Package 'GateFinder'

December 13, 2024

Type Package

Title Projection-based Gating Strategy Optimization for Flow and Mass Cytometry

Version 1.27.0

Date 2018-03-28

Author Nima Aghaeepour <naghaeep@gmail.com>, Erin F. Simonds <erin.simonds@gmail.com>

Maintainer Nima Aghaeepour <naghaeep@gmail.com>

Requires alphahull, maptools

Imports splancs, mvoutlier, methods, stats, diptest, flowCore, flowFP

Suggests RUnit, BiocGenerics

Description Given a vector of cluster memberships for a cell population, identifies a sequence of gates (polygon filters on 2D scatter plots) for isolation of that cell type.

License Artistic-2.0

biocViews ImmunoOncology, FlowCytometry, CellBiology, Clustering

NeedsCompilation no

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

git_branch devel

git_last_commit 77959e6

git_last_commit_date 2024-10-29

Repository Bioconductor 3.21

Date/Publication 2024-12-13

Contents

GateFinder-package *GateFinder*

Description

Given a vector of cluster memberships for a cell population, identifies a sequence of gates (polygon filters on 2D scatter plots) for isolation of that cell type.

Details

Package:	GateFinder
Type:	Package
Version:	1.0
Date:	2013-12-21
License:	Artistic-2.0

~~ An overview of how to use the package, including the most important ~~ ~~ functions ~~

Author(s)

Nima Aghaeepour <naghaeep@gmail.com> and Erin F. Simonds <erin.simonds@gmail.com>

Examples

```
library(flowCore)

data(LPSData)
##Select the target population. In this case cells with those with a pP38 expression (dimension 34) of higher than 3.
targetpop <- (exprs(rawdata)[,34]>3.5)

##Subset the markers that should be considered for gating.
x=exprs(rawdata)[,prop.markers]
colnames(x)=marker.names[prop.markers]

##Run GateFinder.
ans=GateFinder(x, targetpop)

##Make the plots.
plot (x, ans, c(2,3), targetpop)
plot(ans)


##Alternatively, using a flowFrame:
x=new('flowFrame', exprs=x)
ans=GateFinder(x, targetpop)
```

```
##Now you can use the gates and filters to subset the flowFrame. E.g.:
split(x, ans@flowEnv$Filter2)

##This function relies on an EXPERIMENTAL feature in flowUtils. Please be cautious when replying on this.
##Don't run without the optional flowUtils package installed.
##To write the gates into a GatingML file:
##library(flowUtils)
##write.gatingML(ans@flowEnv, 'GatingML.xml')
```

GateFinder *GateFinder*

Description

Given a vector of cluster memberships for a cell population, identifies a sequence of gates (polygon filters on 2D scatter plots) for isolation of that cell type.

Usage

```
GateFinder(x, targetpop, update.gates=FALSE, max.iter=2, beta=1, outlier.percentile=0.05, subsample=1
```

Arguments

x	A flowFrame or an expression matrix in which columns are markers and rows are cells.
targetpop	A vector of logical values one for each cell (TRUE = cells in the target population). If instead a vector of integers are supplied each integer value will be treated as a separate celltype of interest and a list of gating strategies will be returned.
update.gates	A boolean value indicating if the polygon gates should be updated after each gating step. update.gates=TRUE makes the analysis slower.
max.iter	The number of requested gating steps.
beta	A positive real value which control the trade-off between precision and recall in the F-measure calculation. Values smaller than 1 (and closer to 0) emphasize recall values and values larger than 1 emphasize precision.
outlier.percentile	The percentile of the empirical distribution of each 2d distribution (each scatter plot) that should be excluded before calculating the polygon gate. If a vector, all provided numbers will be tested and the one with the highest F-measure will be used.
subsample	The number of cells (integer) that should be randomly selected for calculation of the gating strategy (the gating strategy can be applied to all cells therefore the resulting populations will be based on all cells (see update.org.data)).
nstart	The number of randomized runs (integer). The results from the best (or median) randomized run will be used. See selection.criteria).

update.org.data
: A boolean controling weather the gating strategy calculated using a random subset of the cells should be applied to all cells or not.

randomize
: A boolean value to control if selection of the gates should be randomized. If TRUE the selection probability of each gate will be proportional to it's fmeasure. Otherwise the best gate will be selected for each step.

