Package 'CellScore'

December 18, 2024

Title Tool for Evaluation of Cell Identity from Transcription Profiles

Version 1.27.0

Description The CellScore package contains functions to evaluate the cell identity of a test sample, given a cell transition defined with a starting (donor) cell type and a desired target cell type. The evaluation is based upon a scoring system, which uses a set of standard samples of known cell types, as the reference set. The functions have been carried out on a large set of microarray data from one platform (Affymetrix Human Genome U133 Plus 2.0). In principle, the method could be applied to any expression dataset, provided that there are a sufficient number of standard samples and that the data are normalized.

Suggests hgu133plus2CellScore, knitr, testthat (>= 3.0.0)

VignetteBuilder knitr

biocViews GeneExpression, Transcription, Microarray, MultipleComparison, ReportWriting, DataImport, Visualization

Imports Biobase (>= 2.39.1), graphics (>= 3.5.0), grDevices (>= 3.5.0), gplots (>= 3.0.1), lsa (>= 0.73.1), methods (>= 3.5.0), RColorBrewer(>= 1.1-2), squash (>= 1.0.8), stats (>= 3.5.0), utils(>= 3.5.0), SummarizedExperiment

Depends R (>= 4.3.0)

License GPL-3

Encoding UTF-8

LazyData true

RoxygenNote 7.2.3

NeedsCompilation no

Config/testthat/edition 3

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

git_branch devel

git_last_commit 9e8a1d2

git_last_commit_date 2024-10-29

Repository Bioconductor 3.21

Date/Publication 2024-12-18

Author Nancy Mah [aut, cre],
Katerina Taskova [aut],
Justin Marsh [aut]

Maintainer Nancy Mah <nancy.l.mah@googlemail.com>

Contents

`BarplotOnOff` *Pyramid barplot of on/off scores*

Description

This function will generate a horizontal barplot of the OnOff scores of test cell types, as defined by the eset$sub_cell_type1 column of the input dataset. Note that if the cell types as provided in the second argument (score data frame as produced by the function OnOff, are not matching the phenotype of the input dataset, the function will return an error.

Usage

```
BarplotOnOff(eset, group.score)
```

Arguments

`eset`	an ExpressionSet containing data matrices of normalized expression data, present/absent calls, a gene annotation data.frame and a phenotype data.frame.
`group.score`	a data frame with cell type specific on/off scores as generated by the OnOff function.

Value

This function returns a list of two objects, as follows:

`GroupComparisonsForPlot`
: an ordered data.frame of on/off scores,

`OnOffBarplotData`
: a data frame of marker gain/loss and aditional features, used for making the plot.

See Also

`OnOff` for details on on/off score calculations, and `hgu133plus2CellScore` for details on the specific expressionSet object that should be provided as an input.

Examples

```
## Load the expression set for the standard cell types
library(Biobase)
library(hgu133plus2CellScore) # eset.std

## Locate the external data files in the CellScore package
rdata.path <- system.file("extdata", "eset48.RData", package = "CellScore")
tsvdata.path <- system.file("extdata", "cell_change_test.tsv",
                            package = "CellScore")

if (file.exists(rdata.path) && file.exists(tsvdata.path)) {

   ## Load the expression set with normalized expressions of 48 test samples
   load(rdata.path)

   ## Import the cell change info for the loaded test samples
   cell.change <- read.delim(file= tsvdata.path, sep="\t",
                             header=TRUE, stringsAsFactors=FALSE)

   ## Combine the standards and the test data
   eset <- combine(eset.std, eset48)

   ## Generate a marker list
   group.OnOff <- OnOff(eset, cell.change, out.put="marker.list")

   ## Calculate on/off score for individual samples
   individ.OnOff <- OnOff(eset, cell.change, out.put="individual")

   ## Plot pyramid bar plot of on/off scores
   BarplotOnOff(eset, group.OnOff$scores)
}
```

BoxplotCellScore *Boxplot of the CellScore values across test samples*

Description

This function will plot a boxplot of the CellScore values for each selected transition (defined in the cell.change data frame). The function will only plot the scores for the test samples of valid subtypes (as annotated by cellscore$sub_cell_type1). Scores for the standards are not included. Note that if a subtype is specified by two different transitions, the coresponding scores will be plotted in both transitions.

Usage

```
BoxplotCellScore(cellscore, cell.change)
```

Arguments

cellscore	a data.frame of CellScore values as calculated by CellScore().
cell.change	a data frame containing three columns, one for the start (donor) test and target cell type. Each row of the data frame describes one transition from the start to a target cell type.

Value

Invisibly, it returns list of the CellScore values by groups (in the same order as on the plot)

See Also

CellScore for details on CellScore calculation.

