Package 'flowUtils'

April 15, 2020

Type Package

Title Utilities for flow cytometry

Version 1.50.0

Author J. Spidlen., N. Gopalakrishnan, F. Hahne, B. Ellis, R. Gentleman, M. Dalphin, N. Le Meur, B. Purcell, W. Jiang

Maintainer Josef Spidlen <jspidlen@gmail.com>

Description Provides utilities for flow cytometry data.

Depends R (>= 2.2.0)

Imports Biobase, graph, methods, stats, utils, corpcor, RUnit, XML, flowCore (>= 1.32.0)

Suggests gatingMLData

Collate AllClasses.R gatingML.R helperFunctions.R gate-methods.R transforms.R parameter-methods.R compensation.R workflow2FlowJo.R writeGatingML.R zzz.R

License Artistic-2.0

biocViews ImmunoOncology, Infrastructure, FlowCytometry, CellBasedAssays, DecisionTree

URL https://github.com/jspidlen/flowUtils

BugReports https://github.com/jspidlen/flowUtils/issues

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

git_branch RELEASE_3_10

git_last_commit ca5bbfb

git_last_commit_date 2019-10-29

Date/Publication 2020-04-14

R topics documented:

`flowUtils-package` *Utilities for flow cytometry data*

Description

This package includes functions to import gates, transformations and compensations defined in compliance with Gating-ML specification version 1.5 and 2.0. This package depends on the flowCore package for methods to evaluate the Gating-ML files read into the workspace.

The main features of this package provide compatability to the data standards defined by the Gating-ML specification version 1.5 and 2.0.

The package also includes a Test Suite, which allows the user to test whether the implmentation of gates an, transformations are in compliance with the Gating-ML standard.

Author(s)

Maintainer: Josef Spidlen <jspidlen@bccrc.ca>

Authors: J. Spidlen, N. Gopalakrishnan, F. Hahne, B. Ellis, R. Gentleman, M. Dalphin, N. Le Meur, B. Purcell

References

Spidlen J, ISAC DSTF, Brinkman RR. 2014.
Gating-ML 2.0. International Society for Advancement of Cytometry (ISAC) standard for representing gating descriptions in flow cytometry.
http://flowcyt.sf.net/gating/20141009.pdf
http://flowcyt.sf.net/gating/20141009.full.zip

Spidlen J, Leif RC, Moore W, Roederer M, ISAC DSTF, Brinkman RR. 2008.
Gating-ML: XML-based gating descriptions in flow cytometry.
Cytometry A. 2008 Dec; 73A(12):1151–7. doi: 10.1002/cyto.a.20637.

Spidlen J, ISAC DSTF, Brinkman RR. 2008.
Gating-ML Candidate Recommendation for Gating Description in Flow Cytometry version 1.5.
http://flowcyt.sf.net/gating/Gating-ML.v1.5.081030.pdf
http://flowcyt.sf.net/gating/Gating-ML.v1.5.081030.full.zip
http://flowcyt.sf.net/gating/Gating-ML.v1.5.081030.Compliance-tests.081030.zip

See Also

flowCore

`read.gatingML` *Function to parse a Gating-ML XML file into objects in the R environment*

Description

This function parses a Gating-ML XML file defined in compliance with the Gating-ML recommendation into objects in the R environment, which can then be evaluated using functions provided by the flowCore package.

Usage

```
read.gatingML(file, flowEnv, ...)
```

Arguments

file	Gating-ML XML file describing gates, transformations and/or compensations
flowEnv	environment into which the R objects created from the Gating-ML XML file are to be stored
...	additional arguments that are passed to the methods

Details

The Gating-ML specification has been developed as an interchange format for the description of gates relevant to a flow cytometry experiment. Presently, we can read Gating-ML versions 1.5 and 2.0 of the specification. Version 2.0 is the most recent at the time of this writing.

Author(s)

J. Spidlen, N. Gopalakrishnan

References

Spidlen J, ISAC DSTF, Brinkman RR. 2014.
Gating-ML 2.0. International Society for Advancement of Cytometry (ISAC) standard for representing gating descriptions in flow cytometry.
http://flowcyt.sf.net/gating/20141009.pdf
http://flowcyt.sf.net/gating/20141009.full.zip

Spidlen J, Leif RC, Moore W, Roederer M, ISAC DSTF, Brinkman RR. 2008.
Gating-ML: XML-based gating descriptions in flow cytometry.
Cytometry A. 2008 Dec; 73A(12):1151–7. doi: 10.1002/cyto.a.20637.

