

# flowClust

April 19, 2009

---

Map, flowClust-method

*Cluster Assignment Based on Clustering Results*

---

## Description

This method performs cluster assignment according to the posterior probabilities of clustering memberships resulted from the clustering (filtering) operations. Outliers identified will be left unassigned by default.

## Usage

```
## S4 method for signature 'flowClust':  
Map(f, rm.outliers=TRUE, ...)
```

## Arguments

`f` Object returned from `flowClust` or `filter`.  
`rm.outliers` A logical value indicating whether outliers will be left unassigned or not.  
`...` Further arguments to be passed to or from other methods.

## Value

A numeric vector of size  $N$  (the number of observations) indicating to which cluster each observation is assigned. Unassigned observations will be labelled as NA.

## Note

Even if `rm.outliers` is set to FALSE, NA may still appear in the resultant vector due to the filtered observations; see the descriptions about the `min.count`, `max.count`, `min` and `max` arguments of `flowClust`.

## Author(s)

Raphael Gottardo <raph@stat.ubc.ca>, Kenneth Lo <c.lo@stat.ubc.ca>

## References

Lo, K., Brinkman, R. R. and Gottardo, R. (2008) Automated Gating of Flow Cytometry Data via Robust Model-based Clustering. *Cytometry A* **73**, 321-332.

## See Also

[flowClust](#), [filter](#), [posterior](#)

## Examples

```
res <- flowClust(iris[,1:4], K=3)
Map(res)
Map(res, rm.outliers=FALSE)
```

---

SimulateMixture      *Random Generation from a  $t$  Mixture Model*

---

## Description

This function can be used to generate a sample from a multivariate  $t$  mixture model.

## Usage

```
SimulateMixture(N, nu=4, mu, sigma, w)
```

## Arguments

N	The number of observations.
nu	The degrees of freedom used for the $t$ distribution.
mu	A matrix of size $K \times P$ , where $K$ is the number of clusters and $P$ is the dimension, containing the $K$ mean vectors.
sigma	An array of dimension $K \times P \times P$ , containing the $K$ covariance matrices.
w	A vector of length $K$ , containing the $K$ cluster proportions.

## Value

A matrix of size  $N \times P$ .

## Author(s)

Raphael Gottardo <[raph@stat.ubc.ca](mailto:raph@stat.ubc.ca)>, Kenneth Lo <[c.lo@stat.ubc.ca](mailto:c.lo@stat.ubc.ca)>

## See Also

[flowClust](#)

**Examples**

```

### Number of components
K <- 5
### Dimension
p <- 2
### Number of observations
n <- 200
Mu <- matrix(runif(K*p, 0, 20), K, p)
Sigma <- array(0, c(K, p, p))

for (k in 1:K)
{
  Sigma[k,,][outer(1:p, 1:p, ">")] <- runif(p*(p-1)/2,-.1,.1)
  diag(Sigma[k,,]) <- runif(p,0,1)
  ### Make sigma positive definite
  Sigma[k,,] <- Sigma[k,,] %*% t(Sigma[k,,])
}

### Generate the weights
w <- rgamma(K,10,1)
w <- w/sum(w)

y <- SimulateMixture(n, nu=4, Mu, Sigma, w)

```

---

Subset,flowClust-method

*Subsetting Data Based on Clustering Results*


---

**Description**

This method returns a subset of data upon the removal of outliers identified from the clustering (filtering) operations.

**Value**

An object which is a subset of `x`. It also retains the same class as `x`.

**Usage**

```
Subset(x, subset, ...)
```

**Arguments**

`x` A numeric vector, matrix, data frame of observations, or object of class `flowFrame`. This is the object on which `flowClust` or `filter` was performed.

`subset` Object returned from `flowClust` or `filter`.

`...` Further arguments to be passed to or from other methods.

**Author(s)**

Raphael Gottardo <raph@stat.ubc.ca>, Kenneth Lo <c.lo@stat.ubc.ca>

**References**

Lo, K., Brinkman, R. R. and Gottardo, R. (2008) Automated Gating of Flow Cytometry Data via Robust Model-based Clustering. *Cytometry A* **73**, 321-332.

**See Also**

`split`, `flowClust`, `filter`

---

box

*Box-Cox Transformation*

---

**Description**

This function performs Box-Cox transformation on the inputted data matrix.

**Usage**

```
box(data, lambda)
```

**Arguments**

<code>data</code>	A numeric vector, matrix or data frame of observations. Negative data values are permitted.
<code>lambda</code>	The transformation to be applied to the data. If negative data values are present, <code>lambda</code> has to be positive.

**Details**

To allow for negative data values, a slightly modified version of the original Box-Cox (1964) is used here. This modified version originated from Bickel and Doksum (1981), taking the following form:

$$f(y) = \frac{\text{sgn}(y)|y|^\lambda - 1}{\lambda}$$

When negative data values are involved, the transformation parameter,  $\lambda$ , has to be positive in order to avoid discontinuity across zero.

**Value**

A numeric vector, matrix or data frame of the same dimension as `data` is returned.

**References**

Bickel, P. J. and Doksum, K. A. (1981) An Analysis of Transformations Revisited. *J. Amer. Statist. Assoc.* **76**(374), 296-311.

Box, G. E. P. and Cox, D. R. (1964) An Analysis of Transformations. *J. R. Statist. Soc. B* **26**, 211-252.

**See Also**[rbox](#)**Examples**

```

data(rituximab)
data <- exprs(rituximab)
summary(data)
# Transform data using Box-Cox with lambda=0.3
dataTrans <- box(data, 0.3)
# Reverse transform data; this should return back to the original rituximab data
summary(rbox(dataTrans, 0.3))

```

---

density,flowClust-method

*Grid of Density Values for the Fitted  $t$  Mixture Model with Box-Cox Transformation*

---

**Description**

This method constructs the `flowDens` object which is used to generate a contour or image plot.

**Usage**

```

## S4 method for signature 'flowClust':
density(x, data=NULL, subset=c(1,2), include=1:(x@K),
        npoints=c(100,100), from=NULL, to=NULL)

```

**Arguments**

<code>x</code>	Object returned from <a href="#">flowClust</a> or from running <a href="#">filter</a> on a <code>flowFrame</code> object.
<code>data</code>	A matrix, data frame of observations, or object of class <code>flowFrame</code> . This is the object on which <code>flowClust</code> or <code>filter</code> was performed. If this argument is not specified, the grid square upon which densities will be computed must be provided (through arguments <code>from</code> and <code>to</code> ).
<code>subset</code>	A numeric vector of length two indicating which two variables are selected for the scatterplot. Alternatively, a character vector containing the names of the two variables is allowed if <code>x@varNames</code> is not <code>NULL</code> .
<code>include</code>	A numeric vector specifying which clusters are included to compute the density values. By default, all clusters are included.
<code>npoints</code>	A numeric vector of size two specifying the number of grid points in $x$ (horizontal) and $y$ (vertical) directions respectively.
<code>from</code>	A numeric vector of size two specifying the coordinates of the lower left point of the grid square. Note that, if this (and <code>to</code> ) is not specified, <code>data</code> must be provided such that the range in the two variables (dimensions) selected will be used to define the grid square.
<code>to</code>	A numeric vector of size two specifying the co-ordinates of the upper right point of the grid square.

**Details**

The `flowDens` object returned is to be passed to the `plot` method for generating a contour or image plot.

**Value**

An object of class `flowDens` containing the following slots is constructed:

<code>dx</code>	A numeric vector of length <code>npoints[1]</code> ; the $x$ -coordinates of the grid points.
<code>dy</code>	A numeric vector of length <code>npoints[2]</code> ; the $y$ -coordinates of the grid points.
<code>value</code>	A matrix of size <code>npoints[1] × npoints[2]</code> ; the density values at the grid points.

