Package ‘FlowSOM’

October 17, 2020

Version 1.20.0

Date 2019-12-04

Title Using self-organizing maps for visualization and interpretation of cytometry data

Author Sofie Van Gassen, Britt Callebaut and Yvan Saeys

Maintainer Sofie Van Gassen <sofie.vangassen@ugent.be>

Depends R (>= 3.2), igraph

Imports stats, utils, flowCore, ConsensusClusterPlus, BiocGenerics, tsne, CytoML, flowWorkspace, XML, RColorBrewer

Suggests BiocStyle

Description FlowSOM offers visualization options for cytometry data, by using Self-Organizing Map clustering and Minimal Spanning Trees.

License GPL (>= 2)

LazyData true

URL http://www.r-project.org, http://dambi.ugent.be

biocViews CellBiology, FlowCytometry, Clustering, Visualization, Software, CellBasedAssays

RoxygenNote 7.1.0

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

git_branch RELEASE_3_11

git_last_commit 13887e0

git_last_commit_date 2020-04-27

Date/Publication 2020-10-16

R topics documented:

`AddFlowFrame` *Add a flowFrame to the data variable of the FlowSOM object*

Description

Add a flowFrame to the data variable of the FlowSOM object

Usage

```
AddFlowFrame(fsom, flowFrame)
```

Arguments

`fsom`	FlowSOM object, as constructed by the ReadInput function
`flowFrame`	flowFrame to add to the FlowSOM object

Value

FlowSOM object with data added

See Also

`ReadInput`

`AggregateFlowFrames` *Aggregate multiple fcs files together*

Description

Aggregate multiple fcs files to analyze them simultaneously. A new fcs file is written, which contains about `cTotal` cells, with `ceiling(cTotal/nFiles)` cells from each file. Two new columns are added: a column indicating the original file by index, and a noisy version of this for better plotting opportunities (index plus or minus a value between 0 and 0.1).

Usage

```
AggregateFlowFrames(
  fileNames,
  cTotal,
  writeOutput = FALSE,
  outputFile = "aggregate.fcs",
  writeMeta = FALSE,
  keepOrder = FALSE,
  verbose = FALSE,
  ...
)
```

Arguments

`fileNames`	Character vector containing full paths to the fcs files to aggregate
`cTotal`	Total number of cells to write to the output file
`writeOutput`	Whether to write the resulting flowframe to a file
`outputFile`	Full path to output file
`writeMeta`	If TRUE, files with the indices of the selected cells are generated
`keepOrder`	If TRUE, the random subsample will be ordered in the same way as they were originally ordered in the file. Default = FALSE.
`verbose`	If TRUE, prints an update every time it starts processing a new file. Default = FALSE.
`...`	Additional arguments to pass to read.FCS

Value

This function does not return anything, but will write a file with about cTotal cells to outputFile

See Also

ceiling

Examples

```
# Define filename
fileName <- system.file("extdata", "68983.fcs", package="FlowSOM")
# This example will sample 2 times 500 cells.
ff_new <- AggregateFlowFrames(c(fileName,fileName),1000)
```

BuildMST *Build Minimal Spanning Tree*

Description

Add minimal spanning tree description to the FlowSOM object

Usage

```
BuildMST(fsom, silent = FALSE, tSNE = FALSE)
```

Arguments

fsom	FlowSOM object, as generated by BuildSOM
silent	If TRUE, no progress updates will be printed
tSNE	If TRUE, an alternative tSNE layout is computed as well

Value

FlowSOM object containing MST description

See Also

BuildSOM, PlotStars

Examples

```
# Read from file, build self-organizing map
fileName <- system.file("extdata", "68983.fcs", package="FlowSOM")
flowSOM.res <- ReadInput(fileName, compensate=TRUE,transform=TRUE,
                         scale=TRUE)
flowSOM.res <- BuildSOM(flowSOM.res,colsToUse=c(9,12,14:18))

# Build the Minimal Spanning Tree
flowSOM.res <- BuildMST(flowSOM.res)
```

BuildSOM *Build a self-organizing map*

Description

Build a SOM based on the data contained in the FlowSOM object

Usage

```
BuildSOM(fsom, colsToUse = NULL, silent = FALSE, ...)
```

Arguments

`fsom`	FlowSOM object containing the data, as constructed by the `ReadInput` function
`colsToUse`	column names or indices to use for building the SOM
`silent`	if `TRUE`, no progress updates will be printed
`...`	options to pass on to the SOM function (xdim, ydim, rlen, mst, alpha, radius, init, distf, importance)

Value

FlowSOM object containing the SOM result, which can be used as input for the `BuildMST` function

References

This code is strongly based on the kohonen package. R. Wehrens and L.M.C. Buydens, Self- and Super-organising Maps in R: the kohonen package J. Stat. Softw., 21(5), 2007

See Also

`ReadInput`,`BuildMST`

Examples

```
# Read from file
fileName <- system.file("extdata", "68983.fcs", package="FlowSOM")
flowSOM.res <- ReadInput(fileName, compensate=TRUE, transform=TRUE,
                         scale=TRUE)

# Build the Self-Organizing Map
# E.g. with gridsize 5x5, presenting the dataset 20 times,
# no use of MST in neighbourhood calculations in between
flowSOM.res <- BuildSOM(flowSOM.res, colsToUse=c(9,12,14:18),
                        xdim=5, ydim=5, rlen=20)

# Build the minimal spanning tree and apply metaclustering
flowSOM.res <- BuildMST(flowSOM.res)
metacl <- MetaClustering(flowSOM.res$map$codes,
                         "metaClustering_consensus", max=10)
```

computeBackgroundColor *Internal function for computing background nodes*

Description

Internal function for computing background nodes

Usage

```
computeBackgroundColor(
  backgroundValues,
  backgroundColor,
  backgroundLim = NULL,
  backgroundBreaks = NULL
)
```

Arguments

backgroundValues	Values to be used for background coloring, either numerical values or something that can be made into a factor (e.g. a clustering)
backgroundColor	Colorpalette to be used for the background coloring . Can be either a function or an array specifying colors
backgroundLim	Only used when backgroundValues are numerical. Defaults to min and max of the backgroundValues.
backgroundBreaks	Breaks to pass on to cut, to split numerical background values. If NULL, the length of backgroundColor will be used (default 100).

CountGroups *Calculate differences in cell counts between groups*

Description

Calculate differences in cell counts between groups

Usage

```
CountGroups(fsom, groups, plot = TRUE, silent = FALSE)
```

Arguments

fsom	FlowSOM object as generated by BuildSOM
groups	List containing an array with file names for each group
plot	Logical. If TRUE, make a starplot of each individual file
silent	Logical. If TRUE, print progress messages

Value

Distance matrix

Examples

```
set.seed(1)

# Build the FlowSOM tree on the example file
fileName <- system.file("extdata", "68983.fcs", package="FlowSOM")
flowSOM.res <- FlowSOM(fileName, compensate=TRUE,transform=TRUE,
                 scale=TRUE,colsToUse=c(9,12,14:18),nClus = 10)

# Have a look at the resulting tree
PlotStars(flowSOM.res[[1]],backgroundValues = as.factor(flowSOM.res[[2]]))

# Select all cells except the branch that corresponds with automated
# cluster 7 (CD3+ TCRyd +) and write te another file for the example
# In practice you would not generate any new file but use your different
# files from your different groups
ff <- flowCore::read.FCS(fileName)
ff_tmp <- ff[flowSOM.res[[1]]$map$mapping[,1] %in%
                  which(flowSOM.res[[2]] != 7),]
flowCore::write.FCS(ff_tmp,file="ff_tmp.fcs")
# Make an extra file without cluster 7 and double amount of cluster 10
ff_tmp <- ff[c(which(flowSOM.res[[1]]$map$mapping[,1] %in%
                              which(flowSOM.res[[2]] != 7)),
               which(flowSOM.res[[1]]$map$mapping[,1] %in%
                              which(flowSOM.res[[2]] == 5))),]
flowCore::write.FCS(ff_tmp,file="ff_tmp2.fcs")

# Compare the original file with the two new files we made
groupRes <- CountGroups(flowSOM.res[[1]],
              groups=list("AllCells"=c(fileName),
                        "Without_ydTcells"=c("ff_tmp.fcs","ff_tmp2.fcs")))
PlotGroups(flowSOM.res[[1]], groupRes)

# Compare only the file with the double amount of cluster 10
groupRes <- CountGroups(flowSOM.res[[1]],
              groups=list("AllCells"=c(fileName),
              "Without_ydTcells"=c("ff_tmp2.fcs")))
PlotGroups(flowSOM.res[[1]], groupRes)
```

`Dist.MST` *Calculate distance matrix using a minimal spanning tree neighbourhood*

Description

Calculate distance matrix using a minimal spanning tree neighbourhood

Usage

```
Dist.MST(X)
```

Arguments

`X`	matrix in which each row represents a point

Value

Distance matrix

`FlowSOM` *Run the FlowSOM algorithm*

Description

Method to run general FlowSOM workflow. Will scale the data and uses consensus meta-clustering by default.

