Package ‘PCAN’

November 21, 2016

Title Phenotype Consensus ANalysis (PCAN)
Version 1.2.0
Date 2016-03-24
Author Matthew Page and Patrice Godard
Maintainer Matthew Page <matthew.page@ucb.com> and Patrice Godard <patrice.godard@gmail.com>

Description Phenotypes comparison based on a pathway consensus approach. Assess the relationship between candidate genes and a set of phenotypes based on additional genes related to the candidate (e.g. Pathways or network neighbors).

biocViews Annotation, Sequencing, Genetics, FunctionalPrediction, VariantAnnotation, Pathways, Network

License CC BY-NC-ND 4.0

Depends R (>= 3.3), BiocParallel
Imports grDevices, stats
Suggests BiocStyle, knitr, rmarkdown, reactome.db, STRINGdb

VignetteBuilder knitr
RoxygenNote 5.0.1
NeedsCompilation no

R topics documented:

calcHpSim ................................................................. 2
compareHPSets ......................................................... 3
computeHpiC ............................................................ 5
geneByHp ............................................................... 6
geneByTrait ............................................................ 7
geneDef ................................................................. 8
hpByTrait ............................................................... 9
hpDef ................................................................. 10
hpGeneHeatmap ......................................................... 11
hpGeneListComp ....................................................... 16
hpSetCompBestMatch ................................................ 22
hpSetCompSummary .................................................. 23
hp_ancestors ......................................................... 24
calcHpSim

Description
This function compares 2 HP terms based on provided Information Content and ancestors.

Usage
```r
calcHpSim(term1, term2, method = c("Resnik"), IC, ancestors)
```

Arguments
- `term1`: one of the HP term to compare
- `term2`: the other HP term to compare
- `method`: the method for computing semantic similarity (default: "Resnik" returns the IC of the MICA: Most Informative common ancestor)
- `IC`: a named vector of Information Content by HP term
- `ancestors`: a named list of ancestors by HP term

Value
A numeric value

Author(s)
Patrice Godard

See Also
`compareHPSets`

Examples
```r
## Prerequisite
data(geneByHp, hp_descendants, package="PCAN")
geneByHp <- unstack(geneByHp, entrez~hp)
ic <- computeHpIC(geneByHp, hp_descendants)

## Compute similarity between different couples of HP terms
data(hp_ancestors, hpDef, package="PCAN")
hp1 <- "HP:0000518"
hp2 <- "HP:0030084"
hp3 <- "HP:0002119"
```
compareHPSets

Compare 2 sets of HP terms based on semantic similarity

Description

This function compares each couple of HP terms from each of the 2 provided sets based on Information Content (IC)

Usage

compareHPSets(hpSet1, hpSet2, IC, ancestors, method = "Resnik", BPPARAM = bpparam())

Arguments

hpSet1 a set of HP terms
hpSet2 another set of HP terms
IC a named vector of Information Content by HP term
ancestors a named list of ancestors by HP term
method the method for computing semantic similarity among those available in calcHpSim (default: "Resnik" returns the IC of the MICA: Most Informative common ancestor)
BPPARAM An optional BiocParallelParam instance defining the parallel back-end to be used during evaluation (used internally by the bpmapply function).

Value

A matrix of semantic similarity

Author(s)

Patrice Godard

See Also

calcHpSim
## Examples

```r
## Prerequisite
data(geneByHp, hp_descendants, package="PCAN")
geneByHp <- unstack(geneByHp, entrez~hp)
ic <- computeHpIC(geneByHp, hp_descendants)

## Use case: comparing a gene and a disease
data(traitDef, geneDef, hp_ancestors, hpDef, package="PCAN")
omim <- "612285"
traitDef[which(traitDef$id==omim),]
entrez <- "57545"
geneDef[which(geneDef$entrez==entrez),]
## Get HP terms associated to the disease
data(hpByTrait, package="PCAN")
hpOfInterest <- hpByTrait$hp[which(hpByTrait$id==omim)]

## Get HP terms associated to the gene
hpByGene <- unstack(stack(geneByHp), ind~values)
geneHps <- hpByGene[[entrez]]
## Comparison of the two sets of HP terms
compMat <- compareHPSets(
  hpSet1=geneHps, hpSet2=hpOfInterest,
  IC=ic,
  ancestors=hp_ancestors,
  method="Resnik",
  BPPARAM=SerialParam()
)
## Get the symmetric semantic similarity score
hpSetCompSummary(compMat, method="bma", direction="symSim")
bm <- hpSetCompBestMatch(compMat, "b")
hpDef[match(c(bm$compared, bm$candidate), hpDef$id),]
## Assessing the significance of this score by comparing to all other genes
hpGeneResnik <- compareHPSets(
  hpSet1=names(ic), hpSet2=hpOfInterest,
  IC=ic,
  ancestors=hp_ancestors,
  method="Resnik",
  BPPARAM=SerialParam()
)
hpMatByGene <- lapply(
  hpByGene,
  function(x){
    hpGeneResnik[x, , drop=FALSE]
  }
)
resnSss <- unlist(lapply(
  hpMatByGene,
  hpSetCompSummary,
  method="bma", direction="symSim"
))
candScore <- resnSss[entrez]
hist(
  resnSss,
  breaks=100, col="grey",
  ylim=c(0,300),
)
```
computeHpIC

Compute Information Content (IC) for each HP based on genes by HP

Description

Compute Information Content (IC) for each HP based on genes by HP

Usage

computeHpIC(content, hp.descendants)

Arguments

- content: a list providing the content associated to each HP
- hp.descendants: a list providing for each HP all its descendant HP terms

Details

This function assumes that all the HP terms taken into account belong to the same family of terms (i.e., they are all descendants of the same term).