Spidlen J, ISAC DSTF, Brinkman RR. 2008.
Gating-ML Candidate Recommendation for Gating Description in Flow Cytometry version 1.5.
http://flowcyt.sf.net/gating/Gating-ML.v1.5.081030.pdf
http://flowcyt.sf.net/gating/Gating-ML.v1.5.081030.full.zip
http://flowcyt.sf.net/gating/Gating-ML.v1.5.081030.Compliance-tests.081030.zip

See Also

write.gatingML

Examples

```
library("flowCore")

#########################
# Gating-ML 2.0 example #
#########################
flowEnv <- new.env()

fcsFile <- system.file("extdata/Gml2/FCSFiles",
  "data1.fcs", package="gatingMLData")
fcs <- read.FCS(fcsFile,
  transformation="linearize-with-PnG-scaling")
```

```
gateFile <- system.file("extdata/Gml2/Gating-MLFiles",
  "gates1.xml", package="gatingMLData")
read.gatingML(gateFile, flowEnv)
ls(flowEnv)

result = filter(fcs, flowEnv$Polygon1)
summary(result)

#########################
# Gating-ML 1.5 example #
#########################
flowEnv <- new.env()

fcsFile <- system.file("extdata/List-modeDataFiles",
  "fcs2_int16_13367ev_8par_GvHD.fcs", package="gatingMLData")
fcs <- read.FCS(fcsFile, transformation=FALSE)

gateFile <- system.file("extdata/Gating-MLFiles",
  "02CtSRectangular.xml", package="gatingMLData")
read.gatingML(gateFile, flowEnv)
ls(flowEnv)

result <- filter(fcs, flowEnv$CtSR_03)
summary(result)
```

`testGatingMLCompliance` *Function to perform all the Gating-ML compliance tests*

Description

This function performs the Gating-ML compliance tests. Either Gating-ML 1.5 or Gating-ML 2.0 compatibility may be checked. The Gating-ML XML files, FCS data files and the expected results provided by the gatingMLData are utilized in performing the compliance tests. The results obtained are compared with the expected results and an HTML summary report is generated.

Usage

```
testGatingMLCompliance(file = "GatingMLComplianceReport", version = 2.0)
```

Arguments

`file`	Name of the file in which the generated Gating-ML compliance report is to be saved. The .html extension will be added.
`version`	The Gating-ML version that is supposed to be checked. Currently, versions 1.5 and 2.0 are supported.

Details

The testGatingMLCompliance function depends on the gatingMLData data package for performing the compliance tests.

Author(s)

Spidlen J., Gopalakrishnan N.

References

Spidlen J, ISAC DSTF, Brinkman RR. 2014.
Gating-ML 2.0. International Society for Advancement of Cytometry (ISAC) standard for representing gating descriptions in flow cytometry.
http://flowcyt.sf.net/gating/20141009.pdf
http://flowcyt.sf.net/gating/20141009.full.zip

Spidlen J, Leif RC, Moore W, Roederer M; International Society for the Advancement of Cytometry Data Standards Task Force, Brinkman RR.
Gating-ML: XML-based gating descriptions in flow cytometry.
Cytometry A. 2008 Dec; 73A(12):1151–7. doi: 10.1002/cyto.a.20637.

Spidlen J, ISAC DSTF, Brinkman RR. 2008.
Gating-ML Candidate Recommendation for Gating Description in Flow Cytometry version 1.5.
http://flowcyt.sf.net/gating/Gating-ML.v1.5.081030.pdf
http://flowcyt.sf.net/gating/Gating-ML.v1.5.081030.full.zip
http://flowcyt.sf.net/gating/Gating-ML.v1.5.081030.Compliance-tests.081030.zip

Examples

```
## Not run:
#####################################################################
### Performs Gating-ML 1.5 compliance tests and                   #
### writes the results to Gating-ML_1.5_Compliance_Report.html #
#####################################################################
testGatingMLCompliance("Gating-ML_1.5_Compliance_Report", version = 1.5)

#####################################################################
### Performs Gating-ML 2.0 compliance tests and                   #
### writes the results to Gating-ML_2.0_Compliance_Report.html #
#####################################################################
testGatingMLCompliance("Gating-ML_2.0_Compliance_Report", version = 2.0)

## End(Not run)
```

---

write.gatingML *Function to write a Gating-ML XML file based on gating and transformation objects stored in an R environment.*

Description

This function saves gating and transformation objects stored in an R environment to a Gating-ML 2.0 XML file. The objects expected and supported in the R environment are those that can normally be created by the read.gatingML function when a Gating-ML 2.0 XML file is read.

