Package ‘immunoClust’

February 2, 2024

Type Package

Title immunoClust - Automated Pipeline for Population Detection in Flow Cytometry

Version 1.34.0

Date 2023-09-19

Maintainer Till Soerensen <till.soerensen@bioretis.com>

Depends R(>= 3.6), flowCore

Imports methods, stats, graphics, grid, lattice, grDevices

Suggests BiocStyle, utils, testthat

Description immunoClust is a model based clustering approach for Flow Cytometry samples. The cell-events of single Flow Cytometry samples are modelled by a mixture of multinominal normal- or t-distributions. The cell-event clusters of several samples are modelled by a mixture of multinominal normal-distributions aiming stable co-clusters across these samples.


biocViews Clustering, FlowCytometry, SingleCell, CellBasedAssays, ImmunoOncology

License Artistic-2.0

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

git_branch RELEASE_3_18

git_last_commit 6c8b7f4

git_last_commit_date 2023-10-24

Repository Bioconductor 3.18

Date/Publication 2024-02-02

Author Till Soerensen [aut, cre]
R topics documented:

immunoClust-package ...................................................... 2
bhattacharyya ............................................................. 3
cell.ClustData ............................................................. 4
cell.EM ....................................................................... 6
cell.FitModel ............................................................ 8
cell.hclust ............................................................... 9
cell.ME ................................................................. 10
cell.process ............................................................ 12
cell.removed ........................................................... 16
cell.SubClustering ..................................................... 17
dat.exp ................................................................. 19
dat.fcs ................................................................. 20
dat.meta .............................................................. 21
generics.immunoclust .................................................. 21
immunoClust-object .................................................... 24
immunoMeta-class ....................................................... 25
meta.clustering .......................................................... 26
meta.export ............................................................ 28
meta.expres .......................................................... 30
meta.hclust ............................................................ 31
meta.ME ............................................................... 32
meta.normalize ......................................................... 34
meta.process .......................................................... 35
meta.regnorm .......................................................... 37
meta.SON.clustering ................................................... 38
meta.SON.combineClustering ......................................... 39
meta.SubClustering ................................................... 41
methods.immunoClust ................................................. 43
methods.immunoMeta .................................................. 45
plot.immunoClust ....................................................... 49
plot.immunoMeta ....................................................... 51
splom.immunoClust ..................................................... 52
trans.ApplyToData ..................................................... 53
trans.FitToData ........................................................ 55

Index 57

immunoClust-package  immunoClust - Automated Pipeline for Population Detection in Flow Cytometry
bhattacharyya

Description

Model based clustering and meta-clustering routines for Flow Cytometry (FC) data. The immunoClust-pipeline consists of two major procedures:

- **cell.process**: Clustering of cell-events
- **meta.process**: Meta-clustering of cell-clusters

Cell-events clustering is performed for each FC data sample separately. After this all cell-clustering results are collected in a vector and meta-clustering is performed to obtain the across samples populations.

Details

- **Package**: immunoClust
- **Type**: Package
- **Version**: 1.0.0
- **Depends**: R(>= 2.13.0), methods, stats, graphics, grid, lattice, flowCore
- **Date**: 2015-01-28
- **License**: Artistic-2.0

Author(s)

Till Sörensen <till-antoni.soerensen@charited.de>

References


bhattacharyya

**Bhattacharyya Distance, Coefficient and Probability**

Description

Calculates the Bhattacharyya Distance, Coefficient and Probability

Usage

- `bhattacharyya.prob(gM, gS, cM, cS, alpha=1)`
- `bhattacharyya.dist(gM, gS, cM, cS)`
- `bhattacharyya.coeff(gM, gS, cM, cS, alpha=1)`
Arguments

- \(g_M, c_M\) P-dimensional vector of cluster means
- \(g_S, c_S\) PxP-dimensinal matrix of clusters co-variances
- \(\alpha\) A value between 0 and 1 used to balance the bhattacharrya probabilities, co-efficients calculated with either the full covariance matrices or using only the diagonal elements of it.

Details

Calculates the bhattacharyya probaility, distance or coefficient of the clusters, i.e. Gaussian distributions. Distance and Coefficient are symetric for both clusters, whereas the probability is not.

Value

The Bhattacharyya probability, distance or coefficient

Author(s)

Till Sörensen <till-antoni.soerensen@charite.de>

Examples

data(dat.meta)

prob <- bhattacharyya.prob(prop(dat.meta,"M",c()), prop(dat.meta,"S"),
    mu(dat.meta,1), sigma(dat.meta,1))
dist <- bhattacharyya.dist(prop(dat.meta,"M",c()), prop(dat.meta,"S"),
    mu(dat.meta,1), sigma(dat.meta,1))
coeff <- bhattacharyya.coeff(prop(dat.meta,"M",c()), prop(dat.meta,"S"),
    mu(dat.meta,1), sigma(dat.meta,1))

Description

Performs EM-iteration on cell events, where an initial event cluster membership is obtained by hierarchical clustering on a sample subset given a number of clusters.

Usage

cell.ClustData(data, K, parameters=NULL, expName="immunoClust Experiment",
    sample.number=1500, sample.standardize=TRUE,
    B=50, tol=1e-5, modelName="mvt")
Arguments

data  A numeric matrix, data frame of observations, or object of class flowFrame. Rows correspond to observations and columns correspond to measured parameters.

K  Given number of clusters for the final model.

parameters  A character vector specifying the parameters (columns) to be included in clustering. When it is left unspecified, all the parameters will be used.

expName  The name of the clustering experiment.

sample.number  The maximum number of samples used for initial hierarchical clustering.

sample.standardize  A numeric indicating whether the samples for hierarchical clustering are standardized (mean=0, SD=1).

B  The maximum number of EM-iterations.

tol  The tolerance used to assess the convergence of the EM-algorithm.

modelName  Used mixture model; either "mvt" for a t-mixture model or "mvn" for a Gaussian Mixture model.