**Author(s)**

Raphael Gottardo <[raph@stat.ubc.ca](mailto:raph@stat.ubc.ca)>, Kenneth Lo <[c.lo@stat.ubc.ca](mailto:c.lo@stat.ubc.ca)>

**See Also**

[plot](#), [flowClust](#)

**Examples**

```
res <- flowClust(iris[,1:4], K=3)

res.den <- density(res, data=iris)
plot(res.den)
plot(res.den, scale="sqrt", drawlabels=FALSE)
plot(res.den, type="image", nlevels=100)
plot(res.den, type="image", scale="sqrt", nlevels=100,
      color="gray")

plot(density(res, data=iris, subset=c("Petal.Length",
  "Petal.Width"), include=c(2,3)))
plot(density(res, subset=c(3,4), include=c(2,3), from=c(3,0.75),
  to=c(7,2.5)))
```

**Description**

This function computes the densities at the inputted points of the multivariate  $t$  distribution with the specified means and covariance matrix.

**Usage**

```
dmvt(x, mu, sigma, nu, log=FALSE)
```

**Arguments**

<code>x</code>	A matrix or data frame of size $N \times P$ , where $N$ is the number of observations and $P$ is the dimension. Each row corresponds to one observation.
<code>mu</code>	A numeric vector of length $P$ specifying the mean.
<code>sigma</code>	A matrix of size $P \times P$ specifying the covariance matrix.
<code>nu</code>	The degrees of freedom used for the $t$ distribution. If <code>nu=Inf</code> , Gaussian distribution will be used.
<code>log</code>	A logical value. If <code>TRUE</code> then the logarithm of the densities is returned.

**Value**

A list with the following components:

<code>value</code>	A vector of length $N$ containing the density values.
<code>md</code>	A vector of length $N$ containing the Mahalanobis distances.

**Author(s)**

Raphael Gottardo <raph@stat.ubc.ca>, Kenneth Lo <c.lo@stat.ubc.ca>

---

flowClust-package *Clustering for Flow Cytometry*

---

**Description**

Robust model-based clustering using a  $t$  mixture model with Box-Cox transformation.

**Details**

Package: flowClust  
 Type: Package  
 Version: 1.7.0  
 Depends: R(>= 2.5.0), methods, mnormt, mclust, ellipse, flowCore  
 Collate: SetClasses.R SetMethods.R plot.R flowClust.R SimulateMixture.R  
 biocViews: Clustering, Statistics, Visualization  
 License: Artistic-2.0  
 Built: R 2.6.1; universal-apple-darwin8.10.1; 2008-03-26 20:54:42; unix

**Index**

[box](#) Box-Cox Transformation

[density, flowClust-method](#) Grid of Density Values for the Fitted  $t$  Mixture Model with Box-Cox Transformation

[dmvt](#) Density of the Multivariate  $t$  Distribution

`flowClust` Robust Model-based Clustering for Flow Cytometry

`hist, flowClust-method` 1-D Density Plot (Histogram) of Clustering Results

`Map, flowClust-method` Cluster Assignment Based on Clustering Results

`miscellaneous` Various Functions for Retrieving Information from Clustering Results

`plot, flowClust-method` Scatterplot of Clustering Results

`plot, flowDens-method` Contour or Image Plot of Clustering Results

`plot, flowFrame, tmixFilterResult-method` Scatterplot / 1-D Density Plot of Filtering (Clustering) Results

`rbox` Reverse Box-Cox Transformation

`ruleOutliers, flowClust-method` Showing or Modifying the Rule used to Identify Outliers

`show, flowClust-method` Show Method for `flowClust` / `tmixFilterResult` Object

`show, tmixFilter-method` Show Method for `tmixFilter` Object

`SimulateMixture` Random Generation from a  $t$  Mixture Model

`split, flowClust-method` Splitting Data Based on Clustering Results

`Subset, flowClust-method` Subsetting Data Based on Clustering Results

`summary, flowClust-method` Summary Method for `flowClust` Object

`tmixFilter` Creating Filters and Filtering Flow Cytometry Data

**Note**

Further information is available in the vignette.

**Author(s)**

Raphael Gottardo <raph@stat.ubc.ca>, Kenneth Lo <c.lo@stat.ubc.ca>

Maintainer: Raphael Gottardo <raph@stat.ubc.ca>

**References**

Lo, K., Brinkman, R. R. and Gottardo, R. (2008) Automated Gating of Flow Cytometry Data via Robust Model-based Clustering. *Cytometry A* **73**, 321-332.



**Description**

This function performs automated clustering for identifying cell populations in flow cytometry data. The approach is based on the  $t$  mixture model with the Box-Cox transformation, which provides a unified framework to handle outlier identification and data transformation simultaneously.

**Usage**

```
flowClust(x, expName="Flow Experiment", varNames=NULL, K, B=500,
          tol=1e-5, nu=4, lambda=1, trans=TRUE, min.count=10,
          max.count=10, min=NULL, max=NULL, level=0.9,
          u.cutoff=NULL, z.cutoff=0, randomStart=0,
          B.init=B, tol.init=1e-2, seed=1, criterion="BIC")
```

**Arguments**

<code>x</code>	A numeric vector, matrix, data frame of observations, or object of class <code>flowFrame</code> . Rows correspond to observations and columns correspond to variables.
<code>expName</code>	A character string giving the name of the experiment.
<code>varNames</code>	A character vector specifying the variables (columns) to be included in clustering. When it is left unspecified, all the variables will be used.
<code>K</code>	An integer vector indicating the numbers of clusters.
<code>B</code>	The maximum number of EM iterations.
<code>tol</code>	The tolerance used to assess the convergence of the EM.
<code>nu</code>	The degrees of freedom used for the $t$ distribution. Default is 4. If <code>nu=Inf</code> , Gaussian distribution will be used.
<code>lambda</code>	The initial transformation to be applied to the data.
<code>trans</code>	A logical value indicating whether the Box-Cox transformation parameter is estimated from the data.
<code>min.count</code>	An integer specifying the threshold count for filtering data points from below. The default is 10, meaning that if 10 or more data points are smaller than or equal to <code>min</code> , they will be excluded from the analysis. If <code>min</code> is <code>NULL</code> , then the minimum of data as per each variable will be used. To suppress filtering, set it as -1.
<code>max.count</code>	An integer specifying the threshold count for filtering data points from above. Interpretation is similar to that of <code>min.count</code> .
<code>min</code>	The lower boundary set for data filtering. Note that it is a vector of length equal to the number of variables (columns), implying that a different value can be set as per each variable.
<code>max</code>	The upper boundary set for data filtering. Interpretation is similar to that of <code>min</code> .
<code>level</code>	A numeric value between 0 and 1 specifying the threshold quantile level used to call a point an outlier. The default is 0.9, meaning that any point outside the 90% quantile region will be called an outlier.

