rfcdmin

October 25, 2011

`FCSRobj.currentmin` *Script using current version of 'rflowcyt' package to read in FCS R*

Description

These two FCS R objects are read in as S4 FCS-class R objects using the 'read.FCS' function in the current version of **rflowcyt** package.

Usage

```
data(SEB.NP22.FCSRobj.currentmin)
data(facscan256.FCSRobj.currentmin)
```

Format

Two FCS-class R objects.

Source

Raw binary fcs files (in the "fcs" directory of 'rfcdmin' package) read in by using 'read.FCS' in the current version of **rflowcyt**.

0. FCS R object name (Raw Binary FCS File): Source

1. facscan256.FCSRobj ("facscan256.fcs"): Australian National University (ANU)

2. SEB.NP22.FCSRobj ("SEB-NP22.fcs"): Fred Hutchinson Cancer Research Ceneter : (FHCRC)

References

A.J. Rossini, J.Y. Wan

The FCS R objects correspond to the examples in the following documentation of the raw fcs files : `SEB-NP22.fcs`, `facscan256.fcs`,`FCSRobjmin` Function 'read.FCS' in **rflowcyt**

See documentation for 'FCSRobj.current' in **rfcdorig** package.

Examples

```
data(SEB.NP22.FCSRobj.currentmin)
data(facscan256.FCSRobj.currentmin)
```

`FCSRobjmin` *Archived FCS R OBJECTS corresponding to the binary FCS files*

Description

These FCS R objects are archived S4 FCS-class R objects.

Usage

```
data(FCSRobjmin)
```

Format

Two FCS-class R objects.

Source

Archived on April 20, 2004 by using 'read.FCS' in **rflowcyt**.

0. FCS R object name (Raw Binary FCS File): Source

1. facscan256.FCSRobj ("facscan256.fcs"): Australian National University (ANU)

Archived on September 14, 2004 by using 'read.FCS' in **rflowcyt**.

0. FCS R object name (Raw Binary FCS File): Source

2. SEB.NP22.FCSRobj ("SEB-NP22.fcs"): FHCRC

Also see 'FCSRobj' data in **rfcdorig** data package.

References

A.J. Rossini, J.Y. Wan

See also `SEB-NP22.fcs`, `facscan256.fcs`,`FCSRobj.currentmin` Function 'read.FCS' in **rflowcyt**

Examples

```
data(FCSRobjmin)
```

MC.053min *Archived Flow Cytometry Standard (FCS) R-object -Fred Hutchinson Cancer*

Description

Archived using the version of 'read.FCS' in **rflowcyt** on April 20, 2004.

There is one FCS R-object:

1. MC.053 (read from 042402c1.053.fcs, FCS file; ICS data file; analysis using Cell Quest application) negative control

Usage

```
## FCS R objects, see examples for details
```

Format

Three FCS R-objects

Source

Fred Hutchinson Cancer Research Center, Seattle, WA

Also from the 'FHCRC' data from **rfcdorig** data package.

Examples

```
##obtaining the FCS objects from FHCRC.RData compressed data file
## this is the long way
#location<-system.file("data", package="rfcdmin")
#filelocFHCRC<-paste(location, "/MC.053min.RData", sep="")
#load(filelocFHCRC)

## easier way is to just use data() function call

data(MC.053min)


## can now obtain FCS R object:
##  MC.053
## in the parent.frame()
```

PRIM.example.data *PRIM example: Simulated clustered data*

Description

PRIM-2Dimensional Dataset:

The simulated data comprises of three actual normally distributed clusters, denoted by 1, 2, and 3, which are true category statuses denoted by the vector 'true.Y.status.PRIM'. In reality, the truth is unknown, and thus, the information about the true category status is hidden. The rules obtained from PRIM will be compared to the true cluster categories, from which the data was generated.

The stochastic structure of the data is a mixture model of normals. In our test dataset there were 1000 observations.

Each of the 1000 'Y.PRIM' response comes from independent bernoulli distributions with a probability of 0.5.