Details

Although this function provides the possibility to cluster an arbitrary set of observed data into a fixed number of clusters, this function is used in the immunoClust-pipeline only for the calculation of the initial model with one cluster.

Value

The fitted model cluster information in an object of class immunoClust.

Author(s)

Till Sörensen <till-antoni.soerensen@charite.de>

References


See Also

immunoClust-object, cell.hclust

Examples

data(dat.fcs)
res <- cell.ClustData(dat.fcs, parameters=c("FSC-A", "SSC-A"), 5)
summary(res)
cell.EM  

**immunoClust EM-iteration on Cell-events given initial Model Parameters**

**Description**

Performs EM-iteration on cell event observations giving initial model parameters and returns the fitted clusters information in an object of class `immunoClust`.

**Usage**

```r
cell.EM(data, parameters=NULL, expName="immunoClust Experiment", history=NULL, state=NULL, K, w, m, s, B=50, tol=1e-5, bias=0.5, modelName="mvt")
cell.EMt(data, K, w, m, s, parameters=NULL, expName="immunoClust Experiment", B=50, tol=1e-5, bias=0.5, modelName="mvt")
cell.EMstep(data, K, w, m, s, parameters=NULL, expName="immunoClust EMstep", B=1, tol=1e-5, modelName="mvt")
cell.Estimation(data, parameters=NULL, expName="immunoClust Experiment", history=NULL, state=NULL, K, w, m, s, scale_Z=TRUE, modelName="mvt")
cell.Estep(data, K, w, m, s, parameters=NULL, expName="immunoClust Estep", scale_Z=TRUE, modelName="mvt")
```

**Arguments**

- `data`: A numeric matrix, data frame of observations, or object of class `flowFrame`.
- `parameters`: A character vector specifying the parameters (columns) to be included in clustering. When it is left unspecified, all the parameters will be used.
- `expName`: The name of the clustering experiment.
- `history`: Experimental; unused so far.
- `state`: Experimental; unused so far.
- `K`: The number of clusters.
- `w`: The $K$-dimensional vector of the mixture proportions.
- `m`: The $K \times P$-dimensional matrix of the $K$ estimated cluster means.
- `s`: The $K \times P \times P$-dimensional matrix of the $K$ estimated cluster covariance matrices.
cell.EM

8  The maximum number of EMt-iterations.
tol The tolerance used to assess the convergence of the EMt-algorithms.
bias The ICL-bias used in the EMt-algorithm.
scale_Z Scale the returned a-posteriori probabilities to one for each observed event.
modelName Used mixture model; either "mvt" or "mvn" for a t- or Gaussian mixture model respectively.

Details

 Whereas cell.EM performs a complete EMt-iteration, cell.Estimate only calculates the a-posteriori probabilities and the Maximum-A-Posteriori estimators of cluster membership for all events.

 cell.EM is misspelling since it dose an EMt-iteration and becomes deprecated in future, so better use cell.EMt. For an EM-iteration use cell.EMstep.

 cell.Estep and cell.Estimation do the same call. In cell.Estep the calling options are a bit better structured and cell.Estimation becomes deprecated in future.

Value

 The fitted clusters information in an object of class immunoClust.

Author(s)

 Till Sörensen <till-antoni.soerensen@charite.de>

References


See Also

cell.ME, cell.FitModel

Examples

data(dat.fcs)
data(dat.exp)
## cell.clustering result for dat.fcs
r <- dat.exp[[1]]
summary(r)
## apply model parameter to all (unfiltered) events
dat.trans <- trans.ApplyToData(r, dat.fcs)
r2 <- cell.EM(dat.trans, parameters(r), K=ncls(r),
    w=weights(r), m=mu(r), s=sigma(r))
summary(r2)
Description

The function fits initial model parameters to specific observed cell event data. The function returns the cluster information of the fitted model in an object of class `immunoClust`.

Usage

```r
cell.FitModel(x, data, B=50, tol=1e-5, bias=0.5, modelName="mvt")
cell.Classify(x, data, modelName="mvt")
```

Arguments

- `x`: An immunoClust object with the initial model parameter (`parameters`, `K`, `w`, `mu`, `sigma`).
- `data`: A numeric matrix, data frame of observations, or object of class flowFrame.
- `B`: The maximum number of EMt-iterations.
- `tol`: The tolerance used to assess the convergence of the EMt-algorithms.
- `bias`: The ICL-bias used in the EMt-algorithm.
- `modelName`: Used mixture model; either "mvt" or "mvn" for a t- or Gaussian mixture model respectively.

Details

These functions are wrappers of the functions `cell.EM` and `cell.Estimation`, when model cluster parameters are combined in an object of class `immunoClust` and are used in the iterative cell event clustering process `cell.process` of `immunoClust` and are not intended to be called directly.

Value

The fitted model cluster information in an object of class `immunoClust`.

Author(s)

Till Sörensen <till-antoni.soerensen@charite.de>

References

cell.hclust

See Also

cell.process, cell.EM, cell.Estimation

Examples

data(dat.fcs)
data(dat.exp)
r1 <- dat.exp[[1]]
dat.trans <- trans.ApplyToData(r1, dat.fcs)
r2 <- cell.FitModel(r1, dat.trans)

Description

Performs model based agglomerative clustering on cell event observations with weights. It is used in the interactive cell event clustering approach of immunoClust to obtain an initial cluster membership for the EM(t)-iteration.

Usage

    cell.hclust(data, weights=NULL)

Arguments

data          The numeric $N \times P$-dimensional data matrix to cluster. Each row contains a $P$-dimensional overservation vector.
weights       The $N$-dimensional vector of optional weights to be applied for the overservations.

Details

This function is used internally in cell.TestSubCluster procedure of immunoClust.