Usage

```
FlowSOM(
  input,
  pattern = ".fcs",
  compensate = FALSE,
  spillover = NULL,
  transform = FALSE,
  toTransform = NULL,
  transformFunction = flowCore::logicleTransform(),
  scale = TRUE,
  scaled.center = TRUE,
  scaled.scale = TRUE,
  silent = TRUE,
  colsToUse,
  nClus = NULL,
  maxMeta,
  importance = NULL,
  seed = NULL,
  ...
)
```

Arguments

`input`	a flowFrame, a flowSet or an array of paths to files or directories
`pattern`	if input is an array of file- or directorynames, select only files containing pattern
`compensate`	logical, does the data need to be compensated
`spillover`	spillover matrix to compensate with If NULL and compensate=TRUE, we will look for $SPILL description in fcs file.
`transform`	logical, does the data need to be transformed with a logicle transform
`toTransform`	column names or indices that need to be transformed. If `NULL` and transform = `TRUE`, column names of `$SPILL` description in fcs file will be used.
`transformFunction`	Defaults to logicleTransform()

`scale`	logical, does the data needs to be rescaled
`scaled.center`	see `scale`
`scaled.scale`	see `scale`
`silent`	if `TRUE`, no progress updates will be printed
`colsToUse`	column names or indices to use for building the SOM
`nClus`	Exact number of clusters for meta-clustering. If `NULL`, several options will be tried (`1:maxMeta`)
`maxMeta`	Maximum number of clusters to try out for meta-clustering. Ignored if nClus is specified
`importance`	array with numeric values. Parameters will be scaled according to importance
`seed`	Set a seed for reproducible results
`...`	options to pass on to the SOM function (xdim, ydim, rlen, mst, alpha, radius, init, distf)

Value

A `list` with two items: the first is the flowSOM object containing all information (see the vignette for more detailed information about this object), the second is the metaclustering of the nodes of the grid. This is a wrapper function for `ReadInput`, `BuildSOM`, `BuildMST` and `MetaClustering`. Executing them separately may provide more options.

See Also

`scale`,`ReadInput`,`BuildSOM`, `BuildMST`,`MetaClustering`

Examples

```
# Read from file
fileName <- system.file("extdata", "68983.fcs", package="FlowSOM")
flowSOM.res <- FlowSOM(fileName, compensate=TRUE,transform=TRUE,
                      scale=TRUE,colsToUse=c(9,12,14:18),nClus=10)
# Or read from flowFrame object
ff <- flowCore::read.FCS(fileName)
ff <- flowCore::compensate(ff,ff@description$SPILL)
ff <- flowCore::transform(ff,
         flowCore::transformList(colnames(ff@description$SPILL),
                                flowCore::logicleTransform()))
flowSOM.res <- FlowSOM(ff,scale=TRUE,colsToUse=c(9,12,14:18),nClus=10)

# Plot results
PlotStars(flowSOM.res$FlowSOM,
          backgroundValues = flowSOM.res$metaclustering)

# Get metaclustering per cell
flowSOM.clustering <- GetMetaclusters(flowSOM.res)
```

FlowSOMSubset *FlowSOM subset*

Description

Take a subset from a FlowSOM object

Usage

```
FlowSOMSubset(fsom, ids)
```

Arguments

`fsom`	FlowSOM object, as generated by `BuildMST`
`ids`	Array containing the ids to keep

Value

FlowSOM object containg updated data and medianvalues, but with the same grid

See Also

`BuildMST`

Examples

```
# Read two files (Artificially, as we just split 1 file in 2 subsets)
fileName <- system.file("extdata", "68983.fcs", package="FlowSOM")
ff1 <- flowCore::read.FCS(fileName)[1:1000,]
ff1@description$FIL <- "File1"
ff2 <- flowCore::read.FCS(fileName)[1001:2000,]
ff2@description$FIL <- "File2"

flowSOM.res <- FlowSOM(flowCore::flowSet(c(ff1,ff2)), compensate=TRUE,
                       transform=TRUE, scale=TRUE,
                       colsToUse=c(9,12,14:18), maxMeta=10)
fSOM <- flowSOM.res[[1]]

# see $metadata for subsets:
fSOM$metaData

# Use only the second file, without changing the map
fSOM2 <- FlowSOMSubset(fSOM,
                       (fSOM$metaData[[2]][1]):(fSOM$metaData[[2]][2]))
```

`FMeasure` *F measure*

Description

Compute the F measure between two clustering results

Usage

```
FMeasure(realClusters, predictedClusters, silent = FALSE)
```

Arguments

`realClusters`	Array containing real cluster labels for each sample
`predictedClusters`	Array containing predicted cluster labels for each sample
`silent`	Logical, if FALSE (default), print some information about precision and recall

Value

F measure score

Examples

```
# Generate some random data as an example
realClusters <- sample(1:5,100,replace = TRUE)
predictedClusters <- sample(1:6, 100, replace = TRUE)

# Calculate the FMeasure
FMeasure(realClusters,predictedClusters)
```

`GetClusters` *Get cluster label for all individual cells*

Description

Get cluster label for all individual cells

Usage

```
GetClusters(fsom)
```

Arguments

`fsom`	FlowSOM object as generated by the FlowSOM function or the BuildSOM function

Value

vector label for every cell

Examples

```
fileName <- system.file("extdata", "68983.fcs", package="FlowSOM")
flowSOM.res <- FlowSOM(fileName, compensate=TRUE,transform=TRUE,
                      scale=TRUE,colsToUse=c(9,12,14:18),nClus=10)
cluster_labels <- GetClusters(flowSOM.res)
cluster_labels <- GetClusters(flowSOM.res$FlowSOM)
```

`GetCVs` *Get CV values for all clusters*

Description

Get CV values for all clusters

Usage

```
GetCVs(fsom)
```

Arguments

`fsom`	FlowSOM object as generated by the FlowSOM function or the BuildSOM function

Value

Matrix with coefficient of variation values for each marker

fileName <- system.file("extdata", "68983.fcs", package="FlowSOM") flowSOM.res <- FlowSOM(fileName, compensate=TRUE,transform=TRUE, scale=TRUE,colsToUse=c(9,12,14:18),nClus=10) cvs <- GetCVs(flowSOM.res) cvs <- GetCVs(flowSOM.res$FlowSOM)

`GetFlowJoLabels` *Process a flowjo workspace file*

Description

Reads a flowjo workspace file using the `flowWorkspace` library and returns a list with a matrix containing gating results and a vector with a label for each cell from a set of specified gates

Usage

```
GetFlowJoLabels(
  files,
  wsp_file,
  group = "All Samples",
  cell_types = NULL,
  get_data = FALSE,
  ...
)
```

Arguments

`files`	The fcs files of interest
`wsp_file`	The FlowJo wsp file to read
`group`	The FlowJo group to parse. Default "All Samples".
`cell_types`	Cell types to use for final labeling the cells. Should correspond with a subset of the gate names in FlowJo.
`get_data`	If true, flowframes are returned as well.
`...`	Extra arguments to pass to CytoML::flowjo_to_gatingset

Value

This function returns a list, which for every file contains a list in which the first element ("matrix") is a matrix containing filtering results for each specified gate and the second element ("manual") is a vector which assigns one label to each cell. If only one file is given, only one list is returned instead of a list of lists.