<code>u.cutoff</code>	Another criterion used to identify outliers. If this is <code>NULL</code> , then <code>level</code> will be used. Otherwise, this specifies the threshold (e.g., 0.5) for $u$ , a quantity used to measure the degree of “outlyingness” based on the Mahalanobis distance. Please refer to Lo et al. (2008) for more details.
<code>z.cutoff</code>	A numeric value between 0 and 1 underlying a criterion which may be used together with <code>level/u.cutoff</code> to identify outliers. A point with the probability of assignment $z$ (i.e., the posterior probability that a data point belongs to the cluster assigned) smaller than <code>z.cutoff</code> will be called an outlier. The default is 0, meaning that assignment will be made no matter how small the associated probability is, and outliers will be identified solely based on the rule set by <code>level</code> or <code>cutoff</code> .
<code>randomStart</code>	A numeric value indicating how many times a random partition of the data is generated for initialization. For instance, if <code>randomStart</code> is 10, 10 random partitions of the data will be generated, each of which is followed by a short EM run. The partition leading to the highest likelihood value will be adopted to be the initial partition for the eventual long EM run. The default is 0, meaning that this initialization strategy is not applied and hierarchical clustering is used instead.
<code>B.init</code>	The maximum number of EM iterations following each random partition in random initialization.
<code>tol.init</code>	The tolerance used as the stopping criterion for the short EM runs in random initialization.
<code>seed</code>	An integer giving the seed number used when <code>randomStart</code> >0.
<code>criterion</code>	A character string stating the criterion used to choose the best model. May take either "BIC" or "ICL". This argument is only relevant when <code>length(K) &gt; 1</code> .

## Details

Estimation of the unknown parameters (including the Box-Cox parameter) is done via an Expectation-Maximization (EM) algorithm. At each EM iteration, Brent’s algorithm is used to find the optimal value of the Box-Cox transformation parameter. Conditional on the transformation parameter, all other estimates can be obtained in closed form. Please refer to Lo et al. (2008) for more details.

The **flowClust** package makes extensive use of the GSL as well as BLAS. If an optimized BLAS library is provided when compiling the package, the **flowClust** package will be able to run multi-threaded processes.

Various operations have been defined for the object returned from `flowClust`. These include:

Subsetting operations: `%in%`, `Subset` and `split`  
 Slot retrieval operations: `ruleOutliers`, `Map`, `criterion`, `posterior`, `importance`, `uncertainty` and `ge`  
 Graphical operations: `plot`, `density` and `hist`

In addition, to facilitate the integration with the **flowCore** package for processing flow cytometry data, the `flowClust` operation can be done through a method pair (`tmixFilter` and `filter`) such that various methods defined in **flowCore** can be applied on the object created from the filtering operation.

## Value

If  $K$  is of length 1, the function returns an object of class `flowClust` containing the following slots, where  $K$  is the number of clusters,  $N$  is the number of observations and  $P$  is the number of

variables:

<code>expName</code>	Content of the <code>expName</code> argument.
<code>varNames</code>	Content of the <code>varNames</code> argument if provided; generated if available otherwise.
<code>K</code>	An integer showing the number of clusters.
<code>w</code>	A vector of length $K$ , containing the estimates of the $K$ cluster proportions.
<code>mu</code>	A matrix of size $K \times P$ , containing the estimates of the $K$ mean vectors.
<code>sigma</code>	An array of dimension $K \times P \times P$ , containing the estimates of the $K$ covariance matrices.
<code>lambda</code>	The Box-Cox transformation parameter estimate.
<code>nu</code>	The degrees of freedom used for the $t$ distribution.
<code>z</code>	A matrix of size $N \times K$ , containing the posterior probabilities of cluster memberships. The probabilities in each row sum up to one.
<code>u</code>	A matrix of size $N \times K$ , containing the “weights” (the contribution for computing cluster mean and covariance matrix) of each data point in each cluster. Since this quantity decreases monotonically with the Mahalanobis distance, it can also be interpreted as the level of “outlyingness” of a data point. Note that, when <code>nu=Inf</code> , this slot is used to store the Mahalanobis distances instead.
<code>label</code>	A vector of size $N$ , showing the cluster membership according to the initial partition (i.e., hierarchical clustering if <code>randomStart</code> is <code>FALSE</code> ). Filtered observations will be labelled as <code>NA</code> . Unassigned observations (which may occur since only 1500 observations at maximum are taken for hierarchical clustering) will be labelled as 0.
<code>uncertainty</code>	A vector of size $N$ , containing the uncertainty about the cluster assignment. Uncertainty is defined as 1 minus the posterior probability that a data point belongs to the cluster to which it is assigned.
<code>ruleOutliers</code>	A numeric vector of size 3, storing the rule used to call outliers. The first element is 0 if the criterion is set by the <code>level</code> argument, or 1 if it is set by <code>u.cutoff</code> . The second element copies the content of either the <code>level</code> or <code>u.cutoff</code> argument. The third element copies the content of the <code>z.cutoff</code> argument. For instance, if points are called outliers when they lie outside the 90% quantile region or have assignment probabilities less than 0.5, then <code>ruleOutliers</code> is <code>c(0, 0.9, 0.5)</code> . If points are called outliers only if their “weights” in the assigned clusters are less than 0.5 regardless of the assignment probabilities, then <code>ruleOutliers</code> becomes <code>c(1, 0.5, 0)</code> .
<code>flagOutliers</code>	A logical vector of size $N$ , showing whether each data point is called an outlier or not based on the rule defined by <code>level/u.cutoff</code> and <code>z.cutoff</code> .
<code>rm.min</code>	Number of points filtered from below.
<code>rm.max</code>	Number of points filtered from above.
<code>logLike</code>	The log-likelihood of the fitted mixture model.
<code>BIC</code>	The Bayesian Information Criterion for the fitted mixture model.
<code>ICL</code>	The Integrated Completed Likelihood for the fitted mixture model.

If `K` has a length  $>1$ , the function returns an object of class `flowClustList`. Its data part is a list with the same length as `K`, each element of which is a `flowClust` object corresponding to a specific number of clusters. In addition, the resultant `flowClustList` object contains the following

slots:

`index` An integer giving the index of the list element corresponding to the best model as selected by criterion.

`criterion` The criterion used to choose the best model – either "BIC" or "ICL".

Note that when a `flowClustList` object is used in place of a `flowClust` object, in most cases the list element corresponding to the best model will be extracted and passed to the method/function call.

### Author(s)

Raphael Gottardo <[raph@stat.ubc.ca](mailto:raph@stat.ubc.ca)>, Kenneth Lo <[c.lo@stat.ubc.ca](mailto:c.lo@stat.ubc.ca)>

### References

Lo, K., Brinkman, R. R. and Gottardo, R. (2008) Automated Gating of Flow Cytometry Data via Robust Model-based Clustering. *Cytometry A* **73**, 321-332.

### See Also

[summary](#), [plot](#), [density](#), [hist](#), [Subset](#), [split](#), [ruleOutliers](#), [Map](#), [SimulateMixture](#)

### Examples

```
data(rituximab)

### cluster the data using FSC.H and SSC.H
res1 <- flowClust(rituximab, varNames=c("FSC.H", "SSC.H"), K=1)

### remove outliers before proceeding to the second stage
# %in% operator returns a logical vector indicating whether each
# of the observations lies within the cluster boundary or not
rituximab2 <- rituximab[rituximab %in% res1,]
# a shorthand for the above line
rituximab2 <- rituximab[res1,]
# this can also be done using the Subset method
rituximab2 <- Subset(rituximab, res1)

### cluster the data using FL1.H and FL3.H (with 3 clusters)
res2 <- flowClust(rituximab2, varNames=c("FL1.H", "FL3.H"), K=3)
show(res2)
summary(res2)

# to demonstrate the use of the split method
split(rituximab2, res2)
split(rituximab2, res2, population=list(sc1=c(1,2), sc2=3))

# to show the cluster assignment of observations
table(Map(res2))

# to show the cluster centres (i.e., the mean parameter estimates
# transformed back to the original scale)
getEstimates(res2)$locations
```

```

### demonstrate the use of various plotting methods
# a scatterplot
plot(res2, data=rituximab2, level=0.8)
# a contour / image plot
res2.den <- density(res2, data=rituximab2)
plot(res2.den)
plot(res2.den, type="image", nlevels=100)
# a histogram (1-D density) plot
hist(res2, data=rituximab2, subset="FL1.H")

# the following line illustrates how to select a subset of data
# to perform cluster analysis through the min and max arguments;
# also note the use of level to specify a rule to call outliers
# other than the default
flowClust(rituximab2, varNames=c("FL1.H", "FL3.H"), K=3, B=100,
          min=c(0,0), max=c(400,800), level=0.95, z.cutoff=0.5)

```

---

```
hist,flowClust-method
```

*1-D Density Plot (Histogram) of Clustering Results*

---

## Description

This method generates a one-dimensional density plot for the specified dimension (variable) based on the robust model-based clustering results. A histogram of the actual data or cluster assignment is optional for display.