Value

A vector of IC named with HP terms
Examples

###########################################
## Compute information content of each HP according to associated genes
data(geneByHp, hp_descendants, package="PCAN")
geneByHp <- unstack(geneByHp, entrez~hp)
ic <- computeHpIC(geneByHp, hp_descendants)
hist(
ic,
  breaks=100, col="grey",
  main="Distribution of Information Content",
  xlab="IC base on genes associated to HP"
)

geneByHp  
Entrez gene IDs associated to HP terms (Example data)

Description

Each entrez gene IDs is associated to one or several HP terms

Format

A data frame with 67989 rows and 2 columns:

  entrez  entrez gene IDs
  hp      HP terms

Details

These data are used to examplify the different functions of the package. More data are available in the MultiHumanPhenoDB package.

Source

Two ressources were used in May 27 2015:

- ftp://ftp.ncbi.nlm.nih.gov/pub/clinvar/xml/ClinVarFullRelease_2015-05.xml.gz was used to find genes associated to each OMIM disease with a "Pathogenic" clinical status and one of the following origins: "germline", "de novo", "inherited", "maternal", "paternal", "biparental", "uniparental".
- http://compbio.charite.de/hudson/job/hpo.annotations/1039/artifact/misc/phenotype_annotation.tab was used to find HP associated to each OMIM disease.

Examples

###########################################
## Compute information content of each HP according to associated genes
data(geneByHp, hp_descendants, package="PCAN")
geneByHp <- unstack(geneByHp, entrez~hp)
ic <- computeHpIC(geneByHp, hp_descendants)
hist(
ic,
  breaks=100, col="grey",

geneByTrait

main="Distribution of Information Content",
xlab="IC base on genes associated to HP"
)

geneByTrait

**Gene associated to trait (Example data)**

**Description**

Each trait is associated to one or several genes. Only genes associated to OMIM disease with a "Pathogenic" clinical status and one of the following origins: "germline", "de novo", "inherited", "maternal", "paternal", "biparental", "uniparental".

**Format**

A data frame with 4569 rows and 3 columns:

- **entrez**: Entrez gene IDs.
- **db**: Trait database: always "OMIM" here.
- **id**: Trait ID: OMIM IDs here

**Details**

These data are used to exemplify the different functions of the package. More data are available in the MultiHumanPhenoDB package.

**Source**


**Examples**

data(geneByTrait, traitDef, geneDef, package="PCAN")
omim <- "612285"
traitDef[which(traitDef$id==omim),]
# Gene associated to this disease
entrez <- geneByTrait[which(geneByTrait$id==omim), "entrez"]
gegeneDef[which(geneDef$entrez %in% entrez),]
# All diseases associated to this gene
traitDef[
    which(
        traitDef$id %in%
        geneByTrait[which(geneByTrait$entrez==entrez), "id"]
    ),
]
Description of genes (Example data)

Description

Basic information about genes. Only genes associated to at least one OMIM disease are taken into account.

Format

A data frame with 3265 rows and 3 columns:

- **entrez**: Entrez gene ID.
- **name**: Gene name.
- **symbol**: Gene symbol.

Details

These data are used to examplify the different functions of the package. More data are available in the MultiHumanPhenoDB package.

Source


Examples

```r
## Prerequisite
data(geneByHp, hp_descendants, package="PCAN")
geneByHp <- unstack(geneByHp, entrez~hp)
ic <- computeHpIC(geneByHp, hp_descendants)

## Use case: comparing a gene and a disease
data(traitDef, geneDef, hp_ancestors, hpDef, package="PCAN")
omim <- "612285"
traitDef[which(traitDef$id==omim),]
entrez <- "57545"
geneDef[which(geneDef$entrez==entrez),]

## Get HP terms associated to the disease
data(hpByTrait, package="PCAN")
hpOfInterest <- hpByTrait$hp[which(hpByTrait$id==omim)]

## Get HP terms associated to the gene
hpByGene <- unstack(stack(geneByHp), ind~values)
geneHps <- hpByGene[[entrez]]

## Comparison of the two sets of HP terms
compMat <- compareHPSets(
  hpSet1=geneHps, hpSet2=hpOfInterest, IC=ic,
  ancestors=hp_ancestors, method="Resnik",
  BPPARAM=SerialParam())
```
## Get the symmetric semantic similarity score

```r
hpSetCompSummary(compMat, method="bma", direction="symSim")
```

```r
bm <- hpSetCompBestMatch(compMat, "b")
```

```r
hpDef[match(c(bm$compared, bm$candidate), hpDef$id),]
```

---

### hpByTrait

**HP IDs associated to trait (Example data)**

<table>
<thead>
<tr>
<th>Description</th>
<th>Format</th>
<th>Details</th>
<th>Source</th>
<th>Examples</th>
</tr>
</thead>
</table>
| Each trait is associated to one or several HP terms. | A data frame with 55311 rows and 3 columns: | These data are used to examplify the different functions of the package. More data are available in the MultiHumanPhenoDB package. | [http://compbio.charite.de/hudson/job/hpo.annotations/1039/artifact/misc/phenotype_annotation.tab](http://compbio.charite.de/hudson/job/hpo.annotations/1039/artifact/misc/phenotype_annotation.tab) | ```r
## Prerequisite
data(geneByHp, hp_descendants, package="PCAN")
geneByHp <- unstack(geneByHp, entrez~hp)
ic <- computeHpIC(geneByHp, hp_descendants)