selection.criteria
: A string with values of either 'best' or 'median'. This determines if the run with the best or median fmeasure should be used as the final gating strategy.

unimodalitytest
: A boolean value. If TRUE the unimodality of the first principal component will be tested using a dip test and a warning is issued for $p < 0.05$.

predimx
: A vector of marker numbers for the x-axes of a pre-determined gating strategy.

predimy
: A vector of marker numbers for the y-axes of a pre-determined gating strategy.

convex
: A boolean value indicating if the target population is expected to be convex (for outlier removal purposes).

alpha
: alpha-hull threshold for non-convex gates.

Value

GatingProjection
: A GatingProjection object.

Author(s)

Nima Aghaeepour <naghaeep@gmail.com> and Erin F. Simonds <erin.simonds@gmail.com>.

References

Filzmoser, Peter, Ricardo Maronna, and Mark Werner. "Outlier identification in high dimensions." Computational Statistics & Data Analysis 52, no. 3 (2008): 1694-1711.

Examples

```
library(flowCore)

data(LPSData)
##Select the target population. In this case cells with those with a pP38 expression (dimension 34) of higher than 3.
targetpop <- (exprs(rawdata)[,34]>3.5)

##Subset the markers that should be considered for gating.
x=exprs(rawdata)[,prop.markers]
colnames(x)=marker.names[prop.markers]

##Run GateFinder.
ans=GateFinder(x, targetpop)

##Make the plots.
```

```
plot (x, ans, c(2,3), targetpop)
plot(ans)


##Alternatively, using a flowFrame:
x=new('flowFrame', exprs=x)
ans=GateFinder(x, targetpop)

##Now you can use the gates and filters to subset the flowFrame. E.g.:
split(x, ans@flowEnv$Filter2)

##This function relies on an EXPERIMENTAL feature in flowUtils. Please be cautious when replying on this.
##Don't run without the optional flowUtils package installed.
##To write the gates into a GatingML file:
##library(flowUtils)
##write.gatingML(ans@flowEnv, 'GatingML.xml')
```

GatingProjection-class *Class* "GatingProjection"

Description

An object that stores the final gating projections as well as the scores calculated for each step.

Objects from the Class

Objects can be created by calls of the form `new("GatingProjection", ...)`.

Slots

`fmeasure`: A vector of F-measure values for each step of the identified gating strategy.

`precision`: A vector of precision values for each step of the identified gating strategy.

`recall`: A vector of recall values for each step of the identified gating strategy.

`dimx`: A vector of marker indexes for the x-axis of each step of the identified gating hierarchy.

`dimy`: A vector of marker indexes for the y-axis of each step of the identified gating hierarchy.

`gates`: A list of polygon gates for each step of the identified gating hierarchy.

`pops`: A list of vectors representing the cell population memberships for each step of the identified hierarchy.

`subsampleindex`: A vector of the indexes of the selected subsample of cells (if applicable).

`fmeasures`: A vector of F-measure values of multiple randomized attempts (if applicable).

`flowEnv`: An envirnoment for flowCore's polygon gates and intersect filters.

Methods

plot `signature(x = "GatingProjection", y = "ANY")`: Plot of F-measure, precision, and recall values of each gating step.

plot `signature(x = "matrix", y = "GatingProjection")`: Scatter plots of the raw data (from matrix x) for each step of the gating strategy. Gray dots represent cells that were removed in the previous step. Red dots represent the target cells.

Author(s)

Nima Aghaeepour <naghaeep@gmail.com> and Erin F. Simonds <erin.simonds@gmail.com>.

Examples

```
library(flowCore)

data(LPSData)
##Select the target population. In this case cells with those with a pP38 expression (dimension 34) of higher than 3.
targetpop <- (exprs(rawdata)[,34]>3.5)

##Subset the markers that should be considered for gating.
x=exprs(rawdata)[,prop.markers]
colnames(x)=marker.names[prop.markers]

##Run GateFinder.
ans=GateFinder(x, targetpop)

##Make the plots.
plot (x, ans, c(2,3), targetpop)
plot(ans)


##Alternatively, using a flowFrame:
x=new('flowFrame', exprs=x)
ans=GateFinder(x, targetpop)

##Now you can use the gates and filters to subset the flowFrame. E.g.:
split(x, ans@flowEnv$Filter2)

##This function relies on an EXPERIMENTAL feature in flowUtils. Please be cautious when replying on this.
##Don't run without the optional flowUtils package installed.
##To write the gates into a GatingML file:
##library(flowUtils)
##write.gatingML(ans@flowEnv, 'GatingML.xml')
```