Examples

```
## Load the expression set for the standard cell types
library(Biobase)
library(hgu133plus2CellScore) # eset.std

## Locate the external data files in the CellScore package
rdata.path <- system.file("extdata", "eset48.RData", package = "CellScore")
tsvdata.path <- system.file("extdata", "cell_change_test.tsv",
                            package = "CellScore")

if (file.exists(rdata.path) && file.exists(tsvdata.path)) {

   ## Load the expression set with normalized expressions of 48 test samples
   load(rdata.path)

   ## Import the cell change info for the loaded test samples
   cell.change <- read.delim(file= tsvdata.path, sep="\t",
                            header=TRUE, stringsAsFactors=FALSE)
```

```
  ## Combine the standards and the test data
  eset <- combine(eset.std, eset48)

  ## Generate cosine similarity for the combined data
  ## NOTE: May take 1-2 minutes on the full eset object
  ## so we subset it for 4 cell types
  pdata <- pData(eset)
  sel.samples <- pdata$general_cell_type %in% c("ESC", "EC", "FIB", "KER")
  eset.sub <- eset[, sel.samples]
  cs <- CosineSimScore(eset.sub, cell.change, iqr.cutoff=0.1)

  ## Generate the on/off scores for the combined data
  individ.OnOff <- OnOff(eset.sub, cell.change, out.put="individual")

  ## Generate the CellScore values for all samples
 cellscore <- CellScore(data=eset.sub, transitions=cell.change, scores.onoff=individ.OnOff$scores,
                        scores.cosine=cs$cosine.samples)

  ## Make the boxplot of CellScore values
  BoxplotCellScore(cellscore, cell.change)
}
```

`CellScore` *CellScore evaluates the identity of cells undergoing cell type transition*

Description

This function will calculate the CellScore (summary score) for a cell that is undergoing a transition in cell identity from a starting cell type to a target cell type. ‘transitions‘ is mandatory, and either ‘data‘ or all three of ‘scores.onoff‘, ‘scores.cosine‘ and ‘pdata‘ are as well. If you provide ‘data‘, then ‘scores.onoff‘, ‘scores.cosine‘ and ‘pdata‘ will override the respective calculations only.

Usage

```
CellScore(
  transitions,
  data = NULL,
  scores.onoff = NULL,
  scores.cosine = NULL,
  pdata = NULL
)
```

Arguments

`transitions`	a data frame containing three columns, one for the start (donor) test and target cell type. Each row of the data. frame describes one transition from the start to a target cell type.

data	a SummarizedExperiment or ExpressionSet containing data matrices of normalized expression data, present/absent calls, a gene annotation data frame and a phenotype data frame.
scores.onoff	a data.frame of OnOff Scores for all samples in the expression matrix as generated by the function OnOff().
scores.cosine	a numeric matrix of cosine similarity between general groups, subgroups and individual samples as calculated by the function CosineSimScore().
pdata	a data frame with samples as rows, variables as columns.

Value

The function returns a data frame with 29 columns and M*N rows, where M is the number of unqiue start and target cell types pairs listed in the cell.change argument, while N is the number of all samples in the input dataset eset. The columns include sample phenotype features and all score (components), including the on/off score, cosine similarity and CellScore.

See Also

CosineSimScore, OnOff for details on specfic score calculations, and hgu133plus2CellScore for details on the specific expressionSet object that shoud be provided as an input.

Examples

```
## Load the expression set for the standard cell types
library(Biobase)
library(hgu133plus2CellScore) # eset.std

## Locate the external data files in the CellScore package
rdata.path <- system.file("extdata", "eset48.RData", package = "CellScore")
tsvdata.path <- system.file("extdata", "cell_change_test.tsv",
                            package = "CellScore")

if (file.exists(rdata.path) && file.exists(tsvdata.path)) {

   ## Load the expression set with normalized expressions of 48 test samples
   load(rdata.path)

   ## Import the cell change info for the loaded test samples
   cell.change <- read.delim(file= tsvdata.path, sep="\t",
                            header=TRUE, stringsAsFactors=FALSE)

   ## Combine the standards and the test data
   eset <- combine(eset.std, eset48)

   ## Generate cosine similarity for the combined data
   ## NOTE: May take 1-2 minutes on the full eset object
   ## so we subset it for 4 cell types
   pdata <- pData(eset)
   sel.samples <- pdata$general_cell_type %in% c("ESC", "EC", "FIB", "KER")
   eset.sub <- eset[, sel.samples]
   cs <- CosineSimScore(eset.sub, cell.change, iqr.cutoff=0.1)
```

```
  ## Generate the on/off scores for the combined data
  individ.OnOff <- OnOff(eset.sub, cell.change, out.put="individual")

  ## Generate the CellScore values for all samples
 cellscore <- CellScore(cell.change, data = eset.sub, scores.onoff = individ.OnOff$scores,
                        scores.cosine = cs$cosine.samples)
}
```

`CellScoreReport` *Generate a CellScore report*

Description

This function will generates a CellScore report for each study and transition that can be saved as a pdf. The report includes: 1) scatterplot of the donor-like and target-like scores of relevant test samples and the standards; 2) a density plot of the test and standard cellscores; 3) a rugplot of the cellscores, focussing on the test samples; 4) a heatmap of the OnOff Marker genes for all standards and test samples.