Value

A numeric $(N - 1) \times 2$-dimensional matrix which gives the minimum index for observations in each of the two clusters merged at the $i$th step in each row.

Author(s)

Till Sörensen <till-antoni.soerensen@charite.de>
References


See Also

cell.TestSubCluster, cell.process

Examples

data(dat.fcs)
inc <- sample(1:nrow(dat.fcs), 50)
result <- cell.hclust(exprs(dat.fcs)[inc,])

---

cell.ME

**immunoClust EM-iteration on Cell-events given initial Cluster Membership Assignment**

Description

Performs an EM-iteration on cell event observations given an initial cluster membership for the cell events and returns the fitted cluster information in an object of class immunoClust.

Usage

cell.ME(data, parameters=NULL, expName="immunoClust Experiment", history=NULL, state=NULL, label, B=50, tol=1e-5, modelName="mvt")
cell.MEstep(data, label, parameters=NULL, expName="immunoClust Experiment", B=50, tol=1e-5, modelName="mvt")
cell.Mstep(data, label, parameters=NULL, expName="immunoClust Mstep", modelName="mvt")

Arguments

data A numeric matrix, data frame of observations, or object of class flowFrame.
parameters A character vector specifying the parameters (columns) to be included in clustering. When it is left unspecified, all the parameters will be used.
expName The name of the clustering experiment.
history experimental; unused so far.
state experimental; unused so far.
The N-dimensional vector containing the initial cluster membership. A label-number of 0 for an event indicates that this event is not initially assigned to a cluster.

B The maximum number of EMt-iterations.

tol The tolerance used to assess the convergence of the EMt-algorithms.

modelName Used mixture model; either "mvt" or "mvn" for a t- or Gaussian mixture model respectively.

Details

cell.ME and cell.MEstep do the same call. In cell.MEstep the calling options are a bit better structured and cell.ME becomes deprecated in future.

Value

The fitted clusters information in an object of class immunoClust.

Author(s)

Till Sörensen <till-antoni.soerensen@charite.de>

References


See Also

    cell.EM

Examples

data(dat.fcs)
data(dat.exp)
## cell.clustering result for dat.fcs
r1 <- dat.exp[[1]]
summary(r1)
## apply model parameter to all (unfiltered) events
dat.trans <- trans.ApplyToData(r1, dat.fcs)
r2 <- cell.ME(dat.trans, parameters(r1), label=label(r1))
summary(r2)
cell.process

Clustering of Cell-events in the immunoClust-pipeline

Description

This function performs iterative model based clustering on cell-event data. It takes the observed cell-event data as major input and returns an object of class immunoClust, which contains the fitted mixture model parameter and cluster membership information. The additional arguments control the routines for data preprocessing, major loop and EMt-iteration, the model refinement routine and transformation estimation.

Usage

```
cell.process(fcs, parameters=NULL, 
apply.compensation=FALSE, classify.all=FALSE, 
N=NULL, min.count=10, max.count=10, min=NULL, max=NULL, 
I.buildup=6, I.final=4, I.trans=I.buildup, 
modelName="mvt", tol=1e-5, bias=0.3, 
sub.tol=1e-4, sub.bias=bias, sub.thres=bias, sub.samples=1500, 
sub.extract=0.8, sub.weights=1, sub.standardize=TRUE, 
trans.estimate=TRUE, trans.minclust=10, 
trans.a=0.01, trans.b=0.0, trans.parameters=NULL)
```

```
cell.MajorIterationLoop(dat, x=NULL, parameters=NULL, 
I.buildup=6, I.final=4, 
modelName="mvt", tol=1e-5, bias=0.3, 
sub.bias=bias, sub.thres=0.0, sub.tol=1e-4, sub.samples=1500, 
sub.extract=0.8, sub.weights=1, sub.standardize=TRUE)
```

```
cell.MajorIterationTrans(fcs, x=NULL, parameters=NULL, 
I.buildup=6, I.final=4, I.trans=I.buildup, 
modelName="mvt", tol=1e-5, bias=0.3, 
sub.bias=bias, sub.thres=0.0, sub.tol=1e-4, sub.samples=1500, 
sub.extract=0.8, sub.weights=1, sub.standardize=TRUE, 
trans.minclust=5, trans.a=0.01, trans.decade=-1, trans.scale=1.0, 
trans.proc="vsHtransAw")
```

```
cell.InitialModel(dat, parameters=NULL, trans.a = 0.01, trans.b = 0.0, 
trans.decade=-1, trans.scale=1.0)
```

```
cell.classifyAll(fcs, x, apply.compensation=FALSE)
```

Arguments

**fcs** An object of class flowFrame. Rows correspond to observations and columns correspond to measured parameters.
A numeric matrix, data frame of observations, or object of class flowFrame. Rows correspond to observations and columns correspond to measured parameters.

An object of class immunoClust. Used as initial model in the major iteration loop. When left unspecified the simplest model containing 1 cluster is used as initial model.

**Arguments for data pre and post processing:**

- **parameters**: A character vector specifying the parameters (columns) to be included in clustering. When it is left unspecified, all the parameters will be used.
- **apply.compensation**: A numeric indicator whether the compensation matrix in the flowFrame should be applied.
- **classify.all**: A numeric indicator whether the removed over- and underexposed observations should also be classified after the clustering process.
- **N**: Maximum number of observations used for clustering. When unspecified or higher than the number of observations (i.e., rows) in dat, all observations are used for clustering, otherwise only the first N observations.
- **min.count**: 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 min, they will be excluded from the analysis. If min is NULL, then the minimum value of each parameter will be used. To suppress filtering, it is set to -1.
- **max.count**: An integer specifying the threshold count for filtering data points from above. Interpretation is similar to that of min.count.
- **min**: The lower limit set for data filtering. Note that it is a vector of length equal to the number of parameters (columns), implying that a different value can be set for each parameter.
- **max**: The upper limit set for data filtering. Interpretation is similar to that of min.