LPSData *LPSData*

Description

A dataset of two sets of scores (particularly, correlation with protection against HIV and overlap with the Naive T-cell population) assigned to immunophenotypes measured by flow cytometry. 10 markers were measured: KI-67, CD28, CD45RO, CD8, CD4, CD57, CCR5, CD27, CCR7, and CD127.

Usage

```
data(LPSData)
```

Details

This dataset consists of a matrix and two vectors:

`rawdata` The transformed expression values extracted from the original FCS file.

`prop.markers` the indexes of markers that should be considered for the gating strategy.

`marker.names` name of all markers (columns of matrix rawdata).

Author(s)

Nima Aghaeepour «naghaeep@gmail.com» and Erin F. Simonds «erin.simonds@gmail.com».

References

Bendall, Sean C., et al. "Single-cell mass cytometry of differential immune and drug responses across a human hematopoietic continuum." Science 332.6030 (2011): 687-696.

Examples

```
library(flowCore)

data(LPSData)

plot(exprs(rawdata)[,15:16],xlab=marker.names[15],ylab=marker.names[16])
```

`plot.GateFinder` *plot.GateFinder*

Description

Creates a scatter plot for each of the gating steps.

Usage

```
## S3 method for class 'GateFinder'
plot(x, y, ncolrow=c(1,max(targetpop)), targetpop=NULL, beta=NULL,  cexs=NULL, cols=NULL, subsample=le
```

Arguments

x	A flowFrame or an expression matrix in which columns are markers and rows are cells.
y	A GatingProjection object.
ncolrow	A vector of length 2 indicating the desired number of rows and columns in the plot.
targetpop	The target cell type.
beta	A positive real value which control the trade-off between precision and recall in the F-measure calculation. Values smaller than 1 (and closer to 0) emphasize recall values and values larger than 1 emphasize precision.
cexs	A vector of length 3 indicating the point sizes for 1-previously excluded cells 2-non-selected cells 3-selected cells.
cols	A vector of length 3 indicating the point colors for 1-previously excluded cells 2-non-selected cells 3-selected cells.
subsample	The number of randomized runs (integer). The results from the best (or median) randomized run will be used. See `selection.criteria`).
max.iter	A boolean controling weather the gating strategy calculated using a random subset of the cells should be applied to all cells or not.
pot	A boolean value. If true, the points of interest will be ploted on top of other points to increase visibility.
xlim	Static x-axis limits for the plots (vector of length 2).
ylim	Static y-axis limits for the plots (vector of length 2).
asinh.axis	A boolean value indicating if asinh axis ticks should be plotted (usually used for mass cytometry data).
...	Other arguments passed to the plot function.

Value

Plot	A GateFinder plot.

Author(s)

Nima Aghaeepour <naghaeep@gmail.com> and Erin F. Simonds <erin.simonds@gmail.com>.

Examples

```
library(flowCore)

data(LPSData)
##Select the target population. In this case cells with those with a pP38 expression (dimension 34) of higher than 3.
targetpop <- (exprs(rawdata)[,34]>3.5)

##Subset the markers that should be considered for gating.
x=exprs(rawdata)[,prop.markers]
colnames(x)=marker.names[prop.markers]
```

```
##Run GateFinder.
ans=GateFinder(x, targetpop)

##Make the plots.
plot(x, ans, c(2,3), targetpop)
plot(ans)


##Alternatively, using a flowFrame:
x=new('flowFrame', exprs=x)
ans=GateFinder(x, targetpop)

##Now you can use the gates and filters to subset the flowFrame. E.g.:
split(x, ans@flowEnv$Filter2)

##This function relies on an EXPERIMENTAL feature in flowUtils. Please be cautious when replying on this.
##Don't run without the optional flowUtils package installed.
##To write the gates into a GatingML file:
##library(flowUtils)
##write.gatingML(ans@flowEnv, 'GatingML.xml')
```

Index