The 'X.PRIM' observations with 2 column variables and 1000 row observations was made up of the following distributions conditional on the response 'Y.PRIM':

[True Cluster 1] 1/4 of the observations when the response Y.PRIM=1, came from the following Multivariate Normal distribution.

X.PRIM | Y.PRIM=1 ~ N([500,500],[[625, 0],[0, 625]])

[True Cluster 2] 1/4 of the observations when the response Y.PRIM=1 came from the following Multivariate Normal distribution.

X.PRIM | Y.PRIM=1 ~ N([400,100],[[625, 0],[0, 625]])

[True Cluster 3] 1/4 of the observations when the response Y.PRIM=1 came from the following Multivariate Normal distribution.

X.PRIM | Y.PRIM=1 ~ N([200,900],[[625, 0],[0, 625]])

[Noise] Each of the two column variables for 1/4 of the observations when the response Y.PRIM=1 and all observations when Y.PRIM=0 came from the following uniform distribution with a range of integer values between 0 and 1023.

X1|Y.PRIM=1,Y.PRIM=0 ~ U(0,1023)

X2|Y.PRIM=1,Y.PRIM=0 ~ U(0,1023)

PRIM-4Dimensional Dataset:

The simulated data comprises of three actual clusters, denoted by 1, 2, and 3, that are normally distributed. These values for true category status are denoted by the vector true.Y.status.PRIM4D. In reality, the truth is unknown, and thus, the information about the true category status is hidden. The rules obtained will be compared to the true cluster categories, from which the data was generated.

The stochastic structure of the data is a mixture model of normals. In our test dataset there were 2000 observations.

Each of the 2000 Y.PRIM4D response comes from independent bernoulli distributions with a probability of 0.5.

The X.PRIM4D matrix observations with 4 column variables and 2000 row observations was made up of the following distributions conditional on the response Y.PRIM4D:

[True Cluster 1] 1/4 of the observations when the response Y.PRIM4D=1, came from the following Multivariate Normal distribution.

X.PRIM4D | Y.PRIM4D=1 ~ N([500,500,500,500], [[625, 0,0,0],[0,625,0,0], [0,0,625,0],[0,0,0,625]])

[True Cluster 2] 1/4 of the observations when the response Y.PRIM4D=1 came from the following Multivariate Normal distribution.

X.PRIM4D | Y.PRIM4D=1 ~ N([200,1000,200,800], [[700, 0,0,0],[0,200,0,0], [0,0,100,0],[0,0,0,700]])

[True Cluster 3] 1/4 of the observations when the response Y.PRIM4D=1 came from the following Multivariate Normal distribution.

X.PRIM4D | Y.PRIM4D=1 ~ N([1000,100,900,100], [[400, 0,0,0],[0,500,0,0], [0,0,600,0],[0,0,0,300]])

[Noise] Each of the two column variables for 1/4 of the observations when the response Y.PRIM4D=1 and all observations when Y.PRIM4D=0 came from the following uniform distribution with a range of integer values between 0 and 1023. Let Xi denote the i-th column of X.PRIM4D.

X1|Y.PRIM4D=1,Y.PRIM4D=0 ~ U(0,1023)

X2|Y.PRIM4D=1,Y.PRIM4D=0 ~ U(0,1023)

X3|Y.PRIM4D=1,Y.PRIM4D=0 ~ U(0,1023)

X4|Y.PRIM4D=1,Y.PRIM4D=0 ~ U(0,1023)

Usage

```
data(PRIM.example.data)
```

Format

X.PRIM A matrix of 2 columns named "X1" and "X2" with 1000 row observations.

Y.PRIM A vector denoting the observed response corresponding to each row observation of 'X.PRIM' and having a length of 1000

true.Y.status.PRIM A vector denoting the true cluster category corresponding to each row observation of 'X.PRIM' and having a length of 1000

X.PRIM4D A matrix of 4 columns named "X1", "X2", "X3", and "X4" with 2000 row observations.