## Usage

```

## S4 method for signature 'flowClust':
hist(x, data=NULL, subset=1, include=1:(x@K), histogram=TRUE,
     labels=TRUE, xlim=NULL, ylim=NULL, xlab=NULL, ylab="Density",
     main=NULL, breaks=50, col=NULL, pch=20, cex=0.6, ...)

```

## Arguments

<code>x</code>	Object returned from <code>flowClust</code> or from running <code>filter</code> on a <code>flowFrame</code> object.
<code>data</code>	A numeric vector, matrix, data frame of observations, or object of class <code>flowFrame</code> . This is the object on which <code>flowClust</code> or <code>filter</code> was performed.
<code>subset</code>	An integer indicating which variable is selected for the plot. Alternatively, a character string containing the name of the variable is allowed if <code>x@varNames</code> is not <code>NULL</code> .
<code>include</code>	A numeric vector specifying which clusters are shown on the plot. By default, all clusters are included.
<code>histogram</code>	A logical value indicating whether a histogram of the actual data is made in addition to the density plot or not.
<code>labels</code>	A logical value indicating whether information about cluster assignment is shown or not.
<code>xlim</code>	The range of $x$ -values for the plot. If <code>NULL</code> , the data range will be used.

<code>ylim</code>	The range of $y$ -values for the plot. If <code>NULL</code> , an optimal range will be determined automatically.
<code>xlab, ylab</code>	Labels for the $x$ - and $y$ -axes respectively.
<code>main</code>	Title of the plot.
<code>breaks</code>	Content to be passed to the <code>breaks</code> argument of the generic <code>hist</code> function, if <code>histogram</code> is <code>TRUE</code> . Default is 50, meaning that 50 vertical bars with equal binwidths will be drawn.
<code>col</code>	Colors of the plotting characters displaying the cluster assignment (if <code>labels</code> is <code>TRUE</code> ). If <code>NULL</code> (default), it will be determined automatically.
<code>pch</code>	Plotting character used to show the cluster assignment.
<code>cex</code>	Size of the plotting character showing the cluster assignment.
<code>...</code>	Further arguments passed to <code>curve</code> (and also <code>hist</code> if <code>histogram</code> is <code>TRUE</code> ).

**Author(s)**

Raphael Gottardo <raph@stat.ubc.ca>, Kenneth Lo <c.lo@stat.ubc.ca>

**References**

Lo, K., Brinkman, R. R. and Gottardo, R. (2008) Automated Gating of Flow Cytometry Data via Robust Model-based Clustering. *Cytometry A* **73**, 321-332.

**See Also**

[flowClust](#), [plot](#), [density](#)

**Examples**

```
res <- flowClust(iris[,1:4], K=3)
hist(res, data=iris, subset="Petal.Length", breaks=30)
hist(res, data=iris, subset=3, histogram=FALSE, labels=FALSE)
hist(res, data=iris, subset="Petal.Length", breaks=30, include=2:3)
```

---

miscellaneous

*Various Functions for Retrieving Information from Clustering Results*

---

**Description**

Various functions are available to retrieve the information criteria (`criterion`), the posterior probabilities of clustering memberships  $z$  (`posterior`), the “weights”  $u$  (`importance`), the uncertainty (`uncertainty`), and the estimates of the cluster proportions, means and variances (`getEstimates`) resulted from the clustering (`filtering`) operation.

**Usage**

```
criterion(object, ...)
criterion(object) <- value
posterior(object, assign=FALSE)
importance(object, assign=FALSE)
uncertainty(object)
getEstimates(object, data)
```

**Arguments**

object	Object returned from <code>flowClust</code> or <code>filter</code> . For the replacement method of <code>criterion</code> , the object must be of class <code>flowClustList</code> or <code>tmixFilterResultList</code> .
...	Further arguments. Currently this is <code>type</code> , a character string. May take "BIC", "ICL" or "logLike", to specify the criterion desired.
value	A character string stating the criterion used to choose the best model. May take either "BIC" or "ICL".
assign	A logical value. If TRUE, only the quantity (z for posterior or u for importance) associated with the cluster to which an observation is assigned will be returned. Default is FALSE, meaning that the quantities associated with all the clusters will be returned.
data	A numeric vector, matrix, data frame of observations, or object of class <code>flowFrame</code> ; an optional argument. This is the object on which <code>flowClust</code> or <code>filter</code> was performed.

**Details**

These functions are written to retrieve various slots contained in the object returned from the clustering operation. `criterion` is to retrieve `object@BIC`, `object@ICL` or `object@logLike`. It replacement method modifies `object@index` and `object@criterion` to select the best model according to the desired criterion. `posterior` and `importance` provide a means to conveniently retrieve information stored in `object@z` and `object@u` respectively. `uncertainty` is to retrieve `object@uncertainty`. `getEstimates` is to retrieve information stored in `object@mu` (transformed back to the original scale) and `object@w`; when the data object is provided, an approximate variance estimate (on the original scale, obtained by performing one M-step of the EM algorithm without taking the Box-Cox transformation) will also be computed.

**Value**

Denote by  $K$  the number of clusters,  $N$  the number of observations, and  $P$  the number of variables. For `posterior` and `importance`, a matrix of size  $N \times K$  is returned if `assign=FALSE` (default). Otherwise, a vector of size  $N$  is outputted. `uncertainty` always outputs a vector of size  $N$ . `getEstimates` returns a list with named elements, `proportions`, `locations` and, if the data object is provided, `dispersion`. `proportions` is a vector of size  $P$  and contains the estimates of the  $K$  cluster proportions. `locations` is a matrix of size  $K \times P$  and contains the estimates of the  $K$  mean vectors transformed back to the original scale (i.e., `rbox(object@mu, object@lambda)`). `dispersion` is an array of dimensions  $K \times P \times P$ , containing the approximate estimates of the  $K$  covariance matrices on the original scale.

**Note**

When `object@nu=Inf`, the Mahalanobis distances instead of the "weights" are stored in `object@u`. Hence, `importance` will retrieve information corresponding to the Mahalanobis distances.

**Author(s)**

Raphael Gottardo <raph@stat.ubc.ca>, Kenneth Lo <c.lo@stat.ubc.ca>

**References**

Lo, K., Brinkman, R. R. and Gottardo, R. (2008) Automated Gating of Flow Cytometry Data via Robust Model-based Clustering. *Cytometry A* **73**, 321-332.

**See Also**

[flowClust](#), [filter](#), [Map](#)

**Examples**

```
res <- flowClust(iris[,1:4], K=3)
criterion(res)
posterior(res)
posterior(res, assign=TRUE)
importance(res)
importance(res, assign=TRUE)
uncertainty(res)
getEstimates(res)
```

---

plot,flowClust-method

*Scatterplot of Clustering Results*

---

**Description**

This method generates scatterplot revealing the cluster assignment, cluster boundaries according to the specified percentile as well as supplemental information like outliers or filtered observations.