## Use case: comparing a gene and a disease
data(traitDef, geneDef, hp_ancestors, hpDef, package="PCAN")
omim <- "612285"
traitDef[which(traitDef$id==omim),]
entrez <- "57545"
geneDef[which(geneDef$entrez==entrez),]

## Get HP terms associated to the disease
data(hpByTrait, package="PCAN")
hpOfInterest <- hpByTrait$hp[which(hpByTrait$id==omim)]

## Get HP terms associated to the gene
hpByGene <- unstack(stack(geneByHp), ind~values)
geneHps <- hpByGene[[entrez]]

## Comparison of the two sets of HP terms``` |
compMat <- compareHPSets(
  hpSet1= geneHps, hpSet2=hpOfInterest,
  IC=ic,
  ancestors=hp_ancestors,
  method="Resnik",
  BPPARAM=SerialParam()
)

## Get the symmetric semantic similarity score
hpSetCompSummary(compMat, method="bma", direction="symSim")
bm <- hpSetCompBestMatch(compMat, "b")
hpDef[match(c(bm$compared, bm$candidate), hpDef$id),]

---

### Description

**Description**

HP terms basic information. Only descendants of 'Phenotypic abnormality' were taken into account.

**Format**

A data frame with 10962 rows and 2 columns:

- **id**: HP term IDs
- **name**: HP term names

**Details**

These data are used to exemplify the different functions of the package. More data are available in the MultiHumanPhenoDB package.

**Source**


**Examples**

#### Prerequisite

data(geneByHp, hp_descendants, package="PCAN")
geneByHp <- unstack(geneByHp, entrez~hp)
ic <- computeHpIC(geneByHp, hp_descendants)

#### Compute similarity between different couples of HP terms

data(hp_ancestors, hpDef, package="PCAN")

```r
hp1 <- "HP:0000518"
hp2 <- "HP:0030084"
hp3 <- "HP:0002119"
hp4 <- "HP:0001305"
hpDef[which(hpDef$id %in% c(hp1, hp2)), 1:2]
calcHpSim(hp1, hp2, IC=ic, ancestors=hp_ancestors)
hpDef[which(hpDef$id %in% c(hp2, hp3)), 1:2]
calcHpSim(hp2, hp3, IC=ic, ancestors=hp_ancestors)
hpDef[which(hpDef$id %in% c(hp2, hp4)), 1:2]
```
calcHpSim(hp2, hp4, IC=ic, ancestors=hp_ancestors)
hpDef[which(hpDef$id %in% c(hp3, hp4)), 1:2]
calcHpSim(hp3, hp4, IC=ic, ancestors=hp_ancestors)

---

**hpGeneHeatmap**

**HP to Gene heatmap**

**Description**

This function draw a heatmap corresponding to the result of the pathway consensus method. For each gene of the pathway under focus and each HP of interest it shows the best score.

**Usage**

```r
hpGeneHeatmap(hpGeneListRes, genesOfInterest = NULL, geneLabels = NULL, hpLabels = NULL, clustByGene = TRUE, clustByHp = TRUE, palFun = colorRampPalette(c("white", "red")), goiCol = "blue", ...)
```

**Arguments**

- `hpGeneListRes` the result of the `hpGeneListComp` function.
- `genesOfInterest` a list of gene to highlight.
- `geneLabels` a named vector of gene labels (all the genes id found in `hpGeneListRes` should be in `names(geneLabels)`).
- `hpLabels` a named vector of HP labels (all the HP id found in `hpGeneListRes` should be in `names(hpLabels)`).
- `clustByGene` should the heatmap be clustered according to genes (default: TRUE).
- `clustByHp` should the heatmap be clustered according to HP (default: TRUE).
- `palFun` the palette function for the heatmap.
- `goiCol` the color used to highlight genes of interest.
- `...` parameters for the `codeheatmap` function

**Value**

A list of 2 matrix (invisible return):

- **bmValues** For each gene and each HP of interest the best match value.
- **bestMatches** The gene associated HP best matching the HP of interest.

**See Also**

`hpGeneListComp, hpSetCompBestMatch`
Examples

data(geneByHp, hp_descendants, package="PCAN")
data(hp_ancestors, hpDef, package="PCAN")
data(traitDef, geneDef, package="PCAN")
data(hpByTrait, package="PCAN")
geneByHp <- unstack(geneByHp, entrez~hp)

#################################################################
## Compute information content of each HP according to associated genes
ic <- computeHpIC(geneByHp, hp_descendants)

#################################################################
## Use case: comparing a gene and a disease
omim <- "612285"
traitDef[which(traitDef$id==omim),]
entrez <- "57545"
geneDef[which(geneDef$entrez==entrez),]