**Arguments for the major loop and EMt-iteration:**

- **I.buildup**: The number of major iterations, where the number of used observations is doubled successively.
- **I.final**: The number of major iterations with all observations.
- **I.trans**: The number of iterations where transformation estimation is applied.
- **modelName**: Used mixture model; either "mvt" for a t-mixture model or "mvn" for a Gaussian Mixture model. With "mvt2" an implementation variant for "mvt" is given, which is more reliable for samples with cutted values at the lower or upper edges of the parameter space (e.g., for CyTOF all values below a detection limit are set to zero which leads to wrong co-variance estimators and poor clustering results).
- **tol**: The tolerance used to assess the convergence of the major EM(t)-algorithms of all observations.
- **bias**: The ICL-bias used in the major EMt-algorithms of all observations.
Arguments for model refinement (sub-clustering):

sub.tol  The tolerance used to assess the convergence of the EM-algorithms in the sub-clustering.
sub.bias  The ICL-bias used in the sub-clustering EMT-algorithms, in general the same as the ICL-bias.
sub.thres  Defines the threshold, below which an ICL-increase is meaningless. The threshold is given as the multiple (or fraction) of the costs for a single cluster.
sub.samples  The number of samples used for initial hierarchical clustering.
sub.extract  The threshold used for cluster data extraction.
sub.weights  Power of weights applied to hierarchical clustering, where the used weights are the probabilities of cluster membership.
sub.EM  Used EM-algorithm; either "MET" for EMT-iteration or "ME" for EM-iteration without test step.
sub.standardize  A numeric indicating whether the samples for hierarchical clustering are standardized (mean=0, SD=1).

Arguments for transformation optimization:

trans.estimate  A numeric indicator whether transformation estimation should be applied.
trans.minclust  The minimum number of clusters required to start transformation estimation.
trans.a  A numeric vector, giving the (initial) scaling $a$ for the asinh-transformation $h(y) = \text{asin}(a \cdot y + b)$. A scaling factor of $a = 0$ indicates that a parameter is not transformed.
trans.b  A numeric vector, giving the (initial) translation $b$ for the asinh-transformation.
trans.parameters  A character vector, specifying the parameters (columns) to be applied for transformation. When it is left unspecified, the parameters to be transformed are obtained by the $PxDISPLAY$ information of the flowFrame description parameters. All parameters with LOG display values are transformed.
trans.decade  A numeric scale value for the theoretical maximum of transformed observation value. If below 0, no scaling of the transformed values is applied, which is the default in the immunoClust-pipeline.
trans.scale  A numeric scaling factor for the linear (i.e. not transformed) parameters. By default the linear parameters (normally the scatter parameters) are not scaled.
trans.proc  An experimental switch for alternative procedures; should be "vsHtransAw".

Details

The cell.process function does data preprocessing and calls the major iteration loop either with or without integrated transformation optimization. When transformation optimization is applied the transformation parameters give the initial transformation otherwise they define the fixed transformation.

The major iteration loop with included transformation optimization relies on flowFrames structure from the flowCore-package for the storage of the observed data.
The `cell.InitialModel` builds up an initial `immunoClust`-object with one cluster and the given transformation parameters.

The `cell.classifyAll` calculates the cluster membership for the removed cell events. The assignment of the cluster membership is critical for over- and underexposed observations and the interpretation is problematic.

**Value**

The fitted model information in an object of class `immunoClust`.

**Note**

a) The data preprocessing arguments (`min.count`, `max.count`, `min` and `max`) for removing over- and underexposed observations are adopted from `flowCust`-package with the same meaning.

b) The `sub.thres` value is given in here in relation to the single cluster costs $\frac{1}{2} \cdot P \cdot (P+1) \cdot \log(N)$. An absolute increase of the log-likelihood above is reported as reasonable from the literature. From our experience a higher value is required for this increase in FC data. For the ICL-bias and the `sub.thres` identical values were chosen. For the CyTOF dataset this value had been adjusted to 0.05 since the absolute increase of the log-likelihood became too high due to the high number of parameters.

c) The `sub.extract` value controls the smooth data extraction for a cluster. A higher value includes more events for a cluster in the sub-clustering routine.

d) The default value of `trans.a=0.01` for the initial transformation is optimized for Fluorescence Cytometry. For CyTOF data the initial scaling value was `trans.a=1.0`.

**Author(s)**

Till Sörensen <till-antoni.soerensen@charite.de>

**References**


**See Also**

`immunoClust-object`, `plot`, `splom`, `cell.FitModel`, `cell.SubClustering`, `trans.FitToData`

**Examples**

data(dat.fcs)
res <- cell.process(dat.fcs)
summary(res)
Brief Information of removed Cell-events by immunoClust Cell-event Clustering

Description

Gives information about the amount of overexposed cell-event observation in a FCS-file.

Usage

```r
removed.above(fcs, parameters=NULL, N=NULL, max.count=10, max=NULL)
removed.below(fcs, parameters=NULL, N=NULL, min.count=10, min=NULL)
```

Arguments

- `fcs`: An object of class `flowFrame`. Rows correspond to observations and columns correspond to measured parameters.
- `parameters`: A character vector specifying the parameters (columns) to be included in clustering. When it is left unspecified, all the parameters will be used.
- `N`: Maximum number of observations used for clustering. When unspecified or higher than the number of observations (i.e. rows) in dat, all observations are used for clustering, otherwise only the first `N` observations.
- `max.count`: An integer specifying the threshold count for filtering data points from above. The default is 10, meaning that if 10 or more data points are larger than or equal to `max`, they will be excluded from the analysis. If `max` is `NULL`, then the maximum value of each parameter will be used. To suppress filtering, it is set to `-1`.
- `max`: The upper limit set for data filtering. Note that it is a vector of length equal to the number of parameters (columns), implying that a different value can be set for each parameter.
- `min.count`: analogous to `max.count`.
- `min`: analogous to `min`.