Usage

```
CellScoreReport(cellscore, cell.change, marker.genes, eset)
```

Arguments

`cellscore`	a data.frame of CellScore values, as calculated by CellScore().
`cell.change`	a data frame containing three columns, one for the start (donor) test and target cell type. Each row of the data frame describes one transition from the start to a target cell type.
`marker.genes`	a data.frame of marker genes as generated by function OnOff()
`eset`	an ExpressionSet containing data matrices of normalized expression data, present/absent calls, a gene annotation data.frame and a phenotype data.frame.

Value

This function outputs the plots on the active graphical device and returns invisibly NULL.

See Also

`CellScore` for details on CellScore, and `hgu133plus2CellScore` for details on the specific ExpressionSet object that shoud be provided as an input.

Examples

```
## Load the expression set for the standard cell types
library(Biobase)
library(hgu133plus2CellScore) # eset.std

## Locate the external data files in the CellScore package
rdata.path <- system.file("extdata", "eset48.RData", package = "CellScore")
tsvdata.path <- system.file("extdata", "cell_change_test.tsv",
                            package = "CellScore")

if (file.exists(rdata.path) && file.exists(tsvdata.path)) {

   ## Load the expression set with normalized expressions of 48 test samples
   load(rdata.path)

   ## Import the cell change info for the loaded test samples
   cell.change <- read.delim(file= tsvdata.path, sep="\t",
                            header=TRUE, stringsAsFactors=FALSE)

   ## Combine the standards and the test data
   eset <- combine(eset.std, eset48)

   ## Generate cosine similarity for the combined data
   ## NOTE: May take 1-2 minutes on the full eset object
   ## so we subset it for 4 cell types
   pdata <- pData(eset)
   sel.samples <- pdata$general_cell_type %in% c("ESC", "EC", "FIB", "KER",
                "ASC", "NPC", "MSC", "iPS", "piPS")
   eset.sub <- eset[, sel.samples]
   cs <- CosineSimScore(eset.sub, cell.change, iqr.cutoff=0.1)

   ## Generate the on/off scores for the combined data
   individ.OnOff <- OnOff(eset.sub, cell.change, out.put="individual")

   ## Generate the CellScore values for all samples
  cellscore <- CellScore(data=eset.sub, transitions=cell.change, scores.onoff=individ.OnOff$scores,
                         scores.cosine=cs$cosine.samples)

   ## Generate the group on/off scores for the combined data
   group.OnOff <- OnOff(eset.sub, cell.change, out.put="marker.list")

   ## Make a report and save it the current working directory
   pdf("TestReport.pdf", width=8, height=12)
   CellScoreReport(cellscore, cell.change, group.OnOff$markers, eset.sub)
   dev.off()
}
```

`CosineSimScore` *Cosine similarity score*

Description

This function calculates the cosine similarity for cell transitions.

Usage

```
CosineSimScore(eset, cell.change, iqr.cutoff = 0.1)
```

Arguments

`eset`	an ExpressionSet containing data matrices of normalized expression data, present/absent calls, a gene annotation data frame and a phenotype data frame.
`cell.change`	a data frame containing three columns, one for the start (donor) test and target cell type. Each row of the data frame describes one transition from the start to a target cell type.
`iqr.cutoff`	set the threshold for top most variable genes which should be included for the cosine similarity calculation. Default is the top 10 genes, expressed as a fraction. All samples that are annotated as standards will be used for the iqr calculation.

Value

This function returns a list of five objects, as follows:

pdataSub the phenotype data frame describing the standard samples

esetSub.IQR the expression value matrix, as filtered by IQR threshold

cosine.general.groups a numeric matrix of cosine similarity between the centroids of all groups defined by eset@general_cell_types

cosine.subgroups a numeric matrix of cosine similarity between the centroids of all gsubroups defined by eset@sub_cell_types1

cosine.samples a numeric matrix of cosine similarity between general groups, subgroups and individual samples.

See Also

`hgu133plus2CellScore` for details on the specific ExpressionSet object that shoud be provided as an input.

Examples

```
## Load the expression set for the standard cell types
library(Biobase)
library(hgu133plus2CellScore) # eset.std

## Locate the external data files in the CellScore package
rdata.path <- system.file("extdata", "eset48.RData", package = "CellScore")
tsvdata.path <- system.file("extdata", "cell_change_test.tsv",
                            package = "CellScore")

if (file.exists(rdata.path) && file.exists(tsvdata.path)) {
```

```
  ## Load the expression set with normalized expressions of 48 test samples
  load(rdata.path)

  ## Import the cell change info for the loaded test samples
  cell.change <- read.delim(file= tsvdata.path, sep="\t",
                            header=TRUE, stringsAsFactors=FALSE)

  ## Combine the standards and the test data
  eset <- combine(eset.std, eset48)

  ## Generate cosine similarity for the combined data
  ## NOTE: May take 1-2 minutes on the full eset object
  ## so we subset it for 4 cell types
  pdata <- pData(eset)
  sel.samples <- pdata$general_cell_type %in% c("ESC", "EC", "FIB", "KER")
  eset.sub <- eset[, sel.samples]
  cs <- CosineSimScore(eset.sub, cell.change, iqr.cutoff=0.1)
}
```

`extractTransitions` *Extract scores for given cell transitions*

Description

This function extracts the values of the CellScore for all the test samples of a given set of (valid) cell transition. While it can be used as standalone, it serves as an internal function for several other CellScore functions.

