24-hour interferon-gamma (based on 228 individuals).
- `JKscience_TS2B`: cis-eQTLs in either state (based on 432 individuals).
- `JKscience_TS2B_CD14`: cis-eQTLs only in the resting/CD14+ state (based on 432 individuals).
- `JKscience_TS2B_LPS2`: cis-eQTLs only in the activating state induced by 2-hour LPS (based on 432 individuals).

- JKscience_TS2B_LPS24: cis-eQTLs only in the activating state induced by 24-hour LPS (based on 432 individuals).
- JKscience_TS2B_IFN: cis-eQTLs only in the activating state induced by 24-hour interferon-gamma (based on 432 individuals).
- JKscience_TS3A: trans-eQTLs in either state.

2. eQTLs in B cells. Sourced from Nature Genetics 2012, 44(5):502-510

   - JKng_bcell: cis- and trans-eQTLs.
   - JKng_bcell_cis: cis-eQTLs only.
   - JKng_bcell_trans: trans-eQTLs only.

3. eQTLs in monocytes. Sourced from Nature Genetics 2012, 44(5):502-510

   - JKng_mono: cis- and trans-eQTLs.
   - JKng_mono_cis: cis-eQTLs only.
   - JKng_mono_trans: trans-eQTLs only.

4. eQTLs in neutrophils. Sourced from Nature Communications 2015, 7(6):7545

   - JKnc_neutro: cis- and trans-eQTLs.
   - JKnc_neutro_cis: cis-eQTLs only.
   - JKnc_neutro_trans: trans-eQTLs only.

5. eQTLs in NK cells. Unpublished

   - JK_nk: cis-eQTLs.

6. Tissue-specific eQTLs from GTEx (version 4; incuding 13 tissues). Sourced from Science 2015, 348(6235):648-60

   - GTEx_V4_Adipose_Subcutaneous: cis-eQTLs in tissue 'Adipose Subcutaneous'.
   - GTEx_V4_Artery_Aorta: cis-eQTLs in tissue 'Artery Aorta'.
   - GTEx_V4_Artery_Tibial: cis-eQTLs in tissue 'Artery Tibial'.
   - GTEx_V4_Esophagus_Mucosa: cis-eQTLs in tissue 'Esophagus Mucosa'.
   - GTEx_V4_Esophagus_Muscularis: cis-eQTLs in tissue 'Esophagus Muscularis'.
   - GTEx_V4_Heart_Left_Ventricle: cis-eQTLs in tissue 'Heart Left Ventricle'.
   - GTEx_V4_Lung: cis-eQTLs in tissue 'Lung'.
   - GTEx_V4_Muscle_Skeletal: cis-eQTLs in tissue 'Muscle Skeletal'.
   - GTEx_V4_Nerve_Tibial: cis-eQTLs in tissue 'Nerve Tibial'.
   - GTEx_V4_Skin_Sun_Exposed_Lower_leg: cis-eQTLs in tissue 'Skin Sun Exposed Lower leg'.
   - GTEx_V4_Stomach: cis-eQTLs in tissue 'Stomach'.
   - GTEx_V4_Thyroid: cis-eQTLs in tissue 'Thyroid'.
   - GTEx_V4_Whole_Blood: cis-eQTLs in tissue 'Whole Blood'.

See Also

xRDataLoader

Examples

```
## Not run:
# Load the library
library(Pi)

## End(Not run)

RData.location <- "http://galahad.well.ox.ac.uk/bigdata_dev"
# a) provide the SNPs with the significance info
## get lead SNPs reported in AS GWAS and their significance info (p-values)
#data.file <- "http://galahad.well.ox.ac.uk/bigdata/AS.txt"
#AS <- read.delim(data.file, header=TRUE, stringsAsFactors=FALSE)
ImmunoBase <- xRDataLoader(RData.customised='ImmunoBase',
RData.location=RData.location)
gr <- ImmunoBase$AS$variants
AS <- as.data.frame(GenomicRanges::mcols(gr)[, c('Variant','Pvalue')])

## Not run:
# b) define eQTL genes
df_SGS <- xSNPeqtl(data=AS[,1], include.eQTL="JKscience_TS2A",
RData.location=RData.location)

## End(Not run)
```

---

xSNPhic *Function to extract promoter capture HiC-gene pairs given a list of SNPs*

Description

xSNPhic is supposed to extract HiC-gene pairs given a list of SNPs.