Usage

```
write.gatingML(flowEnv, file = NULL)
```

Arguments

flowEnv	The R environment that is being searched for gating objects and transformations
file	The name of the output Gating-ML XML file. The standard output will be used if file is NULL.

Details

The Gating-ML specification has been developed as an interchange format for the description of gates relevant to a flow cytometry experiment. Presently, flowUtils can read Gating-ML versions 1.5 and 2.0 of the specification (see read.gatingML). Gating-ML version 2.0 only is being used when saving Gating-ML.

Author(s)

Spidlen, J.

References

Spidlen J, ISAC DSTF, Brinkman RR. 2014.
Gating-ML 2.0. International Society for Advancement of Cytometry (ISAC) standard for representing gating descriptions in flow cytometry.
http://flowcyt.sf.net/gating/20141009.pdf
http://flowcyt.sf.net/gating/20141009.full.zip

See Also

read.gatingML

Examples

```
library("flowCore")

#####################################################################
# Read a Gating-ML file and write the objects back in Gating-ML #
#####################################################################
flowEnv <- new.env()
gateFile <- system.file("extdata/Gml2/Gating-MLFiles",
  "gates1.xml", package="gatingMLData")
read.gatingML(gateFile, flowEnv)
ls(flowEnv)
write.gatingML(flowEnv)

################################################
# Create a quad gate and write it to Gating-ML #
################################################
flowEnv=new.env()
myQuad <- quadGate(filterId = "myQuad", "FSC-A" = 15000,
  "SSC-A" = 16000)
flowEnv[['myQuad']] <- myQuad
write.gatingML(flowEnv)
############################################
# If we wanted the output to a file instead: #
############################################
gatingOutputFile <- tempfile(fileext=".gating-ml2.xml")
write.gatingML(flowEnv, gatingOutputFile)
```

```
################################################
# Again a quad gate, but now adding compensation #
################################################
flowEnv=new.env()
myCompQuad <- quadGate(filterId = "myCompQuad", "PE-A" = 100,
  "PerCP-Cy5-5-A" = 200)
compPars = list(
  compensatedParameter(parameters="PE-A", spillRefId="SpillFromFCS",
    transformationId=paste("PE-A", "_compensated_according_to_FCS",
    sep=""), searchEnv=flowEnv),
  compensatedParameter(parameters="PerCP-Cy5-5-A", spillRefId="SpillFromFCS",
    transformationId=paste("PerCP-Cy5-5-A", "_compensated_according_to_FCS",
    sep=""), searchEnv=flowEnv)
)
myCompQuad@parameters = new("parameters", compPars)
flowEnv[['myCompQuad']] <- myCompQuad
write.gatingML(flowEnv)

#############################################################
# Again a quad gate, but now adding a scaling transformation #
#############################################################
flowEnv=new.env()
myTrQuad <- quadGate(filterId = "myTrQuad", "APC-A" = 0.5, "APC-Cy7-A" = 0.5)
trArcSinH1 = asinhtGml2(parameters = "APC-A",
  T = 1000, M = 4.5, A = 0, transformationId="trArcSinH1")
trLogicle1 = logicletGml2(parameters = "APC-Cy7-A",
  T = 1000, W = 0.5, M = 4.5, A = 0, transformationId="trLogicle1")
flowEnv[['trArcSinH1']] <- trArcSinH1
flowEnv[['trLogicle1']] <- trLogicle1
trPars = list(
  transformReference("trArcSinH1", flowEnv),
  transformReference("trLogicle1", flowEnv)
)
myTrQuad@parameters = new("parameters", trPars)
flowEnv[['myTrQuad']] <- myTrQuad
write.gatingML(flowEnv)

#########################################################################
# Now, we will be adding both scaling transformation and compensation #
# Also demonstrating what happens if 'bad' characters are part of the #
# name                                                                  #
#########################################################################
flowEnv=new.env()
myTrCompQuad <- quadGate(filterId = "myTr!Comp Quad", "APC-A" = 0.5,
  "APC-Cy7-A" = 0.5)
trArcSinH2 = asinhtGml2(parameters = "APC-A",
  T = 1000, M = 4, A = 0, transformationId="trArcSinH2")
trLogicle2 = logicletGml2(parameters = "APC-Cy7-A",
  T = 1000, W = 0.3, M = 4.5, A = 0, transformationId="trLogicle2")