See Also

PlotPies

Examples

```
# Identify the files
fcs_file <- system.file("extdata", "68983.fcs", package = "FlowSOM")
wsp_file <- system.file("extdata", "gating.wsp", package = "FlowSOM")

# Specify the cell types of interest for assigning one label per cell
cell_types <- c("B cells",
                "gd T cells", "CD4 T cells", "CD8 T cells",
                "NK cells","NK T cells")

# Parse the FlowJo workspace
gatingResult <- GetFlowJoLabels(fcs_file, wsp_file,
                                cell_types = cell_types,
                                get_data = TRUE)

# Check the number of cells assigned to each gate
colSums(gatingResult$matrix)

# Build a FlowSOM tree
flowSOM.res <- FlowSOM(gatingResult$flowFrame,
                       colsToUse = c(9,12,14:18),
                       nClus = 10,
                       seed = 1)

 # Plot pies indicating the percentage of cell types present in the nodes
 PlotPies(flowSOM.res$FlowSOM,
          gatingResult$manual,
          backgroundValues = flowSOM.res$metaclustering)
```

GetMetaclusters *Get metacluster label for all individual cells*

Description

Get metacluster label for all individual cells

Usage

```
GetMetaclusters(fsom, meta = NULL)
```

Arguments

`fsom`	FlowSOM object as generated by the FlowSOM function or the BuildSOM function
`meta`	Metacluster label for each FlowSOM cluster. If this is NULL, the fsom argument should be as generated by the FlowSOM function, and fsom$metaclustering will be used.

Value

vector label for every cell

Examples

```
fileName <- system.file("extdata", "68983.fcs", package="FlowSOM")
flowSOM.res <- FlowSOM(fileName, compensate=TRUE,transform=TRUE,
                       scale=TRUE,colsToUse=c(9,12,14:18),nClus=10)
metacluster_labels <- GetMetaclusters(flowSOM.res)
metacluster_labels <- GetMetaclusters(flowSOM.res$FlowSOM,
                                      meta = flowSOM.res$metaclustering)
```

GetMFIs *Get MFI values for all clusters*

Description

Get MFI values for all clusters

Usage

```
GetMFIs(fsom, colsUsed = FALSE, prettyColnames = FALSE)
```

Arguments

`fsom`	FlowSOM object as generated by the FlowSOM function or the BuildSOM function
`colsUsed`	logical. Should report only the columns used to build the SOM. Default = FALSE.
`prettyColnames`	logical. Should report pretty column names instead of standard column names. Default = FALSE.

Value

Matrix with median values for each marker

Examples

```
fileName <- system.file("extdata", "68983.fcs", package="FlowSOM")
flowSOM.res <- FlowSOM(fileName, compensate=TRUE,transform=TRUE,
                       scale=TRUE,colsToUse=c(9,12,14:18),nClus=10)
mfis <- GetMFIs(flowSOM.res)
mfis <- GetMFIs(flowSOM.res$FlowSOM)
```

`get_channels` *get_channels*

Description

Get channel names for an array of markers, given a flowframe

Usage

```
get_channels(ff, markers)
```

Arguments

`ff`	The flowFrame of interest
`markers`	Vector with markers or channels of interest

Value

Corresponding channel names

See Also

`get_markers`

Examples

```
   # Read the flowFrame
   fileName <- system.file("extdata", "68983.fcs", package="FlowSOM")
   ff <- flowCore::read.FCS(fileName)
   get_channels(ff, c("FSC-A", "CD3", "FITC-A"))
   get_markers(ff, c("FSC-A", "CD3", "FITC-A"))
```

`get_markers` *get_markers*

Description

Get marker names, given a flowframe. As available in "desc". If this is NA, defaults to channel name.

Usage

```
get_markers(ff, markers)
```

Arguments

`ff`	The flowFrame of interest
`markers`	Vector with markers or channels of interest

Value

Corresponding marker names

See Also

`get_channels`

Examples

```
# Read the flowFrame
fileName <- system.file("extdata", "68983.fcs", package="FlowSOM")
ff <- flowCore::read.FCS(fileName)
get_channels(ff, c("FSC-A", "CD3", "FITC-A"))
get_markers(ff, c("FSC-A", "CD3", "FITC-A"))
```

`Initialize_KWSP` *Select k well spread points from X*

Description

Select k well spread points from X

Usage

```
Initialize_KWSP(X, xdim, ydim)
```

Arguments

`X`	matrix in which each row represents a point
`xdim`	x dimension of the grid
`ydim`	y dimension of the grid

Value

array containing the selected selected rows

Examples

```
points <- matrix(1:1000, ncol = 10)
selection <- Initialize_KWSP(points, 3, 3)
```

`Initialize_PCA` *Create a grid from first 2 PCA components*

Description

Create a grid from first 2 PCA components

Usage

```
Initialize_PCA(data, xdim, ydim)
```

Arguments

`data`	matrix in which each row represents a point
`xdim`	x dimension of the grid
`ydim`	y dimension of the grid

Value

array containing the selected selected rows

Examples

```
points <- matrix(1:1000, ncol = 10)
selection <- Initialize_PCA(points, 3, 3)
```

MapDataToCodes *Assign nearest node to each datapoint*

Description

Assign nearest node to each datapoint

Usage

```
MapDataToCodes(codes, newdata, distf = 2)
```

Arguments

`codes`	matrix with nodes of the SOM
`newdata`	datapoints to assign
`distf`	Distance function (1=manhattan, 2=euclidean, 3=chebyshev, 4=cosine)

Value

Array with nearest node id for each datapoint

MetaclusterCVs *MetaclusterCVs*

Description

Compute the coefficient of variation for the metaclusters

Usage

```
MetaclusterCVs(fsom)
```

Arguments

`fsom`	Result of calling the FlowSOM function

Value

Metacluster CVs

Examples

```
fileName <- system.file("extdata", "68983.fcs", package="FlowSOM")
ff <- flowCore::read.FCS(fileName)
ff <- flowCore::compensate(ff,ff@description$SPILL)
ff <- flowCore::transform(ff,
         flowCore::transformList(colnames(ff@description$SPILL),
                                 flowCore::logicleTransform()))
flowSOM.res <- FlowSOM(ff,scale=TRUE,colsToUse=c(9,12,14:18), nClus=10)
cvs <- MetaclusterCVs(flowSOM.res)
```

MetaClustering *MetaClustering*

Description

Cluster data with automatic number of cluster determination for several algorithms

Usage

```
MetaClustering(data, method, max = 20, ...)
```

Arguments

`data`	Matrix containing the data to cluster
`method`	Clustering method to use
`max`	Maximum number of clusters to try out
`...`	Extra parameters to pass along

Value

Numeric array indicating cluster for each datapoint

See Also

metaClustering_consensus

Examples

```
# Read from file, build self-organizing map and minimal spanning tree
fileName <- system.file("extdata", "68983.fcs", package="FlowSOM")
flowSOM.res <- ReadInput(fileName, compensate=TRUE,transform=TRUE,
                         scale=TRUE)
flowSOM.res <- BuildSOM(flowSOM.res,colsToUse=c(9,12,14:18))
flowSOM.res <- BuildMST(flowSOM.res)

# Apply metaclustering
metacl <- MetaClustering(flowSOM.res$map$codes,
                         "metaClustering_consensus",
                         max=10)

# Get metaclustering per cell
flowSOM.clustering <- metacl[flowSOM.res$map$mapping[,1]]
```

`metaClustering_consensus` *MetaClustering*

Description

Cluster data using hierarchical consensus clustering with k clusters

Usage

```
metaClustering_consensus(data, k = 7, seed = NULL)
```

Arguments

`data`	Matrix containing the data to cluster
`k`	Number of clusters
`seed`	Seed to pass to consensusClusterPlus