Y.PRIM4D A vector denoting the observed response corresponding to each row observation of 'X.PRIM' and having a length of 2000

true.Y.status.PRIM4D A vector denoting the true cluster category corresponding to each row observation of 'X.PRIM' and having a length of 2000

Source

A dataset generated with the following code in the "example" section.

References

See also **rfcprim** for examples using this simulated data.

Examples

```
data(PRIM.example.data)

 ## The following is the source R code
 ## setting the seed will generate the same dataset
if (FALSE){
  if (require(MASS)){
```

```
  set.seed(20)
  ## Y.PRIM response binary data
  Y.PRIM <- rbinom(1000, 1, 0.5)

  ## X.PRIM matrix, rows corresponding to Y.PRIM
  ##           columns are the variables

  ## when Y.PRIM==1 there is a MVN distribution of the X.PRIM
  ## There is simulation for 3 clusters

  Sigma <- matrix(c(25,0,0,25),2,2)^2

  X1.y1 <- mvrnorm(n=ceiling(length(Y.PRIM[Y.PRIM==1])/4),
             c(500, 500), Sigma, empirical = FALSE)

  X2.y1 <- mvrnorm(n=ceiling(length(Y.PRIM[Y.PRIM==1])/4),
             c(400, 100), Sigma, empirical = FALSE)

  X3.y1 <- mvrnorm(n=ceiling(length(Y.PRIM[Y.PRIM==1])/4),
            c(200, 900), Sigma, empirical = FALSE)

  X4.1.y1 <-  runif(ceiling(length(Y.PRIM[Y.PRIM==1])/4), 0, 1023)

  X4.2.y1 <-  runif(ceiling(length(Y.PRIM[Y.PRIM==1])/4), 0, 1023)
  X4.y1 <- cbind(X4.1.y1, X4.2.y1)

  X.y1 <- rbind(X1.y1, X2.y1, X3.y1, X4.y1)[1:length(Y.PRIM[Y.PRIM==1]),]


  ## when Y.PRIM==0 there is only a uniform distribution of X's
  X1.y0 <- runif(length(Y.PRIM[Y.PRIM==0]), 0, 1023)
  X2.y0 <- runif(length(Y.PRIM[Y.PRIM==0]), 0, 1023)
  X.y0 <- cbind(X1.y0, X2.y0)

  ## true Y.PRIM cluster status:
  ## 0, 4 is noise
  ## 1, 2, 3, are the true clusters

  true.Y.status.PRIM <- c(rep(2, dim(X1.y1)[1]),
  rep(1, dim(X2.y1)[1]), rep(3, dim(X3.y1)[1]),
  rep(4, dim(X4.y1)[1]))[1:length(Y.PRIM[Y.PRIM==1])]

  true.Y.status.PRIM <- c(true.Y.status.PRIM, rep(0,dim(X.y0)[1]))

  ## sort the Y.PRIM to correspond with the X.PRIM matrix
  Y.PRIM<- sort(Y.PRIM, decreasing=TRUE)
  X.PRIM <- rbind(X.y1, X.y0)
  colnames(X.PRIM) <- c("X1", "X2")
}
 ## 4D Dataset

if (require(MASS)==TRUE){
  Y.PRIM4D <- rbinom(2000, 1, 0.5)