**Usage**

```
## S4 method for signature 'flowClust':
plot(x, data, subset=c(1,2), ellipse=TRUE, show.outliers=TRUE,
     show.rm=FALSE, include=1:(x@K), main=NULL, grayscale=FALSE,
     col=(if (grayscale) gray(1/4) else 2:(length(include)+1)),
     pch=".", cex=0.6, col.outliers=gray(3/4), pch.outliers=".",
     cex.outliers=cex, col.rm=1, pch.rm=1, cex.rm=0.6, ecol=1,
     elty=1, level=NULL, u.cutoff=NULL, z.cutoff=NULL,
     npoints=501, add=FALSE, ...)
```

**Arguments**

<code>x</code>	Object returned from <a href="#">flowClust</a> .
<code>data</code>	A matrix, data frame of observations, or object of class <code>flowFrame</code> . This is the object on which <code>flowClust</code> was performed.
<code>subset</code>	A numeric vector of length two indicating which two variables are selected for the scatterplot. Alternatively, a character vector containing the names of the two variables is allowed if <code>x@varNames</code> is not <code>NULL</code> .
<code>ellipse</code>	A logical value indicating whether the cluster boundary is to be drawn or not. If <code>TRUE</code> , the boundary will be drawn according to the level specified by <code>level</code> or <code>cutoff</code> .
<code>show.outliers</code>	A logical value indicating whether outliers will be explicitly shown or not.
<code>show.rm</code>	A logical value indicating whether filtered observations will be shown or not.
<code>include</code>	A numeric vector specifying which clusters will be shown on the plot. By default, all clusters are included.



<code>main</code>	Title of the plot.
<code>grayscale</code>	A logical value specifying if a grayscale plot is desired. This argument takes effect only if the default values of relevant graphical arguments are taken.
<code>col</code>	Color(s) of the plotting characters. May specify a different color for each cluster.
<code>pch</code>	Plotting character(s) of the plotting characters. May specify a different character for each cluster.
<code>cex</code>	Size of the plotting characters. May specify a different size for each cluster.
<code>col.outliers</code>	Color of the plotting characters denoting outliers.
<code>pch.outliers</code>	Plotting character(s) used to denote outliers. May specify a different character for each cluster.
<code>cex.outliers</code>	Size of the plotting characters used to denote outliers. May specify a different size for each cluster.
<code>col.rm</code>	Color of the plotting characters denoting filtered observations.
<code>pch.rm</code>	Plotting character used to denote filtered observations.
<code>cex.rm</code>	Size of the plotting character used to denote filtered observations.
<code>ecol</code>	Color(s) of the lines representing the cluster boundaries. May specify a different color for each cluster.
<code>elty</code>	Line type(s) drawing the cluster boundaries. May specify a different line type for each cluster.
<code>level, u.cutoff, z.cutoff</code>	These three optional arguments specify the rule used to identify outliers. By default, all of them are left unspecified, meaning that the rule stated in <code>x@ruleOutliers</code> will be taken. Otherwise, these arguments will be passed to <code>ruleOutliers</code> .
<code>npoints</code>	The number of points used to draw each cluster boundary.
<code>add</code>	A logical value. If <code>TRUE</code> , add to the current plot.
<code>...</code>	Further graphical parameters passed to the generic function <code>plot</code> .

**Note**

The cluster boundaries need not be elliptical since Box-Cox transformation has been performed.

**Author(s)**

Raphael Gottardo <raph@stat.ubc.ca>, Kenneth Lo <c.lo@stat.ubc.ca>

**References**

Lo, K., Brinkman, R. R. and Gottardo, R. (2008) Automated Gating of Flow Cytometry Data via Robust Model-based Clustering. *Cytometry A* **73**, 321-332.

**See Also**

[flowClust](#)

**Examples**

```
res <- flowClust(iris[,1:4], K=3)

plot(res, data=iris, subset=c("Sepal.Width", "Petal.Width"),
      pch=1, pch.outlier=2)
plot(res, data=iris, subset=c(2,4), ellipse=FALSE,
      show.outlier=FALSE, pch=1)

plot(res, data=iris, subset=c("Petal.Length", "Petal.Width"),
      include=c(2,3), pch=1, pch.outlier=1)
plot(res, data=iris, subset=c("Petal.Length", "Petal.Width"),
      include=c(2,3), grayscale=TRUE, pch=1, pch.outlier=1,
      level=0.8)
```

---

```
plot,flowFrame,tmixFilterResult-method
```

*Scatterplot / 1-D Density Plot of Filtering (Clustering) Results*

---

**Description**

Depending on the dimensions specified, this method generates either a scatterplot or a one-dimensional density plot (histogram) based on the robust model-based clustering results.

**Usage**

```
## S4 method for signature 'flowFrame, tmixFilterResult':
plot(x, y, z=NULL, ...)
```

**Arguments**

x	Object of class <code>flowFrame</code> . This is the data object on which <code>filter</code> was performed.
y	Object of class <code>tmixFilterResult</code> or <code>tmixFilterResultList</code> returned from running <code>filter</code> .
z	A character vector of length one or two containing the name(s) of the variable(s) selected for the plot. If it is of length two, a scatterplot will be generated. If it is of length one, a 1-D density plot will be made. If it is unspecified, the first one/two variable(s) listed in <code>y@varNames</code> will be used.
...	All optional arguments passed to the <code>plot</code> or <code>hist</code> method with signature <code>'flowClust'</code> . Note that arguments <code>x</code> , <code>data</code> and <code>subset</code> have already been provided by <code>y</code> , <code>x</code> and <code>z</code> above respectively.

**Note**

This `plot` method is designed such that it resembles the argument list of the `plot` method defined in the `flowCore` package. The actual implementation is done through the `plot` or `hist` method defined for a `flowClust` object.

**Author(s)**

Raphael Gottardo <raph@stat.ubc.ca>, Kenneth Lo <c.lo@stat.ubc.ca>

**References**

Lo, K., Brinkman, R. R. and Gottardo, R. (2008) Automated Gating of Flow Cytometry Data via Robust Model-based Clustering. *Cytometry A* **73**, 321-332.

**See Also**

[filter](#), [plot](#), [hist](#)

---

plot,flowDens-method

*Contour or Image Plot of Clustering Results*

---

**Description**

This method makes use of the `flowDens` object returned by `density` to generate a contour or image plot.

**Usage**

```
## S4 method for signature 'flowDens':
plot(x, type=c("contour", "image"), nlevels=30,
     scale=c("raw", "log", "sqrt"), color=c("rainbow",
     "heat.colors", "terrain.colors", "topo.colors",
     "cm.colors", "gray"), xlab=colnames(x@dx),
     ylab=colnames(x@dy), ...)
```

**Arguments**

<code>x</code>	The <code>flowDens</code> object returned from <code>density</code> .
<code>type</code>	Either "contour" or "image" to specify the type of plot desired.
<code>nlevels</code>	An integer to specify the number of contour levels or colors shown in the plot.
<code>scale</code>	If "log", the logarithm of the density values will be used to generate the plot; similar interpretation holds for "sqrt". The use of a log or sqrt elicits more information about low density regions.
<code>color</code>	A string containing the name of the function used to generate the desired list of colors.
<code>xlab, ylab</code>	Labels for the <i>x</i> - and <i>y</i> -axes respectively.
<code>...</code>	Other arguments to be passed to <code>contour</code> or <code>image</code> , for example, <code>drawlabels</code> and <code>add</code> . Once an image plot is generated, users may impose a contour plot on it by calling this function with an additional argument <code>add=TRUE</code> .