Value

Numeric array indicating cluster for each datapoint

See Also

`MetaClustering`

Examples

```
# Read from file, build self-organizing map and minimal spanning tree
fileName <- system.file("extdata", "68983.fcs", package="FlowSOM")
flowSOM.res <- ReadInput(fileName, compensate=TRUE,transform=TRUE,
                         scale=TRUE)
flowSOM.res <- BuildSOM(flowSOM.res,colsToUse=c(9,12,14:18))
flowSOM.res <- BuildMST(flowSOM.res)

# Apply consensus metaclustering
metacl <- metaClustering_consensus(flowSOM.res$map$codes,k=10)
```

`MetaclusterMFIs` *MetaclusterMFIs*

Description

Compute the median fluorescence intensities for the metaclusters

Usage

```
MetaclusterMFIs(fsom)
```

Arguments

`fsom`	Result of calling the FlowSOM function

Value

Metacluster MFIs

Examples

```
fileName <- system.file("extdata", "68983.fcs", package="FlowSOM")
ff <- flowCore::read.FCS(fileName)
ff <- flowCore::compensate(ff,ff@description$SPILL)
ff <- flowCore::transform(ff,
        flowCore::transformList(colnames(ff@description$SPILL),
                                flowCore::logicleTransform()))
flowSOM.res <- FlowSOM(ff,scale=TRUE,colsToUse=c(9,12,14:18),maxMeta=10)
mfis <- MetaclusterMFIs(flowSOM.res)
```

`NewData` *Map new data to a FlowSOM grid*

Description

New data is mapped to an existing FlowSOM object. The input is similar to the readInput function. A new FlowSOM object is created, with the same grid, but a new mapping, node sizes and mean values. The same preprocessing steps (compensation, tranformation and scaling) will happen to this file as was specified in the original FlowSOM call. The scaling parameters from the original grid will be used.

Usage

```
NewData(
  fsom,
  input,
  mad_allowed = 4,
  compensate = NULL,
  spillover = NULL,
  transform = NULL,
  toTransform = NULL,
  transformFunction = NULL,
  scale = NULL,
  scaled.center = NULL,
  scaled.scale = NULL
)
```

Arguments

`fsom`	FlowSOM object
`input`	A flowFrame, a flowSet or an array of paths to files or directories

`mad_allowed`	A warning is generated if the distance of the new data points to their closest cluster center is too big. This is computed based on the typical distance of the points from the original dataset assigned to that cluster, the threshold being set to median + mad_allowed * MAD. Default is 4.
`compensate`	logical, does the data need to be compensated. If NULL, the same value as in the original FlowSOM call will be used.
`spillover`	spillover matrix to compensate with. If NULL, the same value as in the original FlowSOM call will be used.
`transform`	logical, does the data need to be transformed. If NULL, the same value as in the original FlowSOM call will be used.
`toTransform`	column names or indices that need to be transformed. If NULL, the same value as in the original FlowSOM call will be used.
`transformFunction`	If NULL, the same value as in the original FlowSOM call will be used.
`scale`	Logical, does the data needs to be rescaled. If NULL, the same value as in the original FlowSOM call will be used.
`scaled.center`	See `scale`. If NULL, the same value as in the original FlowSOM call will be used.
`scaled.scale`	See `scale`. If NULL, the same value as in the original FlowSOM call will be used.

Value

A new FlowSOM object

See Also

`FlowSOMSubset` if you want to get a subset of the current data instead of a new dataset

Examples

```
# Build FlowSom result
fileName <- system.file("extdata", "68983.fcs", package="FlowSOM")
  ff <- flowCore::read.FCS(fileName)
  ff <- flowCore::compensate(ff,ff@description$SPILL)
  ff <- flowCore::transform(ff,
            flowCore::transformList(colnames(ff@description$SPILL),
                                    flowCore::logicleTransform()))
  flowSOM.res <- FlowSOM(ff[1:1000,], scale=TRUE, colsToUse=c(9,12,14:18),
                        nClus=10)

  # Map new data
  fSOM2 <- NewData(flowSOM.res, ff[1001:2000,])
```

PlotCenters *Plot cluster centers on a 2D plot*

Description

Plot FlowSOM nodes on a 2D scatter plot of the data

Usage

```
PlotCenters(fsom, marker1, marker2, MST = TRUE)
```

Arguments

`fsom`	FlowSOM object, as generated by `BuildMST`
`marker1`	Marker to show on the x-axis
`marker2`	Marker to show on the y-axis
`MST`	Type of visualization, if 1 plot tree, else plot grid

Value

Nothing is returned. A 2D scatter plot is drawn on which the nodes of the grid are indicated

See Also

`PlotStars`,`PlotPies`, `PlotMarker`,`BuildMST`

Examples

```
# Read from file, build self-organizing map and minimal spanning tree
fileName <- system.file("extdata", "68983.fcs", package="FlowSOM")
flowSOM.res <- ReadInput(fileName, compensate=TRUE,transform=TRUE,
                         scale=TRUE)
flowSOM.res <- BuildSOM(flowSOM.res,colsToUse=c(9,12,14:18))
flowSOM.res <- BuildMST(flowSOM.res)

# Plot centers
PlotCenters(flowSOM.res,"FSC-A","SSC-A")
PlotCenters(flowSOM.res,2,5)
```

PlotClusters2D *Plot nodes on scatter plot*

Description

Plot a 2D scatter plot. All cells of fsom$data are plotted in black, and those of the selected nodes are plotted in red. The nodes in the grid are indexed starting from the left bottom, first going right, then up. E.g. In a 10x10 grid, the node at top left will have index 91.

Usage

```
PlotClusters2D(
  fsom,
  marker1,
  marker2,
  nodes,
  col = "#FF0000",
  maxBgPoints = 10000,
  pchBackground = ".",
  pchCluster = ".",
  main = "",
  xlab = fsom$prettyColnames[marker1],
  ylab = fsom$prettyColnames[marker2],
  xlim = c(min(fsom$data[, marker1]), max(fsom$data[, marker1])),
  ylim = c(min(fsom$data[, marker2]), max(fsom$data[, marker2])),
  ...
)
```

Arguments

`fsom`	FlowSOM object, as generated by `BuildMST`
`marker1`	Marker to plot on the x-axis
`marker2`	Marker to plot on the y-axis
`nodes`	Nodes of which the cells should be plotted in red
`col`	Colors for all the cells in the selected nodes (ordered array)
`maxBgPoints`	Maximum number of background points to plot
`pchBackground`	Character to use for background cells
`pchCluster`	Character to use for cells in cluster
`main`	Title of the plot
`xlab`	Label for the x axis
`ylab`	Label for the y axis
`xlim`	Limits for the x axis
`ylim`	Limits for the y axis
`...`	Other parameters to pass on to plot

Value

Nothing is returned. A plot is drawn in which all cells are plotted in black and the cells of the selected nodes in red.