  ## when Y.PRIM4D==1 there is a MVN distribution of the X
  Sigma1 <- rbind(c(625,0,0,0),
                  c(0, 625, 0, 0),
                  c(0, 0, 625, 0),
```

```
                  c(0, 0, 0, 625))
    Sigma2 <- rbind(c(700, 0, 0, 0),
                  c(0, 200, 0, 0),
                  c(0, 0, 100,0),
                  c(0, 0, 0, 700))
    Sigma3 <- rbind(c(400,0,0,0),
                  c(0,500,0,0),
                  c(0,0,600,0),
                  c(0, 0,0,300))

    X1.y1 <- mvrnorm(n=ceiling(length(Y.PRIM4D[Y.PRIM4D==1])/4),
                  c(500, 500, 500, 500), Sigma1, empirical = FALSE)
    X2.y1 <- mvrnorm(n=ceiling(length(Y.PRIM4D[Y.PRIM4D==1])/4),
                  c(200, 1000, 200, 800), Sigma2, empirical = FALSE)
    X3.y1 <- mvrnorm(n=ceiling(length(Y.PRIM4D[Y.PRIM4D==1])/4),
                  c(1000, 100, 900, 100), Sigma3, empirical = FALSE)

    X4.1.y1 <-  runif(ceiling(length(Y.PRIM4D[Y.PRIM4D==1])/4), 0, 1023)
    X4.2.y1 <-  runif(ceiling(length(Y.PRIM4D[Y.PRIM4D==1])/4), 0, 1023)
    X4.3.y1 <-  runif(ceiling(length(Y.PRIM4D[Y.PRIM4D==1])/4), 0, 1023)
    X4.4.y1 <-  runif(ceiling(length(Y.PRIM4D[Y.PRIM4D==1])/4), 0, 1023)

    X4.y1 <- cbind(X4.1.y1, X4.2.y1, X4.3.y1, X4.4.y1)

    X.y1 <- rbind(X1.y1, X2.y1, X3.y1, X4.y1)[1:length(Y.PRIM4D[Y.PRIM4D==1]),]
    ## when Y.PRIM4D==0 there is only a uniform distribution of X's
    X1.y0 <- runif(length(Y.PRIM4D[Y.PRIM4D==0]), 0, 1023)
    X2.y0 <- runif(length(Y.PRIM4D[Y.PRIM4D==0]), 0, 1023)
    X3.y0 <- runif(length(Y.PRIM4D[Y.PRIM4D==0]), 0, 1023)
    X4.y0 <- runif(length(Y.PRIM4D[Y.PRIM4D==0]), 0, 1023)
    X.y0 <- cbind(X1.y0, X2.y0, X3.y0, X4.y0)
    ## true if in a cluster otherwise FALSE
    true.Y.status.PRIM4D <- c(rep(2, dim(X1.y1)[1]), rep(1, dim(X2.y1)[1]), rep(3, dim(X3
                        rep(4, dim(X4.y1)[1]))[1:length(Y.PRIM4D[Y.PRIM4D==1])]
    ## NOTE: true.Y.status.PRIM4D=0,4 is uniform, random noise; true.Y.status.PRIM4D=1,2,
    ## denotes the real cluster category
    true.Y.status.PRIM4D <- c(true.Y.status.PRIM4D, rep(0,dim(X.y0)[1]))
    Y.PRIM4D <- sort(Y.PRIM4D, decreasing=TRUE)
    X.PRIM4D <- rbind(X.y1, X.y0)
    colnames(X.PRIM4D) <- c("X1", "X2", "X3", "X4")
    }
  }
```