Usage

```
xSNPhic(data = NULL, entity = c("SNP", "chr:start-end", "data.frame",
"bed",
"GRanges"), include.HiC = c(NA, "Monocytes", "Macrophages_M0",
"Macrophages_M1", "Macrophages_M2", "Neutrophils", "Megakaryocytes",
"Endothelial_precursors", "Erythroblasts", "Fetal_thymus",
"Naive_CD4_T_cells", "Total_CD4_T_cells",
"Activated_total_CD4_T_cells",
"Nonactivated_total_CD4_T_cells", "Naive_CD8_T_cells",
"Total_CD8_T_cells",
"Naive_B_cells", "Total_B_cells", "PE.Monocytes", "PE.Macrophages_M0",
"PE.Macrophages_M1", "PE.Macrophages_M2", "PE.Neutrophils",
"PE.Megakaryocytes", "PE.Erythroblasts", "PE.Naive_CD4_T_cells",
"PE.Naive_CD8_T_cells"), GR.SNP = c("dbSNP_GWAS", "dbSNP_Common"),
verbose = TRUE, RData.location =
"http://galahad.well.ox.ac.uk/bigdata")
```

Arguments

`data` NULL or a input vector containing SNPs. If NULL, all SNPs will be considered. If a input vector containing SNPs, SNPs should be provided as dbSNP ID (ie starting with rs) or in the format of 'chrN:xxx', where N is either 1-22 or X, xxx is number; for example, 'chr16:28525386'. Alternatively, it can be other formats/entities (see the next parameter 'entity')

`entity` the data entity. By default, it is "SNP". For general use, it can also be one of "chr:start-end", "data.frame", "bed" or "GRanges"

`include.HiC` genes linked to input SNPs are also included. By default, it is 'NA' to disable this option. Otherwise, those genes linked to SNPs will be included according to Promoter Capture HiC (PCHiC) datasets. Pre-built HiC datasets are detailed in the section 'Note'

`GR.SNP` the genomic regions of SNPs. By default, it is 'dbSNP_GWAS', that is, SNPs from dbSNP (version 146) restricted to GWAS SNPs and their LD SNPs (hg19). It can be 'dbSNP_Common', that is, Common SNPs from dbSNP (version 146) plus GWAS SNPs and their LD SNPs (hg19). Alternatively, the user can specify the customised input. To do so, first save your RData file (containing an GR object) into your local computer, and make sure the GR object content names refer to dbSNP IDs. Then, tell "GR.SNP" with your RData file name (with or without extension), plus specify your file RData path in "RData.location". Note: you can also load your customised GR object directly

`verbose` logical to indicate whether the messages will be displayed in the screen. By default, it sets to true for display

`RData.location` the characters to tell the location of built-in RData files. See `xRDataLoader` for details

Value

If input data is NULL, a data frame with following columns:

- `from`: baited genomic regions (baits)
- `to`: preyed (other end) genomic regions of interactions (preys)
- `score`: CHiCAGO scores quantifying the strength of physical interactions between harbors and partners

If input data is not NULL, a list with two components: "df" and "ig". "df" is a data frame with following columns:

- `from`: 'from/bait' genomic regions
- `to`: 'to/prey' genomic regions
- `score`: CHiCAGO scores quantifying the strength of physical interactions between baits and preys
- `from_genes`: genes associated with 'from/bait' genomic regions
- `to_genes`: genes associated with 'to/prey' genomic regions
- `SNP`: input SNPs (in query)
- `SNP_end`: specify which end SNPs in query fall into (either 'bait/from' or 'prey/to')
- `SNP_harbor`: genomic regions harbors the SNPs in query
- `Context`: the context in which PCHiC data was generated

"ig" is an object of both classes "igraph" and "PCHiC", a direct graph with nodes for genomic regions and edges for CHiCAGO scores between them. Also added node attribute is 1) 'target' storing genes assocated and 2) 'SNP' for input SNPs (if the node harboring input SNPs). If several cell types are queried, "ig" is actually a list of "igraph"/"PCHiC" objects.