See Also

`PlotNumbers`,`PlotCenters`, `BuildMST`

Examples

```
# Read from file, build self-organizing map and minimal spanning tree
fileName <- system.file("extdata", "68983.fcs", package="FlowSOM")
flowSOM.res <- ReadInput(fileName, compensate=TRUE,transform=TRUE,
                         scale=TRUE)
flowSOM.res <- BuildSOM(flowSOM.res,colsToUse=c(9,12,14:18))
flowSOM.res <- BuildMST(flowSOM.res)

# Plot cells
PlotClusters2D(flowSOM.res,1,2,91)
```

`PlotGroups` *Plot differences between groups*

Description

Plot FlowSOM trees, where each node is represented by a star chart indicating mean marker values, the size of the node is relative to the mean percentage of cells present in each

Usage

```
PlotGroups(fsom, groups, tresh = NULL, p_tresh = 0.05, heatmap = FALSE, ...)
```

Arguments

`fsom`	FlowSOM object, as generated by `BuildMST` or the first list item of `FlowSOM`
`groups`	groups result as generated by `CountGroups`
`tresh`	Relative difference in groups before the node is coloured
`p_tresh`	Threshold on p-value from wilcox-test before the node is coloured. If this is not NULL, tresh will be ignored.
`heatmap`	If TRUE, the scores are plotted in a gradient instead of only the selection that passes the threshold
`...`	Other parameters to pass to `PlotStars`

Value

A vector containing the labels assigned to the nodes for all groups except the first

See Also

`PlotStars`,`CountGroups`

Examples

```
## Use the wrapper function to build a flowSOM object (saved in fsom[[1]])
## and a metaclustering (saved in fsom[[2]])
# fsom <- FlowSOM(ff,compensate = FALSE, transform = FALSE,scale = TRUE,
#              colsToUse = colsToUse, nClus = 10, silent = FALSE,
#             xdim=7, ydim=7)
```

```
## Make a list with for each group a list of files
## The reference group should be the first
# groups <- list("C"=file.path(workDir,grep("C",files,value = TRUE)),
#              "D"=file.path(workDir,grep("D",files,value=TRUE)))

## Compute the percentages for all groups
# groups_res <- CountGroups(fsom[[1]],groups)

## Plot the groups. For all groups except the first, differences with the
## first group are indicated
# annotation <- PlotGroups(fsom[[1]],groups_res)
```

PlotLabels *Plot a label in each node*

Description

Plot FlowSOM grid or tree, with in each node a label. Especially useful to show metacluster numbers

Usage

```
PlotLabels(
  fsom,
  labels,
  view = "MST",
  main = NULL,
  nodeSize = fsom$MST$size,
  fontSize = 1,
  backgroundValues = NULL,
  backgroundColor = function(n) {     grDevices::rainbow(n, alpha = 0.3) },
  backgroundLim = NULL,
  backgroundBreaks = NULL
)
```

Arguments

`fsom`	FlowSOM object, as generated by `BuildMST`
`labels`	A label for every node
`view`	Preferred view, options: "MST", "grid" or "tSNE" (if this option was selected while building the MST)
`main`	Title of the plot
`nodeSize`	Nodesize. The plot might be easier to read if this is a constant number, e.g. 10 or 15
`fontSize`	Fontsize, passed to label.cex
`backgroundValues`	Values to be used for background coloring, either numerical values or something that can be made into a factor (e.g. a clustering)

`backgroundColor`	Colorpalette to be used for the background coloring . Can be either a function or an array specifying colors
`backgroundLim`	Only used when backgroundValues are numerical. Defaults to min and max of the backgroundValues.
`backgroundBreaks`	Breaks to pass on to `cut`, to split numerical background values. If NULL, the length of backgroundColor will be used (default 100).

Value

Nothing is returned. A plot is drawn in which each node is assigned a label

See Also

`PlotNumbers`

Examples

```
# Read from file, build self-organizing map and minimal spanning tree
fileName <- system.file("extdata", "68983.fcs", package="FlowSOM")
ff <- flowCore::read.FCS(fileName)
ff <- flowCore::compensate(ff, ff@description$SPILL)
ff <- flowCore::transform(ff, flowCore::estimateLogicle(ff,
                                          flowCore::colnames(ff)[8:18]))
flowSOM.res <- FlowSOM(ff,
                       scale=TRUE,
                       colsToUse=c(9,12,14:18),
                       nClus = 10,
                       seed = 1)

# Plot the node IDs
PlotLabels( flowSOM.res$FlowSOM, flowSOM.res$metaclustering, nodeSize=15)
```

`PlotMarker` *Plot marker values*

Description

Plot FlowSOM grid or tree, coloured by node values for a specific marker

Usage

```
PlotMarker(
  fsom,
  marker = NULL,
  view = "MST",
  main = NULL,
 colorPalette = grDevices::colorRampPalette(c("#00007F", "blue", "#007FFF", "cyan",
    "#7FFF7F", "yellow", "#FF7F00", "red", "#7F0000")),
  backgroundValues = NULL,
  backgroundColor = function(n) {     grDevices::rainbow(n, alpha = 0.3) },
  backgroundBreaks = NULL,
  backgroundLim = NULL
)
```

Arguments

`fsom`	FlowSOM object, as generated by `BuildMST`
`marker`	Name or index of marker to plot
`view`	Preferred view, options: "MST" (default), "grid" or "tSNE" (if this option was selected while building the MST)
`main`	Title of the plot
`colorPalette`	Color palette to use
`backgroundValues`	Values to be used for background coloring, either numerical values or something that can be made into a factor (e.g. a clustering)
`backgroundColor`	Colorpalette to be used for the background coloring . Can be either a function or an array specifying colors
`backgroundBreaks`	Breaks to pass on to `cut`, to split numerical background values. If NULL, the length of backgroundColor will be used (default 100).
`backgroundLim`	Only used when backgroundValues are numerical. Defaults to min and max of the backgroundValues.

Value

Nothing is returned. A plot is drawn in which each node is coloured depending on its median value for the given marker

References

This visualization technique resembles SPADE results. M. Linderman, P. Qiu, E. Simonds and Z. Bjornson (). spade: SPADE – An analysis and visualization tool for Flow Cytometry. R package version 1.12.2. http://cytospade.org

See Also

`PlotStars`,`PlotPies`, `PlotCenters`,`BuildMST`

Examples

```
# Read from file, build self-organizing map and minimal spanning tree
fileName <- system.file("extdata", "68983.fcs", package="FlowSOM")
flowSOM.res <- ReadInput(fileName, compensate=TRUE,transform=TRUE,
                         scale=TRUE)
flowSOM.res <- BuildSOM(flowSOM.res,colsToUse=c(9,12,14:18))
flowSOM.res <- BuildMST(flowSOM.res)

# Plot one marker
PlotMarker(flowSOM.res,"FSC-A")
```

PlotNode *Plot star chart*

Description

Plot a star chart indicating median marker values of a single node

Usage

```
PlotNode(
  fsom,
  id,
  markers = fsom$map$colsUsed,
 colorPalette = grDevices::colorRampPalette(c("#00007F", "blue", "#007FFF", "cyan",
    "#7FFF7F", "yellow", "#FF7F00", "red", "#7F0000")),
  main = paste0("Cluster ", id)
)
```

Arguments

`fsom`	FlowSOM object, as generated by `BuildMST` or the first element of the list returned by `FlowSOM`
`id`	Id of the node to plot (check PlotNumbers to get the ids)
`markers`	Array of markers to use. Default: the markers used to build the tree
`colorPalette`	Colorpalette to be used for the markers
`main`	Title of the plot

Value

Nothing is returned. A plot is drawn in which the node is represented by a star chart indicating the median fluorescence intensities.

See Also

`PlotStars`,`PlotNumbers`, `FlowSOM`

Examples

```
# Read from file, build self-organizing map and minimal spanning tree
fileName <- system.file("extdata", "68983.fcs", package="FlowSOM")
flowSOM.res <- FlowSOM(fileName, compensate=TRUE,transform=TRUE,
                         scale=TRUE,colsToUse=c(9,12,14:18),nClus=10)

# Plot stars indicating the MFI of the cells present in the nodes
PlotNode(flowSOM.res$FlowSOM,1)
```

`PlotNumbers` *Plot the index of each node*

Description

Plot FlowSOM grid or tree, with in each node a number indicating its index

Usage

```
PlotNumbers(
  fsom,
  view = "MST",
  main = NULL,
  nodeSize = fsom$MST$size,
  fontSize = 1,
  backgroundValues = NULL,
  backgroundColor = function(n) {     grDevices::rainbow(n, alpha = 0.3) },
  backgroundLim = NULL,
  backgroundBreaks = NULL
)
```

Arguments

`fsom`	FlowSOM object, as generated by `BuildMST`
`view`	Preferred view, options: "MST", "grid" or "tSNE" (if this option was selected while building the MST)
`main`	Title of the plot
`nodeSize`	Nodesize. The plot might be easier to read if this is a constant number, e.g. 10 or 15
`fontSize`	Fontsize, passed to label.cex
`backgroundValues`	Values to be used for background coloring, either numerical values or something that can be made into a factor (e.g. a clustering)
`backgroundColor`	Colorpalette to be used for the background coloring . Can be either a function or an array specifying colors
`backgroundLim`	Only used when backgroundValues are numerical. Defaults to min and max of the backgroundValues.
`backgroundBreaks`	Breaks to pass on to `cut`, to split numerical background values. If NULL, the length of backgroundColor will be used (default 100).