Value

A table with two rows containing the number of events above `max` in each parameter and above in only this parameter. The two last columns give the sum and percentage of all events above `max` in any parameter.

Author(s)

Till Sörensen <till-antoni.soerensen@charite.de>

Examples

```r
data(dat.fcs)
removed.above(dat.fcs)
```
**cell.SubClustering**

**immunoClust Model Refinement Step in iterative Cell-events Clustering**

**Description**

These function tests each cell-cluster of a model for refining it into more sub-clusters and returns the refined model parameter in an object of class `immunoClust`.

**Usage**

```r
cell.SubClustering( x, dat, B=50, tol=1e-5, thres=0.1, bias=0.5, 
sample.weights=1, sample.EM="MEt", 
sample.number=1500, sample.standardize=TRUE, 
extract.thres=0.8, modelName="mvt")
```

```r
cell.TestSubCluster(x, y, t, cluster, J=8, B=500, tol=1e-5, bias=0.5, 
sample.EM="MEt", sample.df=5, sample.number=1500, 
sample.standardize=TRUE, modelName="mvt")
```

**Arguments**

- **x**: An immunoClust object with the initial model parameter \((K, w, mu, sigma)\).
- **dat**: A numeric matrix, data frame of observations, or object of class `flowFrame`.
- **B**: The maximum number of EM(t)-iterations in Sub-Clustering.
- **tol**: The tolerance used to assess the convergence of the EM(t)-algorithms in Sub-Clustering.
- **thres**: Defines the threshold, below which an ICL-increase is meaningless. The threshold is given as the multiple (or fraction) of the costs for a single cluster.
- **bias**: The ICL-bias used in the EMt-algorithm.
- **sample.weights**: Power of weights applied to hierarchical clustering, where the used weights are the probabilities of cluster membership.
- **sample.EM**: Used EM-algorithm; either "MEt" for EMt-iteration or "ME" for EM-iteration without test step.
- **sample.number**: The number of samples used for initial hierarchical clustering.
- **sample.standardize**: A numeric indicating whether the samples for hierarchical clustering are standardized (mean=0, SD=1).
- **extract.thres**: The threshold used for cluster data extraction.
- **modelName**: Used mixture model; either `mvt` for a \(t\)-mixture model or `mvn` for a Gaussian Mixture model.
- **y**: A numeric matrix of the observations belonging to the particular cluster.
cell.SubClustering

t A numeric vector with the probability weights for the observations belonging to the particular cluster.

cluster An integer index of the particular cluster

J The number of sub-models to be built and tested for a particular cluster.
sample.df Degree of freedom for the t-distributions in a t-mixture model. Has to be 5 in immunoClust.

Details

These function are used internally by the cell-clustering procedures of cell.process in immunoClust and are not intended to be used directly.

Value

The cluster parameters of the refined model in an object of class immunoClust.

Author(s)

Till Sörensen <till-antoni.soerensen@charite.de>

References


See Also

cell.process, cell.hclust

Examples

data(dat.fcs)
data(dat.exp)
dat.trans <- trans.ApplyToData(dat.exp[[1]], dat.fcs)
# need to re-calculate the cluster membership probabilities
# not stored in dat.exp
r1 <- cell.Classify(dat.exp[[1]], dat.trans)
summary(r1)
r2 <- cell.SubClustering(r1, dat.trans)
summary(r2)
**Description**

A vector of `immunoClust-objects` with `cell.process` clustering results of five samples.

**Usage**

```r
data("dat.exp")
```

**Details**

Cell-event clustering was performed on reduced (10,000 events) sample data of the dataset of `immunoClust`, MACS-depleted populations datasets 2010. URL [http://flowrepository.org/id/FR-FCM-ZZWB](http://flowrepository.org/id/FR-FCM-ZZWB).

**Value**

A vector of 5 `immunoClust-objects` for the cell clustering results of 5 FC samples.

- [[1]] CD19 MACS-depleted cells
- [[2]] CD15 MACS-depleted cells
- [[3]] CD14 MACS-depleted cells
- [[4]] CD4 MACS-depleted cells
- [[5]] CD3 MACS-depleted cells

**Source**

[http://flowrepository.org/id/FR-FCM-ZZWB](http://flowrepository.org/id/FR-FCM-ZZWB)

**Examples**

```r
data(dat.exp)

## process meta clustering
meta <- meta.process(dat.exp, meta.bias=0.6)

## extract event counts in the 5 samples for all meta clusters
res <- meta.numEvents(meta)
```
dat.fcs

**immunoClust Cell-clustering Sample**

**Description**
flowFrame data sample with 10,000 events in 7 parameters.

**Usage**
data(dat.fcs)

**Details**
This FCS sample is a reduced (10,000 events) dataset in flowFrame format of the first sample in the dataset of immunoClust, MACS-depleted populations datasets 2010. URL http://flowrepository.org/id/FR-FCM-ZZWB.

**Value**
A flowCore flowFrame with 10,000 observations on the following 7 parameters.

- **FCS-A** Forward scatter
- **SSC-A** Sideward scatter
- **FITC-A** CD14
- **PE-A** CD19
- **APC-A** CD15
- **APC-Cy7-A** CD4
- **Pacific Blue-A** CD3

**Source**
http://flowrepository.org/id/FR-FCM-ZZWB

**Examples**
data(dat.fcs)
show(dat.fcs)

```
## Not run:
## process cell clustering
dat.res <- cell.process(dat.fcs)

## apply asinh-transformation
dat.fcs.transformed <- trans.ApplyToData(dat.res, dat.fcs)

## plot results
splom(dat.res, dat.fcs.transformed, N=1000)

## End(Not run)
```
**dat.meta**

---

**Description**

The Meta-clustering result of the `dat.exp` data set.