**Author(s)**

Raphael Gottardo <[raph@stat.ubc.ca](mailto:raph@stat.ubc.ca)>, Kenneth Lo <[c.lo@stat.ubc.ca](mailto:c.lo@stat.ubc.ca)>

**References**

Lo, K., Brinkman, R. R. and Gottardo, R. (2008) Automated Gating of Flow Cytometry Data via Robust Model-based Clustering. *Cytometry A* **73**, 321-332.

**See Also**

[flowClust](#), [density](#)

---

rbox

*Reverse Box-Cox Transformation*

---

**Description**

This function performs back transformation on Box-Cox transformed data.

**Usage**

```
rbox(data, lambda)
```

**Arguments**

`data` A numeric vector, matrix or data frame of observations.  
`lambda` The Box-Cox transformation applied which results in the inputted data matrix.

**Value**

A numeric vector, matrix or data frame of the same dimension as `data` is returned.

**Note**

Please refer to the documentation for `box` for details about the Box-Cox transformation in use.

**See Also**

[box](#)

---

rituximab

*The Rituximab Dataset*

---

**Description**

A flow cytometry dataset produced in a drug-screening project to identify agents that would enhance the anti-lymphoma activity of Rituximab, a therapeutic monoclonal antibody. Cells were stained with anti-BrdU FITC and the DNA binding dye 7-AAD.

**Usage**

```
data(rituximab)
```

**Format**

An object of class `flowFrame` with 1545 cells (rows) and the following eight variables (columns):

FSC.H FSC-Height

SSC.H Side Scatter

FL1.H Anti-BrdU FITC

FL2.H Channel not used

FL3.H 7 AAD

FL1.A Channel not used

FL1.W Channel not used

Time Time

**Source**

Gasparetto, M., Gentry, T., Sebti, S., O'Bryan, E., Nimmanapalli, R., Blaskovich, M. A., Bhalla, K., Rizzieri, D., Haaland, P., Dunne, J. and Smith, C. (2004) Identification of compounds that enhance the anti-lymphoma activity of rituximab using flow cytometric high-content screening. *J. Immunol. Methods* **292**, 59-71.

---

`ruleOutliers,flowClust-method`

*Showing or Modifying the Rule used to Identify Outliers*

---

**Description**

This method shows or modifies the rule used to identify outliers.

**Usage**

```
## S4 method for signature 'flowClust':  
ruleOutliers(object)  
ruleOutliers(object) <- value
```

**Arguments**

<code>object</code>	Object returned from <code>flowClust</code> or <code>filter</code> .
<code>value</code>	A list object with one or more of the following named elements: <code>level</code> , <code>u.cutoff</code> and <code>z.cutoff</code> . Their interpretations are the same as those of the corresponding arguments in the <code>flowClust</code> function. Note that when both <code>level</code> and <code>u.cutoff</code> are missing, the rule set by the original value of <code>level</code> or <code>u.cutoff</code> will be unchanged rather than removed. Likewise, when <code>z.cutoff</code> is missing, the rule set by the original value of <code>z.cutoff</code> will be retained.

**Value**

The replacement method modifies `object@ruleOutliers` (or `object[[k]]@ruleOutliers` if `object` is of class `flowClustList` or `tmixFilterResultList`) AND updates the logical vector `object@flagOutliers` (or `object[[k]]@ruleOutliers`) according to the new rule.

**Author(s)**

Raphael Gottardo <[raph@stat.ubc.ca](mailto:raph@stat.ubc.ca)>, Kenneth Lo <[c.lo@stat.ubc.ca](mailto:c.lo@stat.ubc.ca)>

**References**

Lo, K., Brinkman, R. R. and Gottardo, R. (2008) Automated Gating of Flow Cytometry Data via Robust Model-based Clustering. *Cytometry A* **73**, 321-332.

**See Also**

[flowClust](#), [filter](#)

**Examples**

```
res <- flowClust(iris[,1:4], K=3)
summary(res)
# change the rule to call outliers
ruleOutliers(res) <- list(z.cutoff=0.9)
# more outliers now
summary(res)
ruleOutliers(res) <- list(level=0.95, z.cutoff=0)
# augmented cluster boundaries lead to fewer outliers
summary(res)
```

---

show,flowClust-method

*Show Method for flowClust / tmixFilterResult Object*

---

**Description**

This method lists out the slots contained in a `flowClust` object.

**Usage**

```
## S4 method for signature 'flowClust':
show(object)

## S4 method for signature 'tmixFilterResult':
show(object)
```

**Arguments**

`object`                      Object returned from [flowClust](#) or [filter](#).

**Author(s)**

Raphael Gottardo <raph@stat.ubc.ca>, Kenneth Lo <c.lo@stat.ubc.ca>

**See Also**

[flowClust](#), [filter](#), [summary](#)

---

show,tmixFilter-method

*Show Method for tmixFilter Object*

---

**Description**

This method shows the filtering settings stored in a `tmixFilter` object.

**Usage**

```
## S4 method for signature 'tmixFilter':  
show(object)
```

**Arguments**

`object`            Object of class `tmixFilter`.

**Author(s)**

Raphael Gottardo <raph@stat.ubc.ca>, Kenneth Lo <c.lo@stat.ubc.ca>

---

split,flowClust-method

*Splitting Data Based on Clustering Results*

---

**Description**

This method splits data according to results of the clustering (filtering) operation. Outliers identified will be removed by default.

**Value**

A list object with elements each of which is a subset of `x` and also retains the same class as `x`. If the `split` argument is specified with a list of named elements, those names will be used to name the corresponding elements in the resultant list object.

**Usage**

```
split(x, f, drop=FALSE, population=NULL, split=NULL, rm.outliers=TRUE, ...)
```

**Arguments**

- x A numeric vector, matrix, data frame of observations, or object of class `flowFrame`. This is the object on which `flowClust` or `filter` was performed.
- f Object returned from `flowClust` or `filter`.
- drop A logical value indicating whether to coerce a column matrix into a vector, if applicable. Default is `FALSE`, meaning that a single-column matrix will be retained.
- population An optional argument which specifies how to split the data. If specified, it takes a list object with named or unnamed elements each of which is a numeric vector specifying which clusters are included. If this argument is left unspecified, the data object will be split into  $K$  subsets each of which is formed by one out of the  $K$  clusters used to model the data. See examples for more details.
- split This argument is deprecated. Should use `population` instead.
- rm.outliers A logical value indicating whether outliers are removed or not.
- ... Further arguments to be passed to or from other methods.

**Author(s)**

Raphael Gottardo <raph@stat.ubc.ca>, Kenneth Lo <c.lo@stat.ubc.ca>

**References**

Lo, K., Brinkman, R. R. and Gottardo, R. (2008) Automated Gating of Flow Cytometry Data via Robust Model-based Clustering. *Cytometry A* **73**, 321-332.

**See Also**

`Subset`, `flowClust`, `filter`

---

summary,flowClust-method

*Summary Method for flowClust Object*

---

**Description**

This method prints out various characteristics of the model fitted via robust model-based clustering.

**Usage**

```
## S4 method for signature 'flowClust':
summary(object)

## S4 method for signature 'tmixFilterResult':
summary(object)
```

**Arguments**

object            Object returned from `flowClust` or from `filter`.



**Details**

Various characteristics of the fitted model will be given under the following five categories: Experiment Information, Clustering Summary, Transformation Parameter, Information Criteria, and Data Quality. Under Data Quality, information about data filtering, outliers, and uncertainty is given.