`PRIM.real.data.min` *PRIM output from VRCmin data analysis*

Description

The HIV-protein stimulated st.1829 FCS R-object and the unstimulated unst.1829 FCS R-object from the VRCmin data were used to exemplify the PRIM algorithm on a real dataset. The three steps for the finding of a single rule were saved as output. The out.peel is the 'PRIM.step' class object of the Peeling step. The out.expand is the 'PRIM.step' class object of the Expansion Step. The out.cv is the 'PRIM.crossval.step' class object of the cross-validation step. From these output objects we obtain the necessary indices and vectors for making plots that are exemplified in PRIM.pdf.

Usage

```
data(PRIM.real.data.min)
```

Format

box.idx.list Peeling Step: a list of five boxes that were in the peeling sequence of 'out.peel' which used the whole data 'X'

box.support.PS Peeling Step: The vector of proportions of X that lie in the peeled box for each iteration of the expansion step from 'out.peel'

means.PS Peeling Step: The vector of means(Y) of each peeled box for each iteration from 'out.peel'

type.PS Peeling Step: The vector of peeling types, denoting the direction and variable for the addition of new points into the box for each iteration from 'out.peel'

box.support.ES Expansion Step: The vector of proportions of X that lie in the expanded box for each iteration of the expansion step from 'out.expand'

means.ES Expansion Step: The vector of means(Y) of each expanded box for each iteration from 'out.expand'

type.ES Expansion Step: The vector of expansion types, denoting the direction and variable for the addition of new points into the box for each iteration from 'out.expand'

best.box.X.ranges Expansion Step: The X-variable ranges of the final rule estimated after the Expansion Step from 'out.expand'

Rule1.idx Expansion Step: The index of the final rule obtained after the expansion step from out.expand

TD1 Cross-Validation Step: vector of positional row indices corresponding to the first testdata set which is subset from the original 'X' data (which is constructed in the example below)

TD2 Cross-Validation Step: vector of positional row indices corresponding to the second testdata set which is subset from the original 'X' data (which is constructed in the example below)

box.idx.listTD1 Cross-Validation Step: a list of five boxes in the peeling sequence of the first Testdata set

box.idx.listTD2 Cross-Validation Step: a list of five boxes in the peeling sequence of the second Testdata set

Details

The class information, extraction, plotting, implementation tools for the 'PRIM.step' and the 'PRIM.crossval.step' class objects are detailed in the **rfcprim** package.

Source

See VRCmin

References

See VRCmin and **rfcprim**.

Examples

```
data(PRIM.real.data)

## the following code was used to generate the output
if (FALSE){

  data(VRCmin)
  ## the HIV-protein stimulation status
  Y <- c(rep(1, dim.FCS(st.1829)[1]),
       rep(0, dim.FCS(unst.1829)[1]))

  ## the dataset
  X <- rbind(as(st.1829, "matrix"), as(unst.1829, "matrix"))
  if (require(rfcprim)){
  out.peel <- peel.step(X,
                        Y,
                        min.box.size = 500,
                        alpha=0.10,
                        verbose=TRUE)


  out.expand <- expand.step(X,
                            Y,
                            out.peel@best.box.idx,
                            beta=0.01,
                            verbose=TRUE)


  out.cv <- crossval.step(X,
                          Y,
                          num.testdata=2,
                          prob.testdata=c(0.50, 0.50),
                          alpha=0.10,
                          target.mu.Y=1,
                          min.box.size=500,
                          choose.best.box.decision=c("max.mean.box", "final.box"),

                          beta=0.01,
                          verbose=TRUE)

  box.idx.list <- list(out.peel@best.box.idx,
     out.expand@best.box.idx)

  box.support.PS <- out.peel@box.support.vec

  means.PS <- out.peel@means.vec

  type.PS <- out.peel@type.vec

  box.support.ES <- out.expand@box.support.vec

  means.ES <- out.expand@means.vec

  type.ES <- out.expand@type.vec

  best.box.X.ranges <- out.expand@best.box.X.ranges
```

```
 Rule1.idx <- out.expand@best.box.idx

 TD1 <- out.cv[1, "testdata.idx.list"]
 TD2 <- out.cv[2, "testdata.idx.list"]

 box.idx.listTD1 <- list(out.cv[1,"step.set.list"]@peel.step@best.box.idx,
     out.cv[1, "step.set.list"]@expand.step@best.box.idx)

 box.idx.listTD2 <- list(out.cv[2,"step.set.list"]@peel.step@best.box.idx,
     out.cv[2, "step.set.list"]@expand.step@best.box.idx)
 }
}
```

`PerPosROCmin` *Vaccine Research Center (VRC), NIH, Bethesda, MD data-Percentage*

Description

Note: Percentage Positives = 100*(Percent Positives) of the stimulated sample based on the 99.9th percentile of the unstimulated distribution.

For each sample/individual of the Interferon Gamma variable (cytokine repsonses), a 100*percent positive is defined as 100* the ratio of GAG stimulated cells greater than the 99.9th percentile of corresponding sample of unstimulated cells' (from the same given individual) distribution over the total number of GAG stimulated cells in the given sample from the individual.