Value

Nothing is returned. A plot is drawn in which each node is assigned a number

See Also

`PlotMarker`,`PlotStars`, `PlotPies`,`PlotCenters`, `BuildMST`

Examples

```
# Read from file, build self-organizing map and minimal spanning tree
fileName <- system.file("extdata", "68983.fcs", package="FlowSOM")
flowSOM.res <- ReadInput(fileName, compensate=TRUE,transform=TRUE,
                         scale=TRUE)
flowSOM.res <- BuildSOM(flowSOM.res,colsToUse=c(9,12,14:18))
flowSOM.res <- BuildMST(flowSOM.res)

# Plot the node IDs
PlotNumbers(flowSOM.res)

# Adapt node size for easier readability
PlotNumbers(flowSOM.res, nodeSize=14)
```

`PlotOverview2D` *Plot metaclusters on scatter plots*

Description

Write multiple 2D scatter plots to a png file. All cells of fsom$data are plotted in black, and those of the selected metaclusters are plotted in color.

Usage

```
PlotOverview2D(fsom, markerlist, metaclusters, colors = NULL, ff, ...)
```

Arguments

`fsom`	FlowSOM object, as generated by `FlowSOM`. If using a FlowSOM object as generated by `BuildMST`, it needs to be wrapped in a list, list(FlowSOM = fsom, metaclustering = metaclustering).
`markerlist`	List in which each element is a pair of marker names
`metaclusters`	Metaclusters of interest
`colors`	Named vector with color value for each metacluster. If NULL (default) colorbrewer "paired" is interpolated
`ff`	flowFrame to use as reference for the marker names
`...`	Other parameters to pass on to PlotClusters2D

Value

Nothing is returned, but a plot is drawn for every markerpair and every metacluster. The individual cells are colored, and the center of each FlowSOM cluster is indicated with a blue cross.

See Also

`PlotClusters2D`

Examples

```
# Read from file, build self-organizing map and minimal spanning tree
fileName <- system.file("extdata", "68983.fcs", package="FlowSOM")
flowSOM.res <- FlowSOM(fileName,
                       compensate=TRUE, transform=TRUE, scale=TRUE,
                       colsToUse=c(9,12,14:18),
                       nClus = 10,
                       seed = 1)

# Plot cells
markers_of_interest = list(c("FSC-A", "SSC-A"),
                           c("CD3", "CD19"),
                           c("TCRb", "TCRyd"),
                           c("CD4", "CD8"))
metaclusters_of_interest = 1:10

# Recommended to write to png

png("Markeroverview.png",
    width = 500 * length(markers_of_interest),
    height = 500 * length(metaclusters_of_interest))
PlotOverview2D(flowSOM.res,
               markerlist = markers_of_interest,
               metaclusters = metaclusters_of_interest,
               pchCluster = 19,
               ff = flowCore::read.FCS(fileName))
dev.off()
```

PlotPies *Plot comparison with other clustering*

Description

Plot FlowSOM grid or tree, with pies indicating another clustering or manual gating result

Usage

```
PlotPies(
  fsom,
  cellTypes,
  view = "MST",
  colorPalette = grDevices::colorRampPalette(c("white", "#00007F", "blue", "#007FFF",
    "cyan", "#7FFF7F", "yellow", "#FF7F00", "red")),
  backgroundValues = NULL,
  backgroundColor = function(n) {     grDevices::rainbow(n, alpha = 0.3) },
  backgroundLim = NULL,
  backgroundBreaks = NULL,
  legend = TRUE,
  main = ""
)
```

Arguments

`fsom`	FlowSOM object, as generated by `BuildMST`
`cellTypes`	Array of factors indicating the celltypes
`view`	Preferred view, options: "MST", "grid" or "tSNE" (if this option was selected while building the MST)
`colorPalette`	Colorpalette to be used for the markers
`backgroundValues`	Values to be used for background coloring, either numerical values or something that can be made into a factor (e.g. a clustering)
`backgroundColor`	Colorpalette to be used for the background coloring . Can be either a function or an array specifying colors
`backgroundLim`	Only used when backgroundValues are numerical. Defaults to min and max of the backgroundValues.
`backgroundBreaks`	Breaks to pass on to `cut`, to split numerical background values. If NULL, the length of backgroundColor will be used (default 100).
`legend`	Logicle, if T add a legend
`main`	Title of the plot

Value

Nothing is returned. A plot is drawn in which each node is represented by a pie chart indicating the percentage of cells present of each cell type. At the end, the layout is set to 1 figure again.

See Also

`PlotStars`,`PlotMarker`, `PlotCenters`,`BuildMST`

Examples

```
#' # Identify the files
fcs_file <- system.file("extdata", "68983.fcs", package = "FlowSOM")
wsp_file <- system.file("extdata", "gating.wsp", package = "FlowSOM")

# Specify the cell types of interest for assigning one label per cell
cell_types <- c("B cells",
                "gd T cells", "CD4 T cells", "CD8 T cells",
                "NK cells","NK T cells")

# Parse the FlowJo workspace
library(flowWorkspace)
gatingResult <- GetFlowJoLabels(fcs_file, wsp_file,
                                cell_types = cell_types)

# Check the number of cells assigned to each gate
colSums(gatingResult$matrix)

# Build a FlowSOM tree
flowSOM.res <- FlowSOM(fcs_file,
                       compensate = TRUE,
                       transform = TRUE,
                          toTransform = 8:18,
                          colsToUse = c(9,12,14:18),
                          nClus = 10,
                          seed = 1)

# Plot pies indicating the percentage of cell types present in the nodes
PlotPies(flowSOM.res$FlowSOM,
         gatingResult$manual,
         backgroundValues = flowSOM.res$metaclustering)
```

`PlotSD` *Plot SD*

Description

— Function in development, use with caution — Plot FlowSOM grid or tree, coloured by standard deviaton

Usage

```
PlotSD(
  fsom,
  marker = NULL,
  view = "MST",
  main = NULL,
 colorPalette = grDevices::colorRampPalette(c("#00007F", "blue", "#007FFF", "cyan",
    "#7FFF7F", "yellow", "#FF7F00", "red", "#7F0000")),
  symmetric = FALSE,
  lim = NULL,
  backgroundValues = NULL,
  backgroundColor = function(n) {     grDevices::rainbow(n, alpha = 0.3) },
  backgroundLim = NULL,
  backgroundBreaks = NULL
)
```

Arguments

`fsom`	FlowSOM object, as generated by BuildMST
`marker`	If a marker is given, the sd for this marker is shown. Otherwise, the maximum ratio is used.
`view`	Preferred view, options: "MST", "grid" or "tSNE" (if this option was selected while building the MST)
`main`	Title of the plot
`colorPalette`	Color palette to use
`symmetric`	Plot colours symmetric around zero
`lim`	Variable limits
`backgroundValues`	Values to be used for background coloring, either numerical values or something that can be made into a factor (e.g. a clustering)

`backgroundColor`	Colorpalette to be used for the background coloring . Can be either a function or an array specifying colors
`backgroundLim`	Only used when backgroundValues are numerical. Defaults to min and max of the backgroundValues.
`backgroundBreaks`	Breaks to pass on to `cut`, to split numerical background values. If NULL, the length of backgroundColor will be used (default 100).

Details

From suggestion in email: I am currently considering a way to summarize for each node all the SD as one value. After computing the SD matrix (nrow = # nodes, ncol = # markers), I compute the median value per column, then divide the SD matrix by it, and finally take the maximum ratio of each line (aka node). Doing so I got a unique dispersion score per node.