**Author(s)**

Raphael Gottardo <raph@stat.ubc.ca>, Kenneth Lo <c.lo@stat.ubc.ca>

**See Also**

`flowClust`, `filter`, `show`

---

 tmixFilter

*Creating Filters and Filtering Flow Cytometry Data*


---

**Description**

The `tmixFilter` function creates a filter object which is then passed to the `filter` method that performs filtering on a `flowFrame` object. This method pair is provided to let **flowClust** integrate with the **flowCore** package.

**Usage**

```
tmixFilter(filterId="tmixFilter", parameters="", ...)
```

**Arguments**

<code>filterId</code>	A character string that identifies the filter created.
<code>parameters</code>	A character vector specifying the variables to be used in filtering. When it is left unspecified, all the variables of the <code>flowFrame</code> object are used when running <code>filter</code> . Note that its content will be passed to the <code>varNames</code> argument of <code>flowClust</code> when running <code>filter</code> .
<code>...</code>	Other arguments passed to the <code>flowClust</code> function when running <code>filter</code> , namely, <code>expName</code> , <code>K</code> , <code>B</code> , <code>tol</code> , <code>nu</code> , <code>lambda</code> , <code>trans</code> , <code>min.count</code> , <code>max.count</code> , <code>min</code> , <code>max</code> , <code>level</code> , <code>u.cutoff</code> , <code>z.cutoff</code> , <code>randomStart</code> and <code>seed</code> . All arguments are optional except <code>K</code> that specifies the number of clusters.

**Value**

The `tmixFilter` function returns an object of class `tmixFilter` that stores all the settings required for performing the `filter` operations.

The `filter` method is defined in package `flowCore` and returns an object of class `tmixFilterResult` (or `tmixFilterResultList` if `filter@K` has a length >1) that stores the filtering results.

**Note**

The `tmixFilter` function returns an object of class `tmixFilter` that extends the virtual parent `filter` class in the **flowCore** package. Hence, the **filter operators**, namely, `&`, `|`, `!` and `%subset%`, also work for the `tmixFilter` class.

If `filter@K` is of length 1, the `filter` method returns an object of class `tmixFilterResult`. This class extends both the `multipleFilterResult` class (in the **flowCore** package) and the `flowClust` class. Operations defined for the `multipleFilterResult` class, like `%in%`, `Subset` and `split`, also work for the `tmixFilterResult` class. Likewise, methods or functions designed to retrieve filtering (clustering) information from a `flowClust` object can also be applied on a `tmixFilterResult` object. These include `criterion`, `ruleOutliers`, `ruleOutliers<-`, `Map`, `posterior`, `importance`, `uncertainty` and `getEstimates`. Various functionalities for plotting the filtering results are also available (see the links below).

If `filter@K` has a length >1, the function returns an object of class `tmixFilterResultList`. This class extends both the `flowClustList` class and the `multipleFilterResult` class. Note that when a `tmixFilterResultList` object is used in place of a `tmixFilterResult` object, in most cases the list element corresponding to the best model will be extracted and passed to the method/function call.

**Author(s)**

Raphael Gottardo <raph@stat.ubc.ca>, Kenneth Lo <c.lo@stat.ubc.ca>

**References**

Lo, K., Brinkman, R. R. and Gottardo, R. (2008) Automated Gating of Flow Cytometry Data via Robust Model-based Clustering. *Cytometry A* **73**, 321-332.

**See Also**

`flowClust`, `summary`, `plot`, `density`, `hist`, `Subset`, `split`, `ruleOutliers`, `Map`

**Examples**

```
### The example below largely resembles the one in the flowClust
### man page. The main purpose here is to demonstrate how the
### entire cluster analysis can be done in a fashion highly
### integrated into flowCore.
```

```
data(rituximab)
```

```
### create a filter object
s1filter <- tmixFilter("s1", c("FSC.H", "SSC.H"), K=1)
### cluster the data using FSC.H and SSC.H
res1 <- filter(rituximab, s1filter)
```

```
### remove outliers before proceeding to the second stage
# %in% operator returns a logical vector indicating whether each
# of the observations lies inside the gate or not
rituximab2 <- rituximab[rituximab %in% res1,]
# a shorthand for the above line
rituximab2 <- rituximab[res1,]
# this can also be done using the Subset method
rituximab2 <- Subset(rituximab, res1)
```

```
### cluster the data using FL1.H and FL3.H (with 3 clusters)
s2filter <- tmixFilter("s2", c("FL1.H", "FL3.H"), K=3)
res2 <- filter(rituximab2, s2filter)

show(s2filter)
show(res2)
summary(res2)

# to demonstrate the use of the split method
split(rituximab2, res2)
split(rituximab2, res2, population=list(sc1=c(1,2), sc2=3))

# to show the cluster assignment of observations
table(Map(res2))

# to show the cluster centres (i.e., the mean parameter estimates
# transformed back to the original scale) and proportions
getEstimates(res2)

### demonstrate the use of various plotting methods
# a scatterplot
plot(rituximab2, res2, level=0.8)
# a contour / image plot
res2.den <- density(res2, data=rituximab2)
plot(res2.den)
plot(res2.den, type="image", nlevels=100)
# a histogram (1-D density) plot
plot(rituximab2, res2, "FL1.H")

# the following line illustrates how to select a subset of data
# to perform cluster analysis through the min and max arguments;
# also note the use of level to specify a rule to call outliers
# other than the default
s2t <- tmixFilter("s2t", c("FL1.H", "FL3.H"), K=3, B=100,
  min=c(0,0), max=c(400,800), level=0.95, z.cutoff=0.5)
filter(rituximab2, s2t)
```

# Index

- \*Topic **cluster**
  - flowClust, 9
  - Map, flowClust-method, 1
  - miscellaneous, 14
  - tmixFilter, 25
- \*Topic **datagen**
  - SimulateMixture, 2
- \*Topic **datasets**
  - rituximab, 20
- \*Topic **distribution**
  - dmvt, 6
- \*Topic **graphs**
  - density, flowClust-method, 5
  - hist, flowClust-method, 13
  - plot, flowClust-method, 16
  - plot, flowDens-method, 19
  - plot, flowFrame, tmixFilterResult-method, 18
- \*Topic **manip**
  - ruleOutliers, flowClust-method, 21
  - split, flowClust-method, 23
  - Subset, flowClust-method, 3
- \*Topic **math**
  - box, 4
  - rbox, 20
- \*Topic **models**
  - flowClust, 9
  - tmixFilter, 25
- \*Topic **package**
  - flowClust-package, 7
- \*Topic **print**
  - show, flowClust-method, 22
  - show, tmixFilter-method, 23
  - summary, flowClust-method, 24
- [, flowFrame, flowClust-method  
(flowClust), 9
- [, flowFrame, flowClustList-method  
(flowClust), 9
- [, flowFrame, tmixFilterResult-method  
(tmixFilter), 25
- [, flowFrame, tmixFilterResultList-method  
(tmixFilter), 25
- [, tmixFilterResultList, ANY-method  
(tmixFilter), 25
- %in%, 26
- %in%, ANY, flowClust-method  
(flowClust), 9
- %in%, ANY, flowClustList-method  
(flowClust), 9
- %in%, ANY, tmixFilterResult-method  
(tmixFilter), 25
- %in%, ANY, tmixFilterResultList-method  
(tmixFilter), 25
- %in%, flowFrame, tmixFilter-method  
(tmixFilter), 25
- %in%, flowFrame, tmixFilterResult-method  
(tmixFilter), 25
- box, 4, 7, 20
- criterion, 10, 26
- criterion (miscellaneous), 14
- criterion, flowClust-method  
(miscellaneous), 14
- criterion, flowClustList-method  
(miscellaneous), 14
- criterion<- (miscellaneous), 14
- criterion<-, flowClustList, character-method  
(miscellaneous), 14
- density, 10, 12, 14, 19, 20, 26
- density, flowClust-method, 5, 7
- density, flowClustList-method  
(density, flowClust-method), 5
- density-method  
(density, flowClust-method), 5
- density.flowClust  
(density, flowClust-method), 5
- dmvt, 6, 7
- filter, 1-5, 10, 13, 15, 16, 18, 19, 21-26
- filter (tmixFilter), 25
- filter operators, 26