**Usage**

```r
data("dat.meta")
```

**Details**

The Meta-clustering was performed with an ICL-bias of 0.4.

**Value**

A list-object containing the meta-clustering result. A detailed description is documented in the value section for the `meta.process` function.

**Source**

http://flowrepository.org/id/FR-FCM-ZZWB

**Examples**

```r
data(dat.meta)

## extract event counts in the 5 samples for all meta clusters
res <- meta.numEvents(dat.meta)
```

---

**generics.immunoclust**

---

**Description**

Collection of generic function definitions used in immunoClust either for an `immunoClust` or an `immunoMeta` object.
Usage

nsam(object, ...)

sam_ncls(object, ...)

sam_clsWeights(object, ...)

sam_clsEvents(object, ...)

sam_clsMu(object, ...)

sam_clsSigma(object, ...)

nobs(object, ...)

npar(object, ...)

ncls(object, ...)

weights(object, ...)

mu(object, ...)

sigma(object, ...)

label(object, ...)

aposteriori(object, ...)

subset(x, ...)

parameters(object, ...)

transformParams(object, ...)

clusterCoeff(object, ...)

clusterDist(object, ...)

clusterProb(object, ...)

Arguments

object, x an object to apply the function.

... addional options to be passed to methods
**Value**

The appropriate value for the specific cal (see section Details).

**Details**

- **nsam** returns the number of cell-event immunoClust-objects co-clustered in the immunoMeta-object.
- **sam_clsWeights** returns the cluster weights of all samples cell-clusters.
- **sam_clsEvents** returns the cluster event numbers of all samples cell-clusters.
- **sam_clsMu** returns the cluster means of all samples cell-clusters.
- **sam_clsSigma** returns the cluster co-variance matrices of all samples cell-clusters.
- **nobs** already generic in stats. Here, returns the number of clustered objects either cell-events or cell-clusters in cell event or meta clustering.
- **npar** returns the number of parameters used for clustering.
- **ncls** returns the number of clusters, either cell-event cluster or meta-cluster.
- **weights** already generic in stats. Here, returns the weights of the mixture models for the cell-event of meta-clustering.
- **mu** returns the cluster means.
- **sigma** already generic in stats. Here, returns the co-variance matrices of the clusters.
- **label** returns the cluster label, i.e. the assignment of the clustered objects to the clusters.
- **aposteriori** returns the a posteriori probabilities of cluster membership for the clustered objects.
- **events** returns the number of cell-events for the clusters.
- **subset** already generic in stats. Here, returns an object with mixture model on a subset of parameters.
- **parameters** already generic in flowCore. Here, lists the parameters used for clustering.
- **parameters<-** Modifies the list of parameters used for clustering.
- **transformParam** return an object with transformed mixture model parameters.
- **clusterCoeff** returns the bhattacharrya coefficient of meta clusters for a meta level.
- **clusterDist** returns the bhattacharrya distance of meta clusters for a meta level.
- **clusterProb** returns the bhattacharrya probability of meta clusters for a meta level.

**Author(s)**

Till Sörensen <till-antoni.soerensen@charite.de>

**See Also**

immunoClust, immunoMeta
immunoClust-object

---

**Description**

The `immunoClust` object contains the clustering results in the `immunoClust`-pipeline as obtained by `cell.process` or `meta.process`.

**Usage**

```r
## S4 method for signature 'immunoClust'
summary(object)
## S4 method for signature 'immunoClust'
show(object)
```

**Arguments**

- `object` An object of class `immunoClust` as returned by the `cell.process` or `meta.process` functions of the `immunoClust`-pipeline.

**Value**

An object of class `immunoClust` has the following slots:

- `expName` The name of the clustering experiment.
- `fcsName` The path of the clustered FCS-file.
- `parameters` The parameters used for clustering.
- `removed.below` Number of observations removed from below.
- `removed.above` Number of observations removed from above.
- `trans.a` The $P$-dimensional vector of the scaling factors for the asinh-transformation of each used parameter. A scaling factor of 0 indicates that a parameter is not transformed.
- `trans.b` The $P$-dimensional vector of the translations for the asinh-transformation of each used parameter.
- `trans.decade` experimental; should be -1.
- `trans.scale` experimental; should be 1.0.
- `K` The number of clusters.
- `N` The number of observations.
- `P` The number of used parameters.
- `w` The $K$-dimensional vector of the mixture proportions.
- `mu` The $K \times P$-dimensional matrix of the $K$ estimated cluster means.
- `sigma` The $K \times P \times P$-dimensional matrix of the $K$ estimated cluster covariance matrices.
- `z` The $K \times N$-dimensional matrix containing the a-posteriori probabilities of cluster membership.
- `label` The $N$-dimensional vector containing the maximum a posteriori estimator for cluster membership.
- `logLike` A vector of length 3 containing the BIC, ICL and the classification likelihood without penalty of the fitted model.
- `BIC` The Bayesian Information Criterion for the fitted mixture model.
- `ICL` The Integrate Classification Likelihood for the fitted model.
- `history` experimental; unused so far.
- `state` experimental; unused so far.
**immunoMeta-class**

**Author(s)**
Till Sörensen <till-antoni.soerensen@charite.de>

**References**

**See Also**
cell.process, meta.process

**Examples**
```r
data(dat.exp)
summary(dat.exp[[1]])
```

---

**Description**

The `immunoMeta` object contains the clustering results in the `immunoClust`-pipeline obtained by `meta.process`. Additionally, it offers methods to structure the meta-clusters and build up a hierarchical annotation tree.