Usage

```
extractTransitions(cellscore, cell.change)
```

Arguments

`cellscore`	a data.frame of CellScore values as calculated by the function CellScore().
`cell.change`	a data frame containing three columns, one for the start (donor) test and target cell type. Each row of the data. frame describes one transition from the start to a target cell type.

Value

This function returns a data frame with the same columns as the input data frame cellscore, extended with additional column that is used as a single identifier of each valid cell transition. Technically, the output is subselection of the input data frame.

See Also

`CellScore` for details on CellScore calcualtion.

Examples

```
## Load the expression set for the standard cell types
library(Biobase)
library(hgu133plus2CellScore) # eset.std

## Locate the external data files in the CellScore package
rdata.path <- system.file("extdata", "eset48.RData", package = "CellScore")
tsvdata.path <- system.file("extdata", "cell_change_test.tsv",
                            package = "CellScore")

if (file.exists(rdata.path) && file.exists(tsvdata.path)) {

   ## Load the expression set with normalized expressions of 48 test samples
   load(rdata.path)

   ## Import the cell change info for the loaded test samples
   cell.change <- read.delim(file= tsvdata.path, sep="\t",
                             header=TRUE, stringsAsFactors=FALSE)

   ## Combine the standards and the test data
   eset <- combine(eset.std, eset48)

   ## Generate cosine similarity for the combined data
   ## NOTE: May take 1-2 minutes on the full eset object
   ## so we subset it for 4 cell types
   pdata <- pData(eset)
   sel.samples <- pdata$general_cell_type %in% c("ESC", "EC", "FIB", "KER",
                "ASC", "NPC", "MSC", "iPS", "piPS")
   eset.sub <- eset[, sel.samples]
   cs <- CosineSimScore(eset.sub, cell.change, iqr.cutoff=0.1)

   ## Generate the on/off scores for the combined data
   individ.OnOff <- OnOff(eset.sub, cell.change, out.put="individual")

   ## Generate the CellScore values for all samples
  cellscore <- CellScore(data=eset.sub, transitions=cell.change, scores.onoff=individ.OnOff$scores,
                         scores.cosine=cs$cosine.samples)

   ## Extract the scores for the transitions given in cell.change
   cellscore.cc <- extractTransitions(cellscore, cell.change)

   ## View the sub_cell_type1 in the extracted object, it should be the same
   ## as the test cell types named in cell.change
   table(cellscore.cc$sub_cell_type1)
}
```

---

heatmapOnOffMarkers *heatmapOnOffMarkers*

Description

This function is called by CellScoreReport to make a heatmap of the standards (donor and target) marker genes and the test samples for the defined transition, as generated by the OnOff() function. Gene symbols are not plotted as this is only intended as an overview of marker expression in test samples.

Usage

```
heatmapOnOffMarkers(test.data, markergenes, pdata, calls)
```

Arguments

`test.data`	a data.frame of CellScore values as calculated by CellScore(), for only a group of test samples.
`markergenes`	a data.frame of marker genes, as calculated by OnOff().
`pdata`	a data.frame containing the phenotype of the expression dataset.
`calls`	a matrix containing the present/absent calls where genes are in rows and samples in columns.

Value

This function returns invisibly the visualised binary matrix of the absence/presence of the cell type markers (rows) across the samples (columns) in the given study.

See Also

`CellScore` for details on CellScore, and `hgu133plus2CellScore` for details on the specific ExpressionSet object that shoud be provided as an input.

Examples

```
## Load the expression set for the standard cell types
library(Biobase)
library(hgu133plus2CellScore) # eset.std

## Locate the external data files in the CellScore package
rdata.path <- system.file("extdata", "eset48.RData", package = "CellScore")
tsvdata.path <- system.file("extdata", "cell_change_test.tsv",
                            package = "CellScore")

if (file.exists(rdata.path) && file.exists(tsvdata.path)) {

   ## Load the expression set with normalized expressions of 48 test samples
   load(rdata.path)

   ## Import the cell change info for the loaded test samples
   cell.change <- read.delim(file= tsvdata.path, sep="\t",
                            header=TRUE, stringsAsFactors=FALSE)

   ## Combine the standards and the test data
```

```
  eset <- combine(eset.std, eset48)

  ## Generate cosine similarity for the combined data
  ## NOTE: May take 1-2 minutes on the full eset object
  ## so we subset it for 4 cell types
  pdata <- pData(eset)
  sel.samples <- pdata$general_cell_type %in% c("ESC", "EC", "FIB", "KER",
                "ASC", "NPC", "MSC", "iPS", "piPS")
  eset.sub <- eset[, sel.samples]
  cs <- CosineSimScore(eset.sub, cell.change, iqr.cutoff=0.1)

  ## Generate the on/off scores for the combined data
  individ.OnOff <- OnOff(eset.sub, cell.change, out.put="individual")

  ## Generate the CellScore values for all samples
 cellscore <- CellScore(data=eset.sub, transitions=cell.change, scores.onoff=individ.OnOff$scores,
                        scores.cosine=cs$cosine.samples)
  ## Get the CellScore fvalues rom valid transitions defined by cell.change
  ## table
  plot.data <- extractTransitions(cellscore, cell.change)

  ## Define a plot group variable
  plot.data$plot_group <- paste(plot.data$experiment_id,
                                plot.data$cxkey.subcelltype, sep="_")
  ## Sort the scores 1) by target 2) by donor 3) by study
  plot.data.ordered <- plot.data[order(plot.data$target,
                                       plot.data$donor_tissue,
                                       plot.data$experiment_id), ]
## How many plot_groups are there?
table(plot.data$plot_group)

  ## pick one plot_group to plot
  group <- unique(plot.data$plot_group)[4]

  ## Select scores for only one plot group
  test.data <- plot.data.ordered[plot.data.ordered$plot_group %in% group, ]

  ## Generate the group on/off scores for the combined data
  group.OnOff <- OnOff(eset.sub, cell.change, out.put="marker.list")

  calls <- assayDataElement(eset.sub, "calls")
 rownames(calls) <- if("feature_id" %in% names(fData(eset.sub))) { fData(eset.sub)[, "feature_id"] } else { fData

  ## Plot
  heatmapOnOffMarkers(test.data, group.OnOff$markers, pData(eset.sub),
                      calls)
}
```