## Get HP terms associated to the disease
hpOfInterest <- hpByTrait$hp[which(hpByTrait$id==omim)]

## Get HP terms associated to the gene
geneHps <- unstack(stack(geneByHp), ind~values)

## HP Comparison
hpGeneResnik <- compareHPSets(
  hpSet1=names(ic), hpSet2=hpOfInterest,
  IC=ic,
  ancestors=hp_ancestors,
  method="Resnik",
  BPPARAM=SerialParam()
)

hpMatByGene <- lapply(
  hpByGene,
  function(x){
    hpGeneResnik[x, , drop=FALSE]
  }
)

resnSss <- unlist(lapply(
  hpMatByGene,
  hpSetCompSummary,
  method="bma", direction="symSim"
))
candScore <- resnSss[entrez]

#################################################################
## The pathway consensus approach
## What about genes belonging to the same pathways as the candidate
data(rPath, hsEntrezByRPath, package="PCAN")
candPath <- names(hsEntrezByRPath)[which(unlist(lapply(
  hsEntrezByRPath,
  function(x) entrez %in% x
)))]
rPath[which(rPath$Pathway %in% candPath),]
rPathRes <- hpGeneListComp(
  geneList=hsEntrezByRPath[[candPath]],
  ssMatByGene = hpMatByGene,
  geneSSScore = resnSss
)
> hist(
  resnSss,
  breaks=100, col="grey",
  ylim=c(0,5),
  xlab=expression(Sim[sym]),
  ylab="Density",
  main=paste(  
    "Distribution of symmetric semantic similarity scores for all the",
    length(resnSss), "genes"
  ),
  probability=TRUE
)

> toAdd <- hist(
  rPathRes$scores,
  breaks=100,
  plot=FALSE
)

for(i in 1:length(toAdd$density)){
  polygon(
    x=toAdd$breaks[c(i, i+1, i+1, i)],
    y=c(0, 0, rep(toAdd$density[i], 2)),
    col="#BE000040",
    border="#800000FF"
  )
}

> legend(
  "topright",
  paste0(  
    "Genes belonging to the ", candPath,
    " pathway:
    "
   桕rPath[which(rPath$Pathway %in% candPath),"Pathway_name"],
    "\n\nand with a symmetric semantic similarity score (",
    sum(!is.na(rPathRes$scores)),
    "/",
    length(rPathRes$scores),
    ")\n",
    "p-value: ", signif(rPathRes$p.value, 2)
  ),
  pch=15,
  col="#BE000040",
  bty="n",
  cex=0.6
)

## Assessing the symmetric semantic similarity for each gene in the pathway

pathSss <- rPathRes$scores[which(!is.na(rPathRes$scores))]

names(pathSss) <- geneDef[match(names(pathSss), geneDef$entrez), "symbol"]
opar <- par(mar=c(7.1, 4.1, 4.1, 2.1))

barplot(  
  sort(pathSss),
  las=2,
  ylab=expression(Sim[sym]),
  main=rPath[which(rPath$Pathway %in% candPath),"Pathway_name"]
)

p <- c(0.25, 0.5, 0.75, 0.95)

abline(  
  h=quantile(resnSss, probs=p),
  col="#BE000000",
  lty=1
)
```r

ty=c(2, 1, 2, 2),
lwd=c(2, 2, 2, 1)

```

```r

text(
  rep(0, 4),
  quantile(resnSss, probs=p),
  pos=3,
  offset=0,
  col="#BE0000",
  cex=0.6
)

```

```r

legend(
  "topleft",
  paste0(
    "Quantiles of the distribution of symmetric semantic similarity\n", 
    "scores for all the genes."
  ),
  lty=1,
  col="#BE0000",
  cex=0.6
)

```

```
par(opar)

```r

```r

# A heatmap showing the best HP match for each gene in the pathway
geneLabels <- geneDef$symbol[which(! duplicated(geneDef$entrez))]
names(geneLabels) <- geneDef$entrez[which(! duplicated(geneDef$entrez))]
hpLabels <- hpDef$name
names(hpLabels) <- hpDef$id
hpGeneHeatmap(
  rPathRes,
  genesOfInterest=entrez,
  geneLabels=geneLabels,
  hpLabels=hpLabels,
  clustByGene=TRUE,
  clustByHp=TRUE,
  palFun=colorRampPalette(c("white", "red")),
  goiCol="blue",
  main=rPath[which(rPath$Pathway %in% candPath), "Pathway_name"]
)

```

```r

```r

### What about genes interacting with the candidate (including itself)
data(hqStrNw, package="PCAN")
neighbors <- unique(c(
  hqStrNw$gene1[which(hqStrNw$gene2==entrez)],
  hqStrNw$gene2[which(hqStrNw$gene1==entrez)],
  entrez
))
neighRes <- hpGeneListComp(
  geneList=neighbors,
  ssMatByGene = hpMatByGene,
  geneSSScore = resnSss
)
hist(
  resnSss,
  breaks=100, col="grey",
)```
ylim=c(0,10),
xlab=expression(Sim[sym]),
ylab="Density",
main=paste(
  "Distribution of symmetric semantic similarity scores for all the",
  length(resnSss), "genes"
),
probability=TRUE
)
toAdd <- hist(
  neighRes$scores,
  breaks=100,
  plot=FALSE
)
for(i in 1:length(toAdd$density)){
  polygon(
    x=toAdd$breaks[c(i, i+1, i+1, i)],
    y=c(0, 0, rep(toAdd$density[i], 2)),
    col="#BE000040",
    border="#800000FF"
  )
}
legend(
  "topright",
  paste0(
    "Genes interacting with ",
    geneDef[which(geneDef$entrez==entrez),"symbol"],
    " (", entrez, ")",
    "nand with a symmetric semantic similarity score (",
    sum(!is.na(neighRes$scores)),
    ",")",
    length(neighRes$scores),
    ")\n",
    "p-value: ", signif(neighRes$p.value, 2)
  ),
  pch=15,
  col="#8E000040",
  bty="n",
  cex=0.6
)
## Assessing the symmetric semantic similarity score for each interacting gene
neighSss <- neighRes$scores[which(!is.na(neighRes$scores))]
names(neighSss) <- geneDef[match(names(neighSss), geneDef$entrez), "symbol"]
opar <- par(mar=c(7.1, 4.1, 4.1, 2.1))
barplot(
  sort(neighSss),
  las=2,
  ylab=expression(Sim[sym]),
  main=paste0(
    "Genes interacting with ",
    geneDef[which(geneDef$entrez==entrez),"symbol"],
    " (", entrez, ")"
  )
)
p <- c(0.25, 0.5, 0.75, 0.95)
abline(
## hpGeneListComp