**Usage**
```r
immunoMeta(res, dat, gating)
## S3 method for class 'immunoMeta'
summary(object, ...)
## S3 method for class 'immunoMeta'
show(object)
```

**Arguments**

- `res` An `immunoClust` object as a result of the meta-clustering.
- `dat` The data on which the meta-clustering was performed.
- `gating` a hierarchical structure annotation of the meta-clusters.
- `object` An object of class `immunoMeta` as returned by the `meta.process` functions of the `immunoClust`-pipeline.
- `...` additional options for underlying methods.
Value

An object of class `immunoMeta` has the following slots:

- `dat.clusters`: A dat list-object of the cell event clusters used for meta-clustering.
- `res.clusters`: The `immunoClust-object` of the fitted meta-clustering mixture model.
- `dat.scatter`: A dat list-object of the scatter parameters for the cell event clusters used for scatter clustering.
- `res.scatter`: The `immunoClust-object` of the fitted scatter-clustering mixture model.
- `gating`: A list-object containing the hierarchical annotation-tree.

The components of the `dat` list-objects are:

- `P`: The number of parameters for the cell event clusters.
- `N`: The number of cell-clustering experiments.
- `K`: The $N$-dimensional vector with the numbers of cell event clusters in each experiment. The total number of clusters is $\sum_{i=1}^{N} K_i$.
- `W`: The $\text{tot}K \times P$-dimensional vector with the mixture proportions of all clusters.
- `M`: The $\text{tot}K \times P$-dimensional matrix of all cluster means.
- `S`: The $\text{tot}K \times P \times P$-dimensional matrix of all cluster covariance matrices.
- `expNames`: The $N$-dimensional character vector with the cell-clustering experiment names.
- `expEvents`: The $N$-dimensional vector with the numbers of events in each cell-clustering experiment.
- `clsEvents`: The $\text{tot}K$-dimensional vector with the number of events in each cluster.
- `desc`: The $P$-dimensional character vector with the parameter description.

Author(s)

Till Sörensen <till-antoni.soerensen@charite.de>

See Also

- `meta.process`

Examples

```
data(dat.meta)
summary(dat.meta)
```

Description

This function provides a direct access to the meta-clustering procedure. The method described and discussed in this manuscript is the EMT-classification (EM-method=20) with the number of events for each cluster as weights. It returns the fitted mixture model parameter in an object of class `immunoClust`.
Usage

```r
meta.Clustering(P, N, K, W, M, S, label=NULL, I.iter=10, B=500, tol=1e-5,
    bias=0.25, sub.thres = bias, alpha=0.5, EM.method=20,
    norm.method=0, norm.blur=2, norm.minG=10, verbose=FALSE)
```

Arguments

- **P**: The number of observed parameters for the cell event clusters.
- **N**: The number of cell-clustering experiments.
- **K**: The \(N\)-dimensional vector with the numbers of cell event clusters in each experiment. The total number of clusters is \(\text{tot}K = \sum_{i=1}^{K} K_i\).
- **W**: The \(\text{tot}K\)-dimensional vector with weights of all clusters.
- **M**: The \(\text{tot}K \times P\)-dimensional matrix of all cluster means.
- **S**: The \(\text{tot}K \times P \times P\)-dimensional matrix of all cluster covariance matrices.
- **label**: Optional initial cluster assignment. If label is NULL all clusters are assigned in one cluster in the initial clustering step.
- **I.iter**: The maximum number of major iteration steps.
- **B**: The \(\text{tot}K \times P \times P\)-dimensional matrix of all cluster covariance matrices.
- **tol**: The tolerance used to assess the convergence of the EM(t)-algorithms.
- **bias**: The ICL-bias used in the EMt-iteration of the meta-clustering.
- **sub.thres**: Defines the threshold, below which an ICL-increase is meaningless. The threshold is given as the multiple (or fraction) of the costs for a single cluster.
- **alpha**: A value between 0 and 1 used to balance the bhattacharrya probabilities calculated with either the full covariance matrices or using only the diagonal elements of it. When working with uncompensated FC data very high correlations between parameters may be observed due to spill over. This leads to a very low bhattacharrya probability for two clusters even if they are located nearby. Using a mixture of the probabilities calculated with the complete covariance matrices and the variance information of each parameter avoids this problem. With a value of alpha=1, only the probabilities with complete covariance matrices are applied. A reasonable value for alpha is 0.5.
- **EM.method**: 0 = KL-minimization not weighted
  1 = BC-maximization not weighted
  10 = BC-maximization weighted
  2 = EMt-classification not weighted
  20 = EMt-classification weighted
- **norm.method**: Normalization function; see `meta.Normalize` for details.
- **norm.blur**: For the normalization step the a-posteriori probabilites of the cell-clusters belonging to a meta.clusters a used. In order to capture narrow cell-clusters reasonable the co-variance of the cell-clusters is blured for the a-posteriori probabilities in the normalization step.
- **norm.minG**: Minimum number of obtained meta-clusters required to process the normalization step in the major iteration loop.
- **verbose**: Detailed messages during process
**Details**

This function is used internally by the meta-clustering procedure `meta.process` in `immunoClust`.

**Value**

The fitted model information in an object of class `immunoClust`.

**Author(s)**

Till Sörensen <till-antoni.soerensen@charite.de>

**References**


**See Also**

`immunoClust-object`, `meta.SubClustering`, `meta.process`

**Examples**

```r
data(dat.exp)
d <- meta.exprs(dat.exp)
res <- meta.Clustering(d$P, d$N, d$K, d$clsEvents, d$M, d$S)
```

---

**Description**

These functions collect the output of the `meta.process` and extracts the event numbers, relative frequencies or mean fluorescence intensities for each meta-cluster and cell-clustering experiment in a numeric table.