- filter, flowFrame, filter-method  
(*tmixFilter*), 25
- filter, flowFrame, filterSet-method  
(*tmixFilter*), 25
- filter, flowFrame, tmixFilter-method  
(*tmixFilter*), 25
- filter, flowSet, filter-method  
(*tmixFilter*), 25
- filter, flowSet, filterSet-method  
(*tmixFilter*), 25
- filter, flowSet, list-method  
(*tmixFilter*), 25
- filter-method(*tmixFilter*), 25
- filter.flowFrame(*tmixFilter*), 25
- flowClust, 1–6, 8, 9, 10, 13–17, 20–26
- flowClust-class(*flowClust*), 9
- flowClust-package, 7
- flowClustList, 15, 26
- flowClustList(*flowClust*), 9
- flowClustList-class(*flowClust*), 9
- flowDens-class  
(*density*, *flowClust*-method),  
5
  
- getEstimates, 10, 26
- getEstimates(*miscellaneous*), 14
  
- hist, 10, 12, 18, 19, 26
- hist, flowClust-method, 8, 13
- hist, flowClustList-method  
(*hist*, *flowClust*-method), 13
- hist.flowClust  
(*hist*, *flowClust*-method), 13
  
- importance, 10, 26
- importance(*miscellaneous*), 14
  
- Map, 10, 12, 16, 26
- Map(*Map*, *flowClust*-method), 1
- Map, flowClust-method, 1, 8
- Map, flowClustList-method  
(*Map*, *flowClust*-method), 1
- Map.flowClust  
(*Map*, *flowClust*-method), 1
- miscellaneous, 8, 14
- multipleFilterResult, 26
  
- plot, 6, 10, 12, 14, 18, 19, 26
- plot, flowClust, missing-method  
(*plot*, *flowClust*-method), 16
- plot, flowClust-method, 8, 16
- plot, flowClustList, missing-method  
(*plot*, *flowClust*-method), 16
- plot, flowDens, missing-method  
(*plot*, *flowDens*-method), 19
- plot, flowDens-method, 8, 19
- plot, flowFrame, tmixFilterResult-method,  
8, 18
- plot, flowFrame, tmixFilterResultList-method  
(*plot*, *flowFrame*, *tmixFilterResult*-method),  
18
- plot, tmixFilterResult-method  
(*plot*, *flowFrame*, *tmixFilterResult*-method),  
18
- plot.flowClust  
(*plot*, *flowClust*-method), 16
- plot.flowDens  
(*plot*, *flowDens*-method), 19
- plot.flowFrame  
(*plot*, *flowFrame*, *tmixFilterResult*-method),  
18
- plot.tmixFilterResult  
(*plot*, *flowFrame*, *tmixFilterResult*-method),  
18
- posterior, 2, 10, 26
- posterior(*miscellaneous*), 14
  
- rbox, 5, 8, 20
- rituximab, 20
- ruleOutliers, 10, 12, 17, 26
- ruleOutliers  
(*ruleOutliers*, *flowClust*-method),  
21
- ruleOutliers, flowClust-method, 8,  
21
- ruleOutliers, flowClustList-method  
(*ruleOutliers*, *flowClust*-method),  
21
- ruleOutliers.flowClust  
(*ruleOutliers*, *flowClust*-method),  
21
- ruleOutliers<-, 26
- ruleOutliers<-  
(*ruleOutliers*, *flowClust*-method),  
21
- ruleOutliers<-, flowClust, list-method  
(*ruleOutliers*, *flowClust*-method),  
21
- ruleOutliers<-, flowClustList, list-method  
(*ruleOutliers*, *flowClust*-method),  
21
  
- show, 25
- show, flowClust-method, 8, 22
- show, flowClustList-method  
(*show*, *flowClust*-method), 22

- show, tmixFilter-method, 8, 23
- show, tmixFilterResult-method  
(*show, flowClust-method*), 22
- show, tmixFilterResultList-method  
(*show, flowClust-method*), 22
- show.flowClust  
(*show, flowClust-method*), 22
- show.tmixFilter  
(*show, tmixFilter-method*),  
23
- show.tmixFilterResult  
(*show, flowClust-method*), 22
- SimulateMixture, 2, 8, 12
- split, 4, 10, 12, 26
- split (*split, flowClust-method*), 23
- split, data.frame, flowClust-method  
(*split, flowClust-method*),  
23
- split, data.frame, flowClustList-method  
(*split, flowClust-method*),  
23
- split, flowClust-method, 8, 23
- split, flowFrame, flowClust-method  
(*split, flowClust-method*),  
23
- split, flowFrame, flowClustList-method  
(*split, flowClust-method*),  
23
- split, flowFrame, tmixFilterResult-method  
(*split, flowClust-method*),  
23
- split, flowFrame, tmixFilterResultList-method  
(*split, flowClust-method*),  
23
- split, matrix, flowClust-method  
(*split, flowClust-method*),  
23
- split, matrix, flowClustList-method  
(*split, flowClust-method*),  
23
- split, vector, flowClust-method  
(*split, flowClust-method*),  
23
- split, vector, flowClustList-method  
(*split, flowClust-method*),  
23
- split.flowClust  
(*split, flowClust-method*),  
23
- split.flowFrame  
(*split, flowClust-method*),  
23
- split.tmixFilterResult  
(*split, flowClust-method*),  
23
- Subset, 10, 12, 24, 26
- Subset (*Subset, flowClust-method*),  
3
- Subset, ANY, flowClustList-method  
(*Subset, flowClust-method*),  
3
- Subset, data.frame, flowClust-method  
(*Subset, flowClust-method*),  
3
- Subset, flowClust-method, 3, 8
- Subset, flowFrame, flowClust-method  
(*Subset, flowClust-method*),  
3
- Subset, flowFrame, tmixFilterResult-method  
(*Subset, flowClust-method*),  
3
- Subset, flowFrame, tmixFilterResultList-method  
(*Subset, flowClust-method*),  
3
- Subset, matrix, flowClust-method  
(*Subset, flowClust-method*),  
3
- Subset, vector, flowClust-method  
(*Subset, flowClust-method*),  
3
- Subset.flowClust  
(*Subset, flowClust-method*),  
3
- Subset.flowFrame  
(*Subset, flowClust-method*),  
3
- Subset.tmixFilterResult  
(*Subset, flowClust-method*),  
3
- summary, 12, 23, 26
- summary, flowClust-method, 8, 24
- summary, flowClustList-method  
(*summary, flowClust-method*),  
24
- summary, tmixFilterResult-method  
(*summary, flowClust-method*),  
24
- summary, tmixFilterResultList-method  
(*summary, flowClust-method*),  
24
- summary.flowClust  
(*summary, flowClust-method*),  
24
- summary.tmixFilterResult

(*summary, flowClust-method*),  
24

*tmixFilter*, 8, 10, 23, 25  
*tmixFilter-class (tmixFilter)*, 25  
*tmixFilterResult-class*  
    (*tmixFilter*), 25  
*tmixFilterResultList*, 15  
*tmixFilterResultList*  
    (*tmixFilter*), 25  
*tmixFilterResultList-class*  
    (*tmixFilter*), 25

uncertainty, 10, 26  
uncertainty (*miscellaneous*), 14