**HP semantic similarity for a whole gene list.**

### Description

This function compares a whole gene list to a set of HP terms using a matrix of semantic similarity.

### Usage

```r
hpGeneListComp(geneList, ssMatByGene, geneSSScore = NULL, ...)
```

### Arguments

- `geneList` a vector providing the genes of interest.
ssMatByGene a list (one element per gene) of matrix of semantic similarity between HP terms as returned by \texttt{compareHPSets}. This list has to be unbiased in order to compute p-values.

geneSSScore a vector of semantic similarity scores for all the genes in ssMatByGene list. If not provided these scores are computed from ssMatByGene.

... parameters for \texttt{hpSetCompSummary} if geneSSScore is not provided.

Value

A list with the following elements:

\textbf{hpoi} The original HP of interest.

\textbf{allScoreDist} The distribution of scores for all genes for the HP of interest.

\textbf{scores} The semantic similarity by gene.

\textbf{best.matches} For each gene which related HP terms best fits with the HP of interest (colnames of the elements of ssMatByGene).

\textbf{median} The median of scores.

\textbf{p.value} According to a \texttt{wilcox.test} comparing genes of interest to all the other genes.

\textbf{best.gene} Gene with the highest score among the genes of interest.

\textbf{max} Maximum score.

\textbf{score.quantiles} Quantile of the scores compared to the whole list of gene.

\textbf{adj.quant} Adjusted quantiles according Benjamini Hochberg (\texttt{link(p.adjust)}).

Author(s)

Patrice Godard

See Also

\texttt{hpGeneHeatmap, compareHPSets, hpSetCompSummary} and \texttt{hpSetCompBestMatch}

Examples

data(geneByHp, hp_descendants, package="PCAN")
data(hp_ancestors, hpDef, package="PCAN")
data(traitDef, geneDef, package="PCAN")
geneByHp <- unstack(geneByHp, entrez=hp)

############################################################
## Compute information content of each HP according to associated genes
ic <- computeHpIC(geneByHp, hp_descendants)

############################################################
## Use case: comparing a gene and a disease
omim <- "612285"
traitDef[which(traitDef$id==omim),]
entrez <- "57545"
geneDef[which(geneDef$entrez==entrez),]

## Get HP terms associated to the disease
hpOfInterest <- hpByTrait$hp[which(hpByTrait$id==omim)]

## Get HP terms associated to the gene
hpGeneListComp

```
hpByGene <- unstack(stack(geneByHp), ind~values)
geneHps <- hpByGene[[entrez]]
## HP Comparison
hpGeneResnik <- compareHPSets(
  hpSet1=names(ic), hpSet2=hpOfInterest,
  IC=ic,
  ancestors=hp_ancestors,
  method="Resnik",
  BPPARAM=SerialParam()
)
hpMatByGene <- lapply(
  hpByGene,
  function(x){
    hpGeneResnik[x, , drop=FALSE]
  }
)
resnSss <- unlist(lapply(
  hpMatByGene,
  hpSetCompSummary,
  method="bma", direction="symSim"
))
candScore <- resnSss[entrez]

###########################################
## The pathway consensus approach
## What about genes belonging to the same pathways as the candidate
data(rPath, hsEntrezByRPath, package="PCAN")
candPath <- names(hsEntrezByRPath)[which(unlist(lapply(
  hsEntrezByRPath,
  function(x) entrez %in% x
)))]
rPath[which(rPath$Pathway %in% candPath),]
rPathRes <- hpGeneListComp(
  geneList=hsEntrezByRPath[[candPath]],
  ssMatByGene = hpMatByGene,
  geneSSScore = resnSss
)

hist(
  resnSss,
  breaks=100, col="grey",
  ylim=c(0,5),
  xlab=expression(Sim[sym]),
  ylab="Density",
  main=paste("Distribution of symmetric semantic similarity scores for all the",
              length(resnSss), "genes"),
  probability=TRUE
)
toAdd <- hist(
  rPathRes$scores,
  breaks=100,
  plot=FALSE
)
for(i in 1:length(toAdd$density)){
  polygon(
    x=toAdd$breaks[c(i, i+1, i+1, i)],
    border=FALSE,
    col=gray(0.5),
    lty=1
  )
}
```
y=c(0, 0, rep(toAdd$density[i], 2)),
col="#BE000040",
border="#800000FF"
)
)
legend(
"topright",
paste0(
"Genes belonging to the ", candPath," pathway:

and with a symmetric semantic similarity score (",
sum(!is.na(rPathRes$scores)), "/",
length(rPathRes$scores), ")

p-value: ", signif(rPathRes$p.value, 2)
),
pch=15,
col="#BE000040",
bty="n",
cex=0.6
)
## Assessing the symmetric semantic similarity for each gene in the pathway
pathSss <- rPathRes$scores[which(!is.na(rPathRes$scores))]
names(pathSss) <- geneDef[match(names(pathSss), geneDef$entrez), "symbol"]
opar <- par(mar=c(7.1, 4.1, 4.1, 2.1))
barplot(
  sort(pathSss),
  las=2,
  ylab=expression(Sim[sym]),
  main=rPath[which(rPath$Pathway %in% candPath),"Pathway_name"]
)
p <- c(0.25, 0.5, 0.75, 0.95)
abline(
  h=quantile(resnSss, probs=p),
  col="#BE000000",
  lty=c(2, 1, 2, 2),
  lwd=c(2, 2, 2, 1)
)
text(
  rep(0,4),
  quantile(resnSss, probs=p),
  p,
  pos=3,
  offset=0,
  col="#BE000000",
  cex=0.6
)
legend(
"topleft",
paste0(
  "Quantiles of the distribution of symmetric semantic similarity
  scores for all the genes."
),
lty=1,
col="#BE000000",
cex=0.6
## A heatmap showing the best HP match for each gene in the pathway

geneLabels <- geneDef$symbol[which(!duplicated(geneDef$entrez))]
names(geneLabels) <- geneDef$entrez[which(!duplicated(geneDef$entrez))]
hpLabels <- hpDef$name
names(hpLabels) <- hpDef$id

hpGeneHeatmap(
    rPathRes,
    genesOfInterest=entrez,
    geneLabels=geneLabels,
    hpLabels=hpLabels,
    clustByGene=TRUE,
    clustByHp=TRUE,
    palFun=colorRampPalette(c("white", "red")),
    goiCol="blue",
    main=rPath[which(rPath$Pathway %in% candPath),"Pathway_name"]
)

###########################################
## What about genes interacting with the candidate (including itself)
data(hqStrNw, package="PCAN")

neighbors <- unique(c(
    hqStrNw$gene1[which(hqStrNw$gene2==entrez)],
    hqStrNw$gene2[which(hqStrNw$gene1==entrez)],
    entrez
))

neighRes <- hpGeneListComp(
    geneList=neighbors,
    ssMatByGene = hpMatByGene,
    geneSSScore = resnSss
)

hist(
    resnSss,
    breaks=100, col="grey",
    ylim=c(0,10),
    xlab=expression(Sim[sym]),
    ylab="Density",
    main=paste(  
        "Distribution of symmetric semantic similarity scores for all the",
        length(resnSss), "genes"
    ),
    probability=TRUE
)

toAdd <- hist(
    neighRes$scores,
    breaks=100,
    plot=FALSE
)

for(i in 1:length(toAdd$density)){
    polygon(
        x=toAdd$breaks[c(i, i+1, i+1, i)],
        y=c(0, 0, rep(toAdd$density[i], 2)),
        col="#BE000040",
        border="#800000FF"
    )
}
legend(
  "topright",
  paste0(
    "Genes interacting with ",
    geneDef[which(geneDef$entrez==entrez),"symbol"],
    ", (", entrez, ")",
    "\n and with a symmetric semantic similarity score (", sum(!is.na(neighRes$scores)), "/", length(neighRes$scores), "\n", "p-value: ", signif(neighRes$p.value, 2)
  ),
  pch=15,
  col="#BE000040",
  bty="n",
  cex=0.6
)
## Assessing the symmetric semantic similarity score for each interacting gene
neighSss <- neighRes$scores[which(!is.na(neighRes$scores))]
names(neighSss) <- geneDef[match(names(neighSss), geneDef$entrez), "symbol"]
opar <- par(mar=c(7.1, 4.1, 4.1, 2.1))
barplot(
  sort(neighSss),
  las=2,
  ylab=expression(Sim[sym]),
  main=paste0(
    "Genes interacting with ",
    geneDef[which(geneDef$entrez==entrez),"symbol"],
    ", (", entrez, ")"
  ),
)
p <- c(0.25, 0.5, 0.75, 0.95)
abline(
  h=quantile(resnSss, probs=p),
  col="#BE000000",
  lty=c(2, 1, 2, 2),
  lwd=c(2, 2, 2, 1)
)
text(
  rep(0,4),
  quantile(resnSss, probs=p),
  p,
  pos=3,
  offset=0,
  col="#BE000000",
  cex=0.6
)
hpSetCompBestMatch

Best matches between two sets of HP terms

Description

This function returns the best matches from a semantic similarity matrix.

Usage

hpSetCompBestMatch(hpSetComp, direction = c("b", "r", "c"))

Arguments

- hpSetComp: a matrix of semantic similarities between couples of HP terms
- direction: taken into account. "r": best match per row. "c": best match per column. "b" (symetric): best match for the whole matrix

Value

A data frame with the compared term, the best match and the value of the match.