Value

Nothing is returned. A plot is drawn in which each node is coloured depending on its standard deviation

See Also

`PlotMarker`,`PlotStars`, `PlotPies`,`PlotCenters`, `BuildMST`

Examples

```
# Read from file, build self-organizing map and minimal spanning tree
fileName <- system.file("extdata", "68983.fcs", package="FlowSOM")
flowSOM.res <- ReadInput(fileName, compensate=TRUE,transform=TRUE,
                         scale=TRUE)
flowSOM.res <- BuildSOM(flowSOM.res,colsToUse=c(9,12,14:18))
flowSOM.res <- BuildMST(flowSOM.res)

PlotSD(flowSOM.res)
```

`plotStarLegend` *Plot legend for star plot*

Description

Plot a single star chart, annotated with labels

Usage

```
plotStarLegend(labels, colors = grDevices::rainbow(length(labels)), main = "")
```

Arguments

`labels`	Names to show in the legend
`colors`	Corresponding colors
`main`	Title of the legend

Value

Nothing is returned. A plot is drawn with 1 star chart, which is filled completely and annotated with the given labels.

See Also

PlotStars

PlotStars *Plot star charts*

Description

Plot FlowSOM grid or tree, where each node is represented by a star chart indicating median marker values

Usage

```
PlotStars(
  fsom,
  markers = fsom$map$colsUsed,
  view = "MST",
 colorPalette = grDevices::colorRampPalette(c("#00007F", "blue", "#007FFF", "cyan",
    "#7FFF7F", "yellow", "#FF7F00", "red", "#7F0000")),
  starBg = "white",
  backgroundValues = NULL,
  backgroundColor = function(n) {     grDevices::rainbow(n, alpha = 0.3) },
  backgroundLim = NULL,
  backgroundBreaks = NULL,
  backgroundSize = NULL,
  thresholds = NULL,
  legend = TRUE,
  query = NULL,
  range = "all",
  main = ""
)
```

Arguments

fsom	FlowSOM object, as generated by BuildMST
markers	Array of markers to use. Default: the markers used to build the tree
view	Preferred view, options: "MST", "grid" or "tSNE" (if this option was selected while building the MST)
colorPalette	Colorpalette to be used for the markers
starBg	Background color inside the star circle. Default is "white". Can also be put to "transparent" (as was the case for older versions).
backgroundValues	Values to be used for background coloring, either numerical values or something that can be made into a factor (e.g. a clustering)

`backgroundColor`	Colorpalette to be used for the background coloring . Can be either a function or an array specifying colors
`backgroundLim`	Only used when backgroundValues are numerical. Defaults to min and max of the backgroundValues.
`backgroundBreaks`	Breaks to pass on to `cut`, to split numerical background values. If NULL, the length of backgroundColor will be used (default 100).
`backgroundSize`	Size of the background circles. Default 15.
`thresholds`	Optional. Array containing a number for each of the markers to be used as the split between high/low. If provided, the percentage of positive cells is indicated instead of the MFI
`legend`	Logical, if TRUE add a legend
`query`	Show a low/high profile for certain markers in the legend. See also `QueryStarPlot`
`range`	If "all" (default), range is computed on all markers passed, if "one", range is computed on every marker separately. The height of the pie pieces will be computed relative to this range.
`main`	Title of the plot

Value

Nothing is returned. A plot is drawn in which each node is represented by a star chart indicating the median fluorescence intensities. Resets the layout back to 1 plot at the end.

See Also

`PlotPies`,`PlotMarker`, `PlotCenters`, `BuildMST`

Examples

```
# Read from file, build self-organizing map and minimal spanning tree
fileName <- system.file("extdata", "68983.fcs", package="FlowSOM")
flowSOM.res <- ReadInput(fileName, compensate=TRUE,transform=TRUE,
                         scale=TRUE)
flowSOM.res <- BuildSOM(flowSOM.res,colsToUse=c(9,12,14:18))
flowSOM.res <- BuildMST(flowSOM.res)

# Plot stars indicating the MFI of the cells present in the nodes
PlotStars(flowSOM.res)
```

`PlotVariable` *Plot a variable for all nodes*

Description

Plot FlowSOM grid or tree, coloured by node values given in variable

Usage

```
PlotVariable(
  fsom,
  variable,
  view = "MST",
  main = NULL,
 colorPalette = grDevices::colorRampPalette(c("#00007F", "blue", "#007FFF", "cyan",
    "#7FFF7F", "yellow", "#FF7F00", "red", "#7F0000")),
  symmetric = FALSE,
  lim = NULL,
  backgroundValues = NULL,
  backgroundColor = function(n) {     grDevices::rainbow(n, alpha = 0.3) },
  backgroundLim = NULL,
  backgroundBreaks = NULL
)
```

Arguments

`fsom`	FlowSOM object, as generated by `BuildMST`
`variable`	Vector containing a value for each node
`view`	Preferred view, options: "MST", "grid" or "tSNE" (if this option was selected while building the MST)
`main`	Title of the plot
`colorPalette`	Color palette to use
`symmetric`	Plot colours symmetric around zero
`lim`	Variable limits
`backgroundValues`	Values to be used for background coloring, either numerical values or something that can be made into a factor (e.g. a clustering)
`backgroundColor`	Colorpalette to be used for the background coloring . Can be either a function or an array specifying colors
`backgroundLim`	Only used when backgroundValues are numerical. Defaults to min and max of the backgroundValues.
`backgroundBreaks`	Breaks to pass on to `cut`, to split numerical background values. If NULL, the length of backgroundColor will be used (default 100).

Value

Nothing is returned. A plot is drawn in which each node is coloured depending on its value for the given variable

See Also

`PlotMarker`,`PlotStars`, `PlotPies`,`PlotCenters`, `BuildMST`

Examples

```
# Read from file, build self-organizing map and minimal spanning tree
fileName <- system.file("extdata", "68983.fcs", package="FlowSOM")
flowSOM.res <- ReadInput(fileName, compensate=TRUE,transform=TRUE,
                         scale=TRUE)
flowSOM.res <- BuildSOM(flowSOM.res,colsToUse=c(9,12,14:18))
flowSOM.res <- BuildMST(flowSOM.res)

# Plot some random values
rand <- runif(flowSOM.res$map$nNodes)
PlotVariable(flowSOM.res,rand)
```

`Purity` *Calculate mean weighted cluster purity*

Description

Calculate mean weighted cluster purity

Usage

```
Purity(realClusters, predictedClusters, weighted = TRUE)
```

Arguments

`realClusters`	array with real cluster values
`predictedClusters`	array with predicted cluster values
`weighted`	logical. Should the mean be weighted depending on the number of poins in the predicted clusters

Value

Mean purity score, worst score, number of clusters with score < 0.75

Examples

```
# Generate some random data as an example
realClusters <- sample(1:5,100,replace = TRUE)
predictedClusters <- sample(1:6, 100, replace = TRUE)

# Calculate the FMeasure
Purity(realClusters,predictedClusters)
```

QueryStarPlot *Query a certain cell type*

Description

Identify nodes in the tree which resemble a certain profile of "high" or "low" marker expressions.