OnOff *On/off score*

Description

This function calculates the on/off score for cell transitions. The score takes into account the cell type spcefific and most variable portion of the detected transcriptome. It can be calculated for a sample or group of samples representing specific (standard or engineered) cell type.

Usage

```
OnOff(
  eset,
  cell.change,
  out.put = c("marker.list", "individual"),
  min.diff.cutoff = 0.8,
  test.cutoff = 0.95
)
```

Arguments

`eset`	an ExpressionSet containing data matrices of normalized expression data, present/absent calls, a gene annotation data frame and a phenotype data frame.
`cell.change`	a data frame containing three columns, one for the start (donor) test and target cell type. Each row of the data frame describes one transition from the start to a target cell type.
`out.put`	a character flag with two possible values, "marker.list" and "individual". The former means the on/off scores will be aggregated accross cell groups and also the marker genes for each cell transition (in cell.change) will be calculated, while the latter will generate the on/off scores for all individual samples.
`min.diff.cutoff`	a real number that represents the minimum difference between the fraction of present calls in donor vs target (in the standards), in order to define the markers for a given cell transition. Default is 0.8.
`test.cutoff`	a real number in (0, 1] that is the minimum fraction of present calls in a test sample/group to decide if a gene is present in a test sample/group. Default is stringently set at 0.95.

Value

This function returns a list of two objects, as follows:

`scores`	a data.frame of on/off scores for each cell group given in cell.change(out.put="marker.list") or for each individual sample (out.put="idividual")
`markers`	a list of marker genes for the selected cell transitions in cell.change (out.put="marker.list") or NULL (out.put="individual")

See Also

`hgu133plus2CellScore` for details on the specific ExpressionSet object that shoud be provided as an input.

Examples

```
## Load the expression set for the standard cell types
library(Biobase)
library(hgu133plus2CellScore) # eset.std

## Locate the external data files in the CellScore package
rdata.path <- system.file("extdata", "eset48.RData", package = "CellScore")
tsvdata.path <- system.file("extdata", "cell_change_test.tsv",
                            package = "CellScore")

if (file.exists(rdata.path) && file.exists(tsvdata.path)) {

   ## Load the expression set with normalized expressions of 48 test samples
   load(rdata.path)

   ## Import the cell change info for the loaded test samples
   cell.change <- read.delim(file= tsvdata.path, sep="\t",
                             header=TRUE, stringsAsFactors=FALSE)

   ## Combine the standards and the test data
   eset <- combine(eset.std, eset48)

   ## Generate a marker list
   group.OnOff <- OnOff(eset, cell.change, out.put="marker.list")

   ## Calculate the on/off scores for individual samples
   individ.OnOff <- OnOff(eset, cell.change, out.put="individual")
}
```

PcaStandards *PCA plot on the most variable portion of the standard expression dataset*

Description

This function will generate a principal component analysis (PCA) plot of the IQR-filtered expression values that were used to generate the cosine similarity scores.

Usage

```
PcaStandards(
  label,
  label.name,
  exps,
  text.label = NULL,
 col.palette = c("blue", "magenta", "green", "red", "goldenrod", "mediumslateblue",
  "olivedrab", "navyblue", "plum", "tomato", "thistle", "limegreen", "burlywood4",
    "cornflowerblue", "deeppink", "chartreuse", "forestgreen", "darkslateblue",
  "blueviolet", "gray50", "darkorange", "black", "lightsalmon4", "mediumseagreen",
```

```
    "palegreen4", "palevioletred4", "peachpuff4", "plum4", "mediumspringgreen",
   "darkred", "khaki4", "lawngreen", "lightseagreen", "orange", "orchid3", "sienna4",
    "snow4", "turquoise3", "wheat3", "goldenrod2",
     "darkorange3")
)
```

Arguments

label	vector to be used for the point colours
label.name	name of the label
exps	an expression matrix of the IQR-filtered values as obtained by the function CosineSimScore().
text.label	a vector of characters to label each point.
col.palette	a vector of colours to be used. There are 41 default colours.