Author(s)

Patrice Godard

See Also

compareHPSets and hpSetCompSummary
Examples

```r
## Prerequisite
data(geneByHp, hp_descendants, package="PCAN")
geneByHp <- unstack(geneByHp, entrez~hp)
ic <- computeHpIC(geneByHp, hp_descendants)

# Use case: comparing a gene and a disease
data(traitDef, geneDef, hp_ancestors, hpDef, package="PCAN")
omim <- "612285"
traitDef[which(traitDef$id==omim),]
entrez <- "57545"
geneDef[which(geneDef$entrez==entrez),]

## Get HP terms associated to the disease
data(hpByTrait, package="PCAN")
hpOfInterest <- hpByTrait$hp[which(hpByTrait$id==omim)]

## Get HP terms associated to the gene
hpByGene <- unstack(stack(geneByHp), ind~values)
geneHps <- hpByGene[[entrez]]

## Comparison of the two sets of HP terms
compMat <- compareHPSets(
  hpSet1=geneHps, hpSet2=hpOfInterest,
  IC=ic,
  ancestors=hp_ancestors,
  method="Resnik",
  BPPARAM=SerialParam()
)

## Get the symmetric semantic similarity score
hpSetCompSummary(compMat, method="bma", direction="symSim")
```

---

### hpSetCompSummary

**Global semantic similarity between 2 HP sets**

This function summarize the comparison of 2 sets of HP terms

#### Usage

```r
hpSetCompSummary(hpSetComp, method = c("bma", "bm", "average"),
direction = c("symSim", "r", "c"))
```

#### Arguments

- `hpSetComp`: a matrix of semantic similarities between couples of HP terms
- `method`: "bma" (Best Match Average): the average of the best matches on rows or columns (see direction param). "bm": the maximum value. "average": the average of the whole matrix.
- `direction`: taken into account only if method="bma". "r": best match per row. "c": best match per column. "symSim" (symmetric semantic similarity): average of calls with "r" and "c"
Value

A numeric value corresponding to the semantic similarity of the 2 HP sets

Author(s)

Patrice Godard

See Also

compareHPSets

Examples

```r
## Prerequisite
data(geneByHp, hp_descendants, package="PCAN")
geneByHp <- unstack(geneByHp, entrez~hp)
ic <- computeHpIC(geneByHp, hp_descendants)

###########################################
## Use case: comparing a gene and a disease
data(traitDef, geneDef, hp_ancestors, hpDef, package="PCAN")
omim <- "612285"
traitDef[which(traitDef$id==omim),]
entrez <- "57545"
geneDef[which(geneDef$entrez==entrez),]

## Get HP terms associated to the disease
data(hpByTrait, package="PCAN")
hpOfInterest <- hpByTrait$hp[which(hpByTrait$id==omim)]

## Get HP terms associated to the gene
geneByGene <- unstack(stack(geneByHp), ind~values)
geneHps <- hpByGene[[entrez]]

## Comparison of the two sets of HP terms
compMat <- compareHPSets(  
  hpSet1=geneHps, hpSet2=hpOfInterest,  
  IC=ic,  
  ancestors=hp_ancestors,  
  method="Resnik",  
  BPPARAM=SerialParam()
)

## Get the symmetric semantic similarity score
hpSetCompSummary(compMat, method="bma", direction="symSim")

bm <- hpSetCompBestMatch(compMat, "b")

hpDef[match(c(bm$compared, bm$candidate), hpDef$id),]
```

---

**hp_ancestors**

**HP ancestors (Example data)**

Description

HP terms which are ancestors of each HP term (including itself) in the Human Phenotype Ontology ([http://www.human-phenotype-ontology.org/](http://www.human-phenotype-ontology.org/)). Only descendants of 'Phenotypic abnormality' were taken into account.
hp_class

Format
A named list of 10962 character vectors.

Details
These data are used to examplify the different functions of the package. More data are available in the MultiHumanPhenoDB package.

Source
http://compbio.charite.de/hudson/job/hpo/1529/artifact/hp/hp.obo

Examples

```r
## Prerequisite
data(geneByHp, hp_descendants, package="PCAN")
geneByHp <- unstack(geneByHp, entrez~hp)
ic <- computeHpIC(geneByHp, hp_descendants)

## Compute similarity between different couples of HP terms
data(hp_ancestors, hpDef, package="PCAN")
hp1 <- "HP:0000518"
hp2 <- "HP:0030084"
hp3 <- "HP:0002119"
hp4 <- "HP:0001305"
hpDef[which(hpDef$id %in% c(hp1, hp2)), 1:2]
calchpSim(hp1, hp2, IC=ic, ancestors=hp_ancestors)
hpDef[which(hpDef$id %in% c(hp2, hp3)), 1:2]
calchpSim(hp2, hp3, IC=ic, ancestors=hp_ancestors)
hpDef[which(hpDef$id %in% c(hp2, hp4)), 1:2]
calchpSim(hp2, hp4, IC=ic, ancestors=hp_ancestors)
hpDef[which(hpDef$id %in% c(hp3, hp4)), 1:2]
calchpSim(hp3, hp4, IC=ic, ancestors=hp_ancestors)
```

---

hp_class

**HP class (Example data)**

Description
Each HP term can be of one or several classes. Classes are HP terms direct descendants of the 'Phenotypic abnormality' term.

Format
A named list of 10962 character vectors.

Details
These data are used to examplify the different functions of the package. More data are available in the MultiHumanPhenoDB package.