Thus, for a given individual, 100*Percent Positive = 100*k/n, where k is the number of stimulated cells >= 99.9th percentile of the corresponding distribution of the unstimulated cells of the same individual and n is the total number of stimulated cells in the sample.

The following are the 100*(percent positives) for different peptide stimulations (GAG, PolA, or PolB) and different individuals (HIV positive & HIV negative).

hivpos.gag 100*(Percent Positives) of the GAG stimulated cells from 29 HIV positive individuals

hivneg.gag 100*(Percent Positives) of the GAG stimulated cells from 10 HIV negative individuals

hivpos.pola 100*(Percent Positives) of the PolA stimulated cells from 29 HIV positive individuals

hivneg.pola 100*(Percent Positives) of the PolA stimulated cells from 10 HIV negative individuals

hivpos.polb 100*(Percent Positives) of the PolB stimulated cells from 29 HIV positive individuals

hivneg.polb 100*(Percent Positives) of the PolB stimulated cells from 10 HIV negative individuals

Usage

```
data(PerPosROC)
```

Format

Four numeric vectors

Source

Cytokine reponse Table from Zoe Moodie PhD who obtained the percent positives from the data from the Vaccine Research Center (VRC), NIH, Bethesda, MD

Also from the 'PerPosROC' data from the **rfcdorig** data package.

Examples

```
   data(PerPosROCmin)
if (require(rflowcyt)){
## the example below comes from ROC.FCS function

#plotting the gag stimulated 100* percent positives
   if (interactive()==TRUE){
      GAG<-ROC.FCS(hivpos.gag, hivneg.gag)
#plotting the pola stimulated 100* percent positives
      POLA<-ROC.FCS(hivpos.pola, hivneg.pola, lineopt=2, colopt=2, overlay=TRUE)
#plotting the polb stimulated 100* percent positives
      POLB<-ROC.FCS(hivpos.polb, hivneg.polb, lineopt=4, colopt=3, overlay=TRUE)
      legend(0.7, 0.7, c("gag", "polA", "polB"), col = c(1,2,3), lty=c(1,2,4))
    }
} else {
cat("Installation of the rflowcyt package is needed to run the examples.", "\n\n")
}
```

SEB-NP22.fcs *SEB-NP22.fcs, Flow Cytometry Standard files version 3.0, Fred*

Description

This file is obtained from the FHCRC.

Usage

```
## see the examples
```

Format

A binary file with text heading.

Source

HVTN, FHCRC.

Also from the 'fcs' directory of **rfcdfhcrc2** data package.

Examples

```
if (FALSE) {
  ## This example always throws an error; it is not real, though!
  if (require(rflowcyt)){
    location<-system.file("fcs", package="rfcdmin")
    SEB.NP22.FCSRobj.current <- read.FCS(paste(location,
                                               "/SEB-NP22.fcs",
                                               sep=""))
  } else {
    cat("rflowcyt package is needed to run the examples.", "\n\n")
  }
}
```

VRCmin *Archived Flow Cytometry Standard (FCS) R-objects- Vaccine Research*

Description

Archived using the version of 'read.FCS' in **rflowcyt** on April 20, 2004.

There are four FCS R-objects:

1. unst.1829 (read from 1829_28+49d.fcs, FCS file): unstimulated HIV- individual (control)
2. st.1829 (read from 1829_GAG.fcs, FCS file): GAG peptide stimulated HIV- individual (case)
3. unst.DRT (read from DRT_28+49d.fcs, FCS file): unstimulated HIV+ individual (control)
4. st.DRT (read from DRT_GAG.fcs, FCS file): GAG peptide stimulated HIV+ individual (case)

Usage

```
data(VRCmin)
```

Format

Four FCS R-objects

Source

Vaccine Research Center (VRC), NIH, Bethesda, MD

Also from the 'VRC' data of the **rfcdorig** package.