Value

The function will plot two panels, a PCA plot on the left and a legend on the right. This is to accommodate that fact that the cell types names are NOT abbreviated and the legend might not fit in the plot area.

See Also

CosineSimScore for details on cosine similarity calculation.

Examples

```
## Load the expression set for the standard cell types
library(Biobase)
library(hgu133plus2CellScore) # eset.std

## Locate the external data files in the CellScore package
rdata.path <- system.file("extdata", "eset48.RData", package = "CellScore")
tsvdata.path <- system.file("extdata", "cell_change_test.tsv",
                            package = "CellScore")

if (file.exists(rdata.path) && file.exists(tsvdata.path)) {

   ## Load the expression set with normalized expressions of 48 test samples
   load(rdata.path)

   ## Import the cell change info for the loaded test samples
   cell.change <- read.delim(file= tsvdata.path, sep="\t",
                             header=TRUE, stringsAsFactors=FALSE)

   ## Combine the standards and the test data
   eset <- combine(eset.std, eset48)

   ## Generate cosine similarity for the combined data
   ## NOTE: May take 1-2 minutes on the full eset object
```

```
   ## so we subset it for 4 cell types
   pdata <- pData(eset)
   sel.samples <- pdata$general_cell_type %in% c("ESC", "EC", "FIB", "KER")
   eset.sub <- eset[, sel.samples]
   cs <- CosineSimScore(eset.sub, cell.change, iqr.cutoff=0.1)

   PcaStandards(cs$pdataSub$experiment_id, "Experiment ID", cs$esetSub.IQR)
}
```

`PlotCosineSimHeatmap` *Plot heatmap of the cosine similarity score*

Description

This function plots a triangular heatmap of the cosine similarity scores.

Usage

```
PlotCosineSimHeatmap(
  data,
  desc = "xx",
  width = 20,
  height = 20,
  x = -30,
  y = 3
)
```

Arguments

`data`	a data.frame of cosine similarity scores, as generated by the function CosineSimScore().
`desc`	a single character, with description for the file name. Suggested are "general.groups", "subgroups", and "samples".
`width`	the width of the output pdf, in inches.
`height`	the height of the output pdf, in inches.
`x`	the x-position of the heatmap legend. It may be necessary to change the value to position the legend in a suitable place on the plot.
`y`	the y-position of the heatmap legend. It may be necessary to change the value to position the legend in a suitable place on the plot.

Value

This function will print a pdf of the cosine similarity scores in the current working directory.

See Also

`CosineSimScore` for details on cosine similarity calculation.

Examples

```
## Load the expression set for the standard cell types
library(Biobase)
library(hgu133plus2CellScore) # eset.std

## Locate the external data files in the CellScore package
rdata.path <- system.file("extdata", "eset48.RData", package = "CellScore")
tsvdata.path <- system.file("extdata", "cell_change_test.tsv",
                            package = "CellScore")

if (file.exists(rdata.path) && file.exists(tsvdata.path)) {

   ## Load the expression set with normalized expressions of 48 test samples
   load(rdata.path)

   ## Import the cell change info for the loaded test samples
   cell.change <- read.delim(file= tsvdata.path, sep="\t",
                             header=TRUE, stringsAsFactors=FALSE)

   ## Combine the standards and the test data
   eset <- combine(eset.std, eset48)

   ## Generate cosine similarity for the combined data
   ## NOTE: May take 1-2 minutes on the full eset object,
   ## so we subset it for 4 cell types
   pdata <- pData(eset)
   sel.samples <- pdata$general_cell_type %in% c("ESC", "EC", "FIB", "KER",
                "ASC", "NPC", "MSC")
   eset.sub <- eset[, sel.samples]
   cs <- CosineSimScore(eset.sub, cell.change, iqr.cutoff=0.1)

   ## Generate pdf of cosine similarity heatmap in the working directory
   PlotCosineSimHeatmap(cs$cosine.general.groups, "general groups",
                        width=7, height=7, x=-3.5, y=1)
}
```

RugplotCellScore *RugplotCellScore*

Description

This function will plot a rugplot of all CellScore values for each transition selected in the cell.change data frame. The function will only plot the scores for the test samples (annotated by the cellscore$column sub_cell_type1). Standards are not included. Samples are coloured by a secondary property, which must be a single column in the cellscore data frame.

Usage

```
RugplotCellScore(cellscore, cell.change, colour.by = NULL)
```

Arguments

cellscore	a data.frame of CellScore values as calculated by CellScore().
cell.change	a data frame containing three columns, one for the start (donor) test and target cell type. Each row of the data. frame describes one transition from the start to a target cell type.
colour.by	the name of the column in the cellscore argument that contains the secondary property.

Value

This function outputs the plot on the active graphical device and returns invisibly NULL.

See Also

CellScore for details on CellScore.