Source
http://compbio.charite.de/hudson/job/hpo/1529/artifact/hp/hp.obo
Examples

data(hpDef, hp_class, package="PCAN")
hp <- "HP:0100089"
hpDef[which(hpDef$id==hp),]
# This term has 2 classes:
hpDef[which(hpDef$id %in% hp_class[[hp]]),]

Description

HP terms which are descendants of each HP term (including itself) in the Human Phenotype Ontology (http://www.human-phenotype-ontology.org/). Only descendants of 'Phenotypic abnormality' were taken into account.

Format

A named list of 10962 character vectors.

Details

These data are used to examplify the different functions of the package. More data are available in the MultiHumanPhenoDB package.

Source

http://compbio.charite.de/hudson/job/hpo/1529/artifact/hp/hp.obo

Examples

#########################################################################
## Compute information content of each HP according to associated genes
#########################################################################
data(geneByHp, hp_descendants, package="PCAN")
geneByHp <- unstack(geneByHp, entrez~hp)
ic <- computeHpIC(geneByHp, hp_descendants)
hist(
ic, 
   breaks=100, col="grey",
   main="Distribution of Information Content",
   xlab="IC base on genes associated to HP"
)
hqStrNw  

**Description**

A network of human entrez gene IDs taken from the STRING database.

**Format**

A data frame of 643683 and 3 columns:

- **gene1**  Entrez gene IDs.
- **gene2**  Entrez gene IDs.
- **upstream**  TRUE if the directionality of the interaction between the 2 genes is known. In this case gene1 is upstream gene 2.

**Source**

Different resources were used in June 2 2015:

- http://string-db.org/newstring_download/protein.actions.v10/9606.protein.actions.v10.txt.gz was used to get the network of Ensembl protein IDs. Only interaction with a score greater or equal to 500 were kept.
- BioMart from http://jan2013.archive.ensembl.org/index.html was used to map Ensembl protein IDs to Ensembl gene IDs. Ensembl gene IDs were mapped to Entrez gene IDs using this resource in addition to ftp://ftp.ncbi.nih.gov/gene/DATA/gene2ensembl.gz.

**Examples**

```r
## Not run: example(hpGeneListComp)
```

hsEntrezByRPath  

**Description**

The human genes coding for proteins involved in the different Reactome pathways.

**Format**

A named list of 1345 character vectors.

**Source**

Two resources were used in June 2 2015:

- http://www.reactome.org/download/current/UniProt2Reactome.txt was used to get list of Uniprot ID associated to each pathway.
- ftp://ftp.uniprot.org/pub/databases/uniprot/current_release/knowledgebase/idmapping/by_organism/HUMAN_9606_idmapping.dat.gz was used to map Uniprot ID to Entrez gene IDs.
Examples

## Not run: example(hpGeneListComp)

---

rPath

*Reactome pathways (Example data)*

**Description**

Pathways taken from the Reactome database.

**Format**

A data frame with 1345 rows and 2 columns:

- **Pathway** Reactome ID.
- **Pathway_name** The name of the pathway.

**Source**


**Examples**

## Not run: example(hpGeneListComp)

---

traitDef

*Description of Traits (Example data)*

**Description**

Basic information about traits. Only OMIM diseases associated to at least one gene are taken into account.

**Format**

A data frame with 3675 rows and 3 columns:

- **db** Always "OMIM" here.
- **id** The trait ID (OMIM IDs here).
- **name** The name of the trait.

**Details**

These data are used to examplify the different functions of the package. More data are available in the MultiHumanPhenoDB package.

**Source**

Examples

```r
## Prerequisite
data(geneByHp, hp_descendants, package="PCAN")
geneByHp <- unstack(geneByHp, entrez~hp)
ic <- computeHpIC(geneByHp, hp_descendants)

###########################################
## Use case: comparing a gene and a disease
data(traitDef, geneDef, hp_ancestors, hpDef, package="PCAN")
omim <- "612285"
traitDef[which(traitDef$id==omim),]
entrez <- "57545"
geneDef[which(geneDef$entrez==entrez),]
## Get HP terms associated to the disease
data(hpByTrait, package="PCAN")
hpOfInterest <- hpByTrait$hp[which(hpByTrait$id==omim)]

## Get HP terms associated to the gene
hpByGene <- unstack(stack(geneByHp), ind~values)
genephps <- hpByGene[[entrez]]
## Comparison of the two sets of HP terms
compMat <- compareHPSets(
  hpSet1=genephps, hpSet2=hpOfInterest,
  IC=ic,
  ancestors=hp_ancestors,
  method="Resnik",
  BPPARAM=SerialParam()
)
## Get the symmetric semantic similarity score
hpSetCompSummary(compMat, method="bma", direction="symSim")
bm <- hpSetCompBestMatch(compMat, "b")
hpDef[match(c(bm$compared, bm$candidate), hpDef$id),]
```
Index

BiocParallelParam, 3
bmaply, 3

calcHpSim, 2, 3
compareHPSets, 2, 3, 17, 22, 24
computeHpIC, 5

geneByHp, 6
geneByTrait, 7
geneDef, 8

heatmap, 11
hp_ancestors, 24
hp_class, 25
hp_descendants, 26
hpByTrait, 9
hpDef, 10
hpGeneHeatmap, 11, 17
hpGeneListComp, 11, 16
hpSetCompBestMatch, 11, 17, 22
hpSetCompSummary, 17, 22, 23
hqStrNw, 27
hsEntrezByRPath, 27

rPath, 28

traitDef, 28

wilcox.test, 17