Examples

```
data(VRCmin)

## can now obtain FCS R objects:
##    unst.1829, st.1829, unst.DRT, and st.DRT
## in the parent.frame()
```

facscan256.fcs *Flow Cytometry Standard (FCS) file version 2.0, University of*

Description

This FCS file is a binary file.

Usage

```
## data file, see examples
```

Format

A binary FCS file with a heading

Source

Australian National University

Examples

```
if (require(rflowcyt)){

  ## the following current FCS R-objects are also produced
  ## by using 'data(*)'
  ## where * is one of the following example object names below
  ## * = facscan256.FCSRobj.current, facscan1024.FCSRobj.current, etc.

  location<-system.file("fcs", package="rfcdmin")

  facscan256.FCSRobj.current <- read.FCS(paste(location, "/facscan256.fcs", sep=""),
                                  fcs.type=fcs.type.facscan256)


} else {
  cat("The rflowcyt package is required to run examples.\n\n")
}
```

`flowcyt.data` *Archived Flow Cytometry FCS objects - British Columbia Cancer Research*

Description

Archived flow cytometry FCS objects, from a serie of `FCS` files, using the version of 'read.series.FCS' in **rflowcyt** on October 1, 2005.

The data come from a time course experiment with 12 time points. At each time point, the sample has been divided in 8 aliquots and labeled to analyse 8 stains (with 4 markers for each stain). The name of one sample corresponds to its position in the 96 wells plate.

The flowcyt.fluors are the 96 `FCS` R-objects.

Usage

```
##FCS R objects, see examples for details
```

Format

List of FCS R objects

Source

British Columbia Cancer Research Center, Vancouver, Canada

Examples

```
data(flowcyt.data)
```

`flowcyt.fcs` *Flow Cytometry Standard (FCS) files from British Columbia Cancer*

Description

Flow Cytometry Standard (FCS) files 'FCS' objects, using the version of `read.series.FCS` in **rflowcyt** on October 1, 2005.

The data come from a screening of a compound library. The screening was performed on a cell line. The name of one file corresponds to the position of the sample (cell line aliquot) in a 96 wells plate.

Usage

```
## FCS files to be read with 'read.FCS' or 'read.series.FCS'
## function, see examples for details
```

Format

binary FCS files

Source

British Columbia Cancer Research Center, Vancouver, Canada

References

Gasparetto, M. *et. al.* (2004) Identification of compounds that enhance the anti-lymphoma activity of rituximab usinf flow cytometric high-content screening, *Journal of Immunollogical Methods*, **292**, 59-71.

Examples

```
## Example I
if (require(rflowcyt)) {

##obtaining the location of the fcs files in the data
 pathFiles <- system.file("bccrc", package="rfcdmin")
 drugFiles <- dir(pathFiles)

## reading in the FCS files
 drugData <- read.series.FCS(drugFiles, path=pathFiles, MY.DEBUG=FALSE)
}


## Example II
if (require(rflowcyt)) {
   ## obtaining the location of the fcs files in the data
   A06path <- paste(system.file("bccrc", package="rfcdmin"), "A06",sep="/")

   ## reading in the FCS files
   A06 <- read.FCS(A06path)
}
```

`flowcyt.fluors` *Archived Flow Cytometry Fluorescence Measurements - British Columbia*

Description

Archived flow cytometry fluorescence measurements, from a serie of `FCS` objects, using the version of 'read.series.FCS' in **rflowcyt** on October 1, 2005.

The data come from a time course experiment with 12 time points. At each time point, the sample has been divided in 8 aliquots and labeled to analyse 8 stains (with 4 markers for each stain). The name of one sample corresponds to its position in the 96 wells plate.

The flowcyt.fluors are the 96 fluorescence data extract from the 96 `FCS` R-objects.

Usage

```
##Fluorescence measurements of a FCS R objects, see examples for details
```

Format

Fluorescence data

Source

British Columbia Cancer Research Center, Vancouver, Canada

Examples

```
data(flowcyt.fluors)
```

Index