Examples

```
## Load the expression set for the standard cell types
library(Biobase)
library(hgu133plus2CellScore) # eset.std

## Locate the external data files in the CellScore package
rdata.path <- system.file("extdata", "eset48.RData", package = "CellScore")
tsvdata.path <- system.file("extdata", "cell_change_test.tsv",
                            package = "CellScore")

if (file.exists(rdata.path) && file.exists(tsvdata.path)) {

   ## Load the expression set with normalized expressions of 48 test samples
   load(rdata.path)

   ## Import the cell change info for the loaded test samples
   cell.change <- read.delim(file= tsvdata.path, sep="\t",
                            header=TRUE, stringsAsFactors=FALSE)

   ## Combine the standards and the test data
   eset <- combine(eset.std, eset48)

   ## Generate cosine similarity for the combined data
   ## NOTE: May take 1-2 minutes on the full eset object
   ## so we subset it for 4 cell types
   pdata <- pData(eset)
   sel.samples <- pdata$general_cell_type %in% c("ESC", "EC", "FIB", "KER",
                "ASC", "NPC", "MSC", "iPS", "piPS")
   eset.sub <- eset[, sel.samples]
   cs <- CosineSimScore(eset.sub, cell.change, iqr.cutoff=0.1)

   ## Generate the on/off scores for the combined data
   individ.OnOff <- OnOff(eset.sub, cell.change, out.put="individual")
```

```
  ## Generate the CellScore values for all samples
 cellscore <- CellScore(data=eset.sub, transitions=cell.change, scores.onoff=individ.OnOff$scores,
                        scores.cosine=cs$cosine.samples)

  ## Rugplot of CellScore, colour samples by transition induction method
  RugplotCellScore(cellscore, cell.change,
                   "transition_induction_method")
}
```

---

rugplotDonorTargetTest *rugplotDonorTargetTest*

Description

This function is called by CellScoreReport to make a rugplot showing the CellScore of all test samples, in relation to the standards. Donor and target individual CellScore values are plotted in one horizontal lane, then test CellScore values are are in another horizontal lane. Z-score cutoffs based on the target standards are shown as dashed vertical lines.

Usage

```
rugplotDonorTargetTest(test.data, cellscore)
```

Arguments

test.data	a data.frame of CellScore values as calculated by CellScore(), for only plot group of test samples.
cellscore	a data.frame of CellScore values as calculated by CellScore().

Value

This function outputs the plot on the active graphical device and returns invisibly NULL.

See Also

CellScore for details on CellScore.

Examples

```
## Not run:
## Load the expression set for the standard cell types
library(Biobase)
library(hgu133plus2CellScore) # eset.std

## Locate the external data files in the CellScore package
rdata.path <- system.file("extdata", "eset48.RData", package = "CellScore")
tsvdata.path <- system.file("extdata", "cell_change_test.tsv",
                            package = "CellScore")
```

```
if (file.exists(rdata.path) && file.exists(tsvdata.path)) {

   ## Load the expression set with normalized expressions of 48 test samples
   load(rdata.path)

   ## Import the cell change info for the loaded test samples
   cell.change <- read.delim(file= tsvdata.path, sep="\t",
                             header=TRUE, stringsAsFactors=FALSE)

   ## Combine the standards and the test data
   eset <- combine(eset.std, eset48)

   ## Generate the on/off scores for the combined data
   individ.OnOff <- OnOff(eset, cell.change, out.put="individual")

   ## Generate cosine similarity for the combined data
   ## NOTE: May take 1-2 minutes on the full eset object
   cs <- CosineSimScore(eset, cell.change, iqr.cutoff=0.05)

   ## Generate the CellScore values for all samples
  cellscore <- CellScore(data=eset, transitions=cell.change, scores.onoff=individ.OnOff$scores,
                         scores.cosine=cs$cosine.samples)
   ## Get the CellScore fvalues rom valid transitions defined by cell.change
   ## table
   plot.data <- extractTransitions(cellscore, cell.change)

   ## Define a plot group variable
   plot.data$plot_group <- paste(plot.data$experiment_id,
                                 plot.data$cxkey.subcelltype, sep="_")
   ## Sort the scores 1) by target 2) by donor 3) by study
   plot.data.ordered <- plot.data[order(plot.data$target,
                                        plot.data$donor_tissue,
                                        plot.data$experiment_id), ]
   ## How many plot_groups are there?
   table(plot.data$plot_group)

   ## pick one plot_group to plot
   group <- unique(plot.data$plot_group)[4]

  ## Select scores for only one plot group
  test.data <- plot.data.ordered[plot.data.ordered$plot_group %in% group, ]

  ## Plot
  rugplotDonorTargetTest(test.data, cellscore)

}

## End(Not run)
```

`ScatterplotCellScoreComponents` *Scatterplot of the the donor-like and target-like scores*

Description

This function will plot the components of the CellScore, namely the donor- like and the target-like scores. The function will only plot the scores for the test samples (annotated by the cellscore$column sub_cell_type1). Standards are not included.

Usage

```
ScatterplotCellScoreComponents(cellscore, cell.change, index.plot = FALSE)
```

Arguments

`cellscore`	a data.frame of CellScore values as calculated by CellScore()
`cell.change`	a data.frame with 3 columns: start cell type, test cell type, target cell type
`index.plot`	a logical variable, with TRUE meaning sample index should be plotted for easy identification of spots. Default is FALSE. This is useful if you want to see where the samples are located on the plot.