Note

Pre-built HiC datasets are described below according to the data sources.
1. Promoter Capture HiC datasets in 17 primary blood cell types. Sourced from Cell 2016, 167(5):1369-1384.e19

- `Monocytes`: physical interactions (CHiCAGO score >=5) of promoters (baits) with the other end (preys) in Monocytes.
- `Macrophages_M0`: promoter interactomes in Macrophages M0.
- `Macrophages_M1`: promoter interactomes in Macrophages M1.
- `Macrophages_M2`: promoter interactomes in Macrophages M2.
- `Neutrophils`: promoter interactomes in Neutrophils.
- `Megakaryocytes`: promoter interactomes in Megakaryocytes.
- `Endothelial_precursors`: promoter interactomes in Endothelial precursors.
- `Fetal_thymus`: promoter interactomes in Fetal thymus.
- `Naive_CD4_T_cells`: promoter interactomes in Naive CD4+ T cells.
- `Total_CD4_T_cells`: promoter interactomes in Total CD4+ T cells.
- `Activated_total_CD4_T_cells`: promoter interactomes in Activated total CD4+ T cells.
- `Nonactivated_total_CD4_T_cells`: promoter interactomes in Nonactivated total CD4+ T cells.
- `Naive_CD8_T_cells`: promoter interactomes in Naive CD8+ T cells.
- `Total_CD8_T_cells`: promoter interactomes in Total CD8+ T cells.
- `Naive_B_cells`: promoter interactomes in Naive B cells.
- `Total_B_cells`: promoter interactomes in Total B cells.

2. Promoter Capture HiC datasets (involving active promoters and enhancers) in 9 primary blood cell types. Sourced from Cell 2016, 167(5):1369-1384.e19

- `PE.Monocytes`: physical interactions (CHiCAGO score >=5) of promoters (baits) with the other end (enhancers as preys) in Monocytes.
- `PE.Macrophages_M0`: promoter-enhancer interactomes in Macrophages M0.
- `PE.Macrophages_M1`: promoter-enhancer interactomes in Macrophages M1.
- `PE.Macrophages_M2`: promoter-enhancer interactomes in Macrophages M2.
- `PE.Neutrophils`: promoter-enhancer interactomes in Neutrophils.
- `PE.Megakaryocytes`: promoter-enhancer interactomes in Megakaryocytes.
- `PE.Erythroblasts`: promoter-enhancer interactomes in Erythroblasts.
- `PE.Naive_CD4_T_cells`: promoter-enhancer interactomes in Naive CD4+ T cells.
- `PE.Naive_CD8_T_cells`: promoter-enhancer interactomes in Naive CD8+ T cells.

See Also

`xRDataLoader`

Examples

```
## Not run:
# Load the library
library(Pi)

## End(Not run)

RData.location <- "http://galahad.well.ox.ac.uk/bigdata_dev"
# a) provide the SNPs with the significance info
## get lead SNPs reported in AS GWAS and their significance info (p-values)
#data.file <- "http://galahad.well.ox.ac.uk/bigdata/AS.txt"
#AS <- read.delim(data.file, header=TRUE, stringsAsFactors=FALSE)
ImmunoBase <- xRDataLoader(RData.customised='ImmunoBase',
RData.location=RData.location)
data <- names(ImmunoBase$AS$variants)

## Not run:
# b) extract HiC-gene pairs given a list of AS SNPs
PCHiC <- xSNPhic(data, include.HiC="Monocytes", GR.SNP="dbSNP_GWAS",
RData.location=RData.location)
head(PCHiC$df)

# c) visualise the interaction (a directed graph: bait->prey)
g <- PCHiC$ig
## a node with SNPs colored in 'skyblue' and the one without SNPs in 'pink'
## the width in an edge is proportional to the interaction strength
xPCHiCplot(g, vertex.label.cex=0.5)
xPCHiCplot(g, glayout=layout_in_circle, vertex.label.cex=0.5)

## End(Not run)
```

Index