Usage

```
QueryStarPlot(fsom, query, plot = TRUE, color = "#ca0020", debug = FALSE, ...)
```

Arguments

fsom	FlowSOM object, as generated by BuildMST or the first list item of FlowSOM
query	Array containing "high" or "low" for the specified column names of the FlowSOM data
plot	If true, a plot with a gradient of scores for the nodes is shown
color	Color to use for nodes with a high score in the plot
debug	If TRUE, some extra output will be printed
...	Other parameters to pass to PlotStars

Value

A list, containing the ids of the selected nodes, the individual scores for all nodes and the scores for each marker for each node

Examples

```
file <- system.file("extdata", "68983.fcs", package="FlowSOM")
# Use the wrapper function to build a flowSOM object (saved in fsom[[1]])
# and a metaclustering (saved in fsom[[2]])
fsom <- FlowSOM(file,compensate = TRUE, transform = TRUE,scale = TRUE,
               colsToUse = c(9,12,14:18), nClus = 10, silent = FALSE,
               xdim=7, ydim=7)
query <- c("PE-Cy7-A" = "high", #CD3
           "APC-Cy7-A" = "high", #TCRb
           "Pacific Blue-A" = "high") #CD8
query_res <- QueryStarPlot(UpdateNodeSize(fsom[[1]],reset=TRUE), query)

cellTypes <- factor(rep("Unknown",49),levels=c("Unknown","CD8 T cells"))
cellTypes[query_res$selected] <- "CD8 T cells"
PlotStars(fsom[[1]],
              backgroundValues=cellTypes,
              backgroundColor=c("#FFFFFF00","#ca0020aa"))
```

`ReadInput` *Read fcs-files or flowframes*

Description

Take some input and return FlowSOM object containing a matrix with the preprocessed data (compensated, transformed, scaled)

Usage

```
ReadInput(
  input,
  pattern = ".fcs",
  compensate = FALSE,
  spillover = NULL,
  transform = FALSE,
  toTransform = NULL,
  transformFunction = flowCore::logicleTransform(),
  scale = FALSE,
  scaled.center = TRUE,
  scaled.scale = TRUE,
  silent = FALSE
)
```

Arguments

`input`	a flowFrame, a flowSet or an array of paths to files or directories
`pattern`	if input is an array of file- or directorynames, select only files containing pattern
`compensate`	logical, does the data need to be compensated
`spillover`	spillover matrix to compensate with If `NULL` and compensate=`TRUE`, we will look for `$SPILL` description in fcs file.
`transform`	logical, does the data need to be transformed
`toTransform`	column names or indices that need to be transformed. If `NULL` and transform=`TRUE`, column names of `$SPILL` description in fcs file will be used.
`transformFunction`	Defaults to logicleTransform()
`scale`	logical, does the data needs to be rescaled
`scaled.center`	see `scale`
`scaled.scale`	see `scale`
`silent`	if `TRUE`, no progress updates will be printed

Value

FlowSOM object containing the data, which can be used as input for the BuildSOM function

See Also

`scale`,`BuildSOM`

Examples

```
# Read from file
fileName <- system.file("extdata", "68983.fcs", package="FlowSOM")
flowSOM.res <- ReadInput(fileName, compensate=TRUE,transform=TRUE,
                         scale=TRUE)

# Or read from flowFrame object
ff <- flowCore::read.FCS(fileName)
ff <- flowCore::compensate(ff,ff@description$SPILL)
ff <- flowCore::transform(ff,
                 flowCore::transformList(colnames(ff@description$SPILL),
                                 flowCore::logicleTransform()))
flowSOM.res <- ReadInput(ff,scale=TRUE)

# Build the self-organizing map and the minimal spanning tree
flowSOM.res <- BuildSOM(flowSOM.res,colsToUse=c(9,12,14:18))
flowSOM.res <- BuildMST(flowSOM.res)

# Apply metaclustering
metacl <- MetaClustering(flowSOM.res$map$codes,
                         "metaClustering_consensus",max=10)

# Get metaclustering per cell
flowSOM.clustering <- metacl[flowSOM.res$map$mapping[,1]]
```

`SaveClustersToFCS` *Write FlowSOM clustering results to the original FCS files*

Description

Write FlowSOM clustering results to the original FCS files

Usage

```
SaveClustersToFCS(
  fsom,
  original_files,
  pp_files = original_files,
  selection_files = NULL,
  silent = FALSE
)
```

Arguments

`fsom`	FlowSOM object as generated by BuildSOM
`original_files`	FCS files that should be extended
`pp_files`	FCS files that correspond to the input of FlowSOM
`selection_files`	Files indicating which cells of the original files correspond to the input files
`silent`	If FALSE (default), print some extra output

Value

Saves the extended fcs file as [originalName]_FlowSOM.fcs

`SOM` *Build a self-organizing map*

Description

Build a self-organizing map

Usage

```
SOM(
  data,
  xdim = 10,
  ydim = 10,
  rlen = 10,
  mst = 1,
  alpha = c(0.05, 0.01),
  radius = stats::quantile(nhbrdist, 0.67) * c(1, 0),
  init = FALSE,
  initf = Initialize_KWSP,
  distf = 2,
  silent = FALSE,
  codes = NULL,
  importance = NULL
)
```

Arguments

`data`	Matrix containing the training data
`xdim`	Width of the grid
`ydim`	Hight of the grid
`rlen`	Number of times to loop over the training data for each MST
`mst`	Number of times to build an MST
`alpha`	Start and end learning rate
`radius`	Start and end radius
`init`	Initialize cluster centers in a non-random way
`initf`	Use the given initialization function if init==T (default: Initialize_KWSP)
`distf`	Distance function (1=manhattan, 2=euclidean, 3=chebyshev, 4=cosine)
`silent`	If FALSE, print status updates
`codes`	Cluster centers to start with
`importance`	array with numeric values. Parameters will be scaled according to importance

Value

A list containing all parameter settings and results

References

This code is strongly based on the kohonen package. R. Wehrens and L.M.C. Buydens, Self- and Super-organising Maps in R: the kohonen package J. Stat. Softw., 21(5), 2007

See Also

`BuildSOM`

`TestOutliers` *Test if any cells are too far from their cluster centers*

Description

For every cluster, the distance from the cells to the cluster centers is used to label cells which deviate too far as outliers. The threshold is chosen as the median distance + `mad_allowed` times the median absolute deviation of the distances.

Usage

```
TestOutliers(
  fsom,
  mad_allowed = 4,
  fsom_reference = NULL,
  plot = FALSE,
  img_file = "testOutliers.pdf"
)
```

Arguments

`fsom`	FlowSOM object
`mad_allowed`	Number of median absolute deviations allowed. Default = 4.
`fsom_reference`	FlowSOM object to use as reference. If NULL (default), the original fsom object is used.
`plot`	Should a plot be generated showing the distribution of the distances. Default is FALSE.
`img_file`	If plot is TRUE, the output will be written to this file. Default is "testOutliers.pdf"

Value

A new FlowSOM object

See Also

`FlowSOMSubset` if you want to get a subset of the current data instead of a new dataset

Examples

```
# Build FlowSom result
fileName <- system.file("extdata", "68983.fcs", package="FlowSOM")
ff <- flowCore::read.FCS(fileName)
flowSOM.res <- FlowSOM(ff,
                       compensate = TRUE, transform = TRUE, scale = TRUE,
                       colsToUse = c(9, 12, 14:18),
                       maxMeta = 10)

# Map new data
outlier_report <- TestOutliers(flowSOM.res)
```

`UpdateNodeSize` *Update nodesize of FlowSOM object*

Description

Add size property to the graph based on cellcount for each node

Usage

```
UpdateNodeSize(
  fsom,
  count = NULL,
  reset = FALSE,
  transform = sqrt,
  maxNodeSize = 15,
  shift = 0,
  scale = NULL
)
```

Arguments

`fsom`	FlowSOM object, as generated by `BuildMST`
`count`	Absolute cell count of the sample
`reset`	Logical. If `TRUE`, all nodes get the same size
`transform`	Transformation function. Use e.g. square root to let counts correspond with area of node instead of radius
`maxNodeSize`	Maximum node size after rescaling. Default: 15
`shift`	Shift of the counts, defaults to 0
`scale`	Scaling of the counts, defaults to the maximum of the value minus the shift. With shift and scale set as default, the largest node will be maxNodesize and an empty node will have size 0

Value

Updated FlowSOM object

See Also

BuildMST

Examples

```
# Read from file, build self-organizing map and minimal spanning tree
fileName <- system.file("extdata", "68983.fcs", package="FlowSOM")
flowSOM.res <- ReadInput(fileName, compensate=TRUE,transform=TRUE,
                         scale=TRUE)
flowSOM.res <- BuildSOM(flowSOM.res,colsToUse=c(9,12,14:18))
flowSOM.res <- BuildMST(flowSOM.res)

# Give all nodes same size
flowSOM.res <- UpdateNodeSize(flowSOM.res,reset=TRUE)
PlotStars(flowSOM.res)

# Node sizes relative to amount of cells assigned to the node
flowSOM.res <- UpdateNodeSize(flowSOM.res)
PlotStars(flowSOM.res)
```

Index