Value

This function outputs the plot on the active graphical device and returns invisibly NULL.

See Also

`CellScore` for details on CellScore.

Examples

```
## Load the expression set for the standard cell types
library(Biobase)
library(hgu133plus2CellScore) # eset.std

## Locate the external data files in the CellScore package
rdata.path <- system.file("extdata", "eset48.RData", package = "CellScore")
tsvdata.path <- system.file("extdata", "cell_change_test.tsv",
                            package = "CellScore")

if (file.exists(rdata.path) && file.exists(tsvdata.path)) {

   ## Load the expression set with normalized expressions of 48 test samples
   load(rdata.path)

   ## Import the cell change info for the loaded test samples
   cell.change <- read.delim(file= tsvdata.path, sep="\t",
                            header=TRUE, stringsAsFactors=FALSE)
```

```
  ## Combine the standards and the test data
  eset <- combine(eset.std, eset48)

  ## Generate cosine similarity for the combined data
  ## NOTE: May take 1-2 minutes on the full eset object
  ## so we subset it for 4 cell types
  pdata <- pData(eset)
  sel.samples <- pdata$general_cell_type %in% c("ESC", "EC", "FIB", "KER",
               "ASC", "NPC", "MSC", "iPS", "piPS")
  eset.sub <- eset[, sel.samples]
  cs <- CosineSimScore(eset.sub, cell.change, iqr.cutoff=0.1)

  ## Generate the on/off scores for the combined data
  individ.OnOff <- OnOff(eset.sub, cell.change, out.put="individual")

  ## Generate the CellScore values for all samples
 cellscore <- CellScore(data=eset.sub, transitions=cell.change, scores.onoff=individ.OnOff$scores,
                        scores.cosine=cs$cosine.samples)

  ## Make the scaterplot of CellScore components
  ScatterplotCellScoreComponents(cellscore, cell.change, FALSE)
}
```

`scatterplotDonorTargetTest` *scatterplotDonorTargetTest*

Description

This function is called by CellScoreReport to make a scatterplot of test and standard samples (donor and target).

Usage

```
scatterplotDonorTargetTest(test.data, cellscore, index.plot = FALSE)
```

Arguments

`test.data`	a data.frame of CellScore values as calculated by CellScore(), for a group of test samples.
`cellscore`	a data.frame of CellScore values as calculated by CellScore().
`index.plot`	a logical variable, with TRUE meaning sample index should be plotted for easy identification of spots. Default is FALSE.

Value

This function outputs the plot on the active graphical device and returns invisibly NULL.

Examples

```
## Not run:
## Load the expression set for the standard cell types
library(Biobase)
library(hgu133plus2CellScore) # eset.std

## Locate the external data files in the CellScore package
rdata.path <- system.file("extdata", "eset48.RData", package = "CellScore")
tsvdata.path <- system.file("extdata", "cell_change_test.tsv",
                            package = "CellScore")

if (file.exists(rdata.path) && file.exists(tsvdata.path)) {

   ## Load the expression set with normalized expressions of 48 test samples
   load(rdata.path)

   ## Import the cell change info for the loaded test samples
   cell.change <- read.delim(file= tsvdata.path, sep="\t",
                            header=TRUE, stringsAsFactors=FALSE)

   ## Combine the standards and the test data
   eset <- combine(eset.std, eset48)

   ## Generate the on/off scores for the combined data
   individ.OnOff <- OnOff(eset, cell.change, out.put="individual")

   ## Generate cosine similarity for the combined data
   ## NOTE: May take 1-2 minutes on the full eset object
   cs <- CosineSimScore(eset, cell.change, iqr.cutoff=0.05)

   ## Generate the CellScore values for all samples
  cellscore <- CellScore(data=eset, transitions=cell.change, scores.onoff=individ.OnOff$scores,
                         scores.cosine=cs$cosine.samples)
   ## Get the CellScore fvalues rom valid transitions defined by cell.change
   ## table
   plot.data <- extractTransitions(cellscore, cell.change)

   ## Define a plot group variable
   plot.data$plot_group <- paste(plot.data$experiment_id,
                                 plot.data$cxkey.subcelltype, sep="_")
   ## Sort the scores 1) by target 2) by donor 3) by study
   plot.data.ordered <- plot.data[order(plot.data$target,
                                        plot.data$donor_tissue,
                                        plot.data$experiment_id), ]

   ## How many plot_groups are there?
   table(plot.data$plot_group)

   ## pick one plot_group to plot
   group <- unique(plot.data$plot_group)[4]

   ## Select scores for only one plot group
```

```
  test.data <- plot.data.ordered[plot.data.ordered$plot_group %in% group, ]

  ## save current graphical parameters
 old.par <- par(no.readonly=TRUE)

  ## Plot: this will plot a 2-paneled plot
 par(mfrow=c(1,2))
  scatterplotDonorTargetTest(test.data, cellscore, FALSE)

  ## Reset graphical parameters
  par(old.par)

}

## End(Not run)
```

Index