trArcSinH2@parameters = compensatedParameter(parameters="APC-A",
  spillRefId="SpillFromFCS", transformationId=paste("FL3-H",
  "_compensated_according_to_FCS", sep=""), searchEnv=flowEnv)
trLogicle2@parameters = compensatedParameter(parameters="APC-Cy7-A",
  spillRefId="SpillFromFCS", transformationId=paste("FL4-H",
  "_compensated_according_to_FCS", sep=""), searchEnv=flowEnv)
```

```
trPars = list(trArcSinH2,trLogicle2)
myTrCompQuad@parameters = new("parameters", trPars)
flowEnv[['myTr!Comp Quad']] <- myTrCompQuad
write.gatingML(flowEnv)

##############################################################
# Creating a rectangle gate on a ratio of two parameters and #
# saving the result to a Gating-ML file.                     #
##############################################################
flowEnv=new.env()
rat1 <- ratio("FSC-A", "SSC-A", transformationId = "rat1")
gate1 <- rectangleGate(filterId="gate1", "rat1"=c(0.8, 1.4))
gate1@parameters = new("parameters", list(rat1))
flowEnv[['gate1']] <- gate1
trArcSinH = asinhtGml2(parameters = "rat2",
  T = 1000, M = 4.5, A = 0, transformationId="trArcSinH")
rat2 <- ratio("FSC-A", "APC-A", transformationId = "rat2")
trArcSinH@parameters = rat2
gate2 <- rectangleGate(filterId="gate2", "rat2"=c(0.6, 1.3))
gate2@parameters = new("parameters", list(trArcSinH))
flowEnv[['gate2']] <- gate2
write.gatingML(flowEnv)

##########################################################
# Example with an ellipse gate on compensated parameters #
##########################################################
flowEnv <- new.env()
covM <- matrix(c(62.5, 37.5, 37.5, 62.5), nrow = 2, byrow=TRUE)
colnames(covM) <- c("FL1-H", "FL2-H")
compPars <- list(
  compensatedParameter(parameters="FL1-H", spillRefId="SpillFromFCS",
    transformationId=paste("FL1-H", "_compensated_according_to_FCS", sep=""),
    searchEnv=flowEnv),
  compensatedParameter(parameters="FL2-H", spillRefId="SpillFromFCS",
    transformationId=paste("FL2-H", "_compensated_according_to_FCS", sep=""),
    searchEnv=flowEnv)
)
myEl <- ellipsoidGate(mean=c(12, 16), distance=1, .gate=covM, filterId="myEl")
myEl@parameters <- new("parameters", compPars)
flowEnv[['myEl']] <- myEl
write.gatingML(flowEnv)

#############################################################################
# Creating some Boolean gates and saving the result to a Gating-ML file. #
#############################################################################
flowEnv=new.env()
rg1 <- rectangleGate(filterId="rg1", list("FSC-A"=c(0200, 16000),
  "SSC-A"=c(0, 34000)))
rg2 <- rectangleGate(filterId="rg2", list("PE-A"=c(100, 8000),
  "APC-Cy7-A"=c(0, 59000)))
orGate <- new("unionFilter", filterId="orGate",
  filters=list(rg1, rg2))
flowEnv[['orGate']] <- orGate
andGate <- new("intersectFilter", filterId="andGate",
  filters=list(rg1, rg2))
flowEnv[['andGate']] <- andGate
notGate <- new("complementFilter", filterId="notGate",
```

```
  filters=list(rg1))
flowEnv[['notGate']] <- notGate
parentGate <- new("subsetFilter", filterId="parentGate",
  filters=list(rg1, rg2))
flowEnv[['parentGate']] <- parentGate
write.gatingML(flowEnv)
################################################
# Or if we wanted to write to a file instead... #
################################################
gatingOutputFile <- tempfile(fileext=".gating-ml2.xml")
write.gatingML(flowEnv, gatingOutputFile)

#############################################################################
# A few of the Gating-ML 1.5 transforms can be converted to Gating-ML 2.0 #
# and therefore be used with the write.gatingML function.                 #
#############################################################################
flowEnv=new.env()
trArcSinHGml1.5 = asinht(parameters = "APC-A", a = 1, b = 1,
  transformationId="trArcSinHGml1.5")
gateAsinhGml1.5 <- rectangleGate(filterId="gateAsinhGml1.5",
  "trArcSinHGml1.5"=c(0.3, 4.7))
gateAsinhGml1.5@parameters = new("parameters", list(trArcSinHGml1.5))
flowEnv[['gateAsinhGml1.5']] <- gateAsinhGml1.5
gatingOutputFile <- tempfile(fileext=".gating-ml2.xml")
write.gatingML(flowEnv, gatingOutputFile)
```

Index