**Usage**

```r
meta.numEvents(meta, out.all=TRUE, out.removed=FALSE, out.unclassified=TRUE)
meta.relEvents(meta, out.all=TRUE, out.removed=FALSE, out.unclassified=TRUE)
meta.relParent(meta, out.all=TRUE, out.unclassified = TRUE)
meta.parMFI(meta, par, out.all=TRUE, out.unclassified = TRUE)
meta.numClusters(meta, out.all=TRUE)
meta.freqTable(meta)
```
meta.export

Arguments

- **meta**: The list-object returned by the function `meta.process`.
- **par**: An integer index to the specific parameter.
- **out.all**: A numeric indicator whether the event numbers of all hierarchical gating levels are obtained or only the meta-clusters themselves.
- **out.removed**: A numeric indicator whether the number of removed events, which are not used for clustering are exported.
- **out.unclassified**: A numeric indicator whether the event numbers of the hierarchical gating levels or all meta-clusters are exported.

Value

A numeric matrix with

- **numEvents**: the number of cell events
- **relEvents**: relative frequencies, i.e. the number of cell events per total measured events
- **relParent**: relative frequencies according to parent relationship in the annotated hierarchy.
- **parMFI**: mean fluorescence intensities in one parameter, i.e. the meta-cluster centers in asinh-transformed scale
- **numClusters**: the number of cell clusters
- **freqTable**: relative frequencies with respect to all gating hierarchie levels

in each meta-cluster (and gating hierarchy level) for each cell-clustering experiment.

Author(s)

Till Sörensen <till-antoni.soerensen@charite.de>

References


See Also

- `meta.process`

Examples

```r
data(dat.exp)
meta <- meta.process(dat.exp)
tbl <- meta.numEvents(meta)
```
Collecting Data of an `immunoClust` vector

Description

The function takes a vector of `immunoClust`-object obtained by the `cell.process` function and extracts this information into a list object.

Usage

```r
meta.exprs(exp, sub=c())
```

Arguments

- `exp` The vector of `immunoClust` object with the cell clustering results.
- `sub` A integer array indicating the parameter subset to be collected.

Value

A list object with the following slots:

- `P` The number of observed parameters for the cell event clusters.
- `N` The number of cell-clustering samples.
- `K` The \(N\)-dimensional vector with the numbers of cell event clusters in each experiment. The total number of clusters is \(totK = \sum_{i=1}^{N} K_i\).
- `W` The \(totK\)-dimensional vector with weights of all clusters.
- `M` The \(totKxP\)-dimensional matrix of all cluster means.
- `S` The \(totKxPxP\)-dimensional matrix of all cluster covariance matrices.
- `expNames` The \(N\)-dimensional vector with the experiment names of the cell clustering samples.
- `expEvents` The \(N\)-dimensional vector for the total number of events of the cell clustering samples.
- `clsEvents` The \(totK\)-dimensional vector for the event number of all clusters.
- `desc` The \(P\)-dimensional vector for the parameter description.

Author(s)

Till Sörensen <till-antoni.soerensen@charite.de>

References

See Also

immunoClust.

Examples

data(dat.exp)
d <- meta.exprs(dat.exp, sub=c(1,2))

Description

Performs agglomerative clustering on cell-clusters. It is used in the interactive meta-clustering approach of immunoClust to obtain an initial meta-cluster membership for the EM(t)-iteration.

Usage

meta.hclust(P, N, W, M, S)

Arguments

P The number of parameters.
N The number of clusters.
W The $N$-dimensional vector with cluster weights, i.e. numbers of events in a cluster.
M The $N \times P$-dimensional vector with cluster means.
S The $N \times P \times P$-dimensional vector with cluster covariance matrices.

Details

This function is used internally in meta.TestSubCluster of immunoClust.

Value

A numeric $(N - 1) \times 2$-dimensional matrix which gives the minimum index for observations in each of the two clusters merged at the $i$th step in each row.

Note

The merging distances need not to be monotonic increasing.

Author(s)

Till Sörensen <till-antoni.soerensen@charite.de>
References


See Also

`meta.TestSubCluster, meta.process`

Examples

```r
data(dat.exp)
 r <- dat.exp[[1]]
 #hcPairs <- meta.hclust(r@Q, r@K, r@w, r@mu, t(apply(r@sigma,1,c)))
 hcPairs <- meta.hclust(npar(r), ncls(r), weights(r),
   mu(r), t(apply(sigma(r),1,c)))
```

---

**meta.ME**  
*immunoClust EM(t)-iteration on Cell-clusters*

Description

Performs an EM(t)-iteration on cell-clusters given an initial meta-cluster membership for the cell-clusters and returns the fitted meta-clusters information in an object of class *immunoClust*.

Usage

```r
meta.ME(P, N, K, W, M, S, label, B=100, tol=1e-5, method=20, bias=0.25, alpha=0.5, min.class=0)
```

Arguments

- **P**: The number of observed parameters for the cell event clusters.
- **N**: The number of cell-clustering experiments.
- **K**: The $N$-dimensional vector with the numbers of cell event clusters in each experiment. The total number of clusters is $\text{totK} = \sum_{i=1}^{K} K_i$.
- **W**: The $\text{totK}$-dimensional vector with weights, i.e. number of events, of all clusters.
- **M**: The $\text{totK} \times P$-dimensional matrix of all cluster means.
- **S**: The $\text{totK} \times P \times P$-dimensional matrix of all cluster covariance matrices.
- **label**: The $\text{totK}$-dimension integer vector with the initial cell-cluster to meta-cluster membership.
- **B**: The $\text{totK} \times P \times P$-dimensional matrix of all cluster covariance matrices.
- **tol**: The tolerance used to assess the convergence of the EM(t)-algorithms.
**meta.ME**

<table>
<thead>
<tr>
<th>method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>KL-minimization not weighted</td>
</tr>
<tr>
<td>1</td>
<td>BC-maximization not weighted</td>
</tr>
<tr>
<td>10</td>
<td>BC-maximization weighted</td>
</tr>
<tr>
<td>2</td>
<td>EMt-classification not weighted</td>
</tr>
<tr>
<td>20</td>
<td>EMt-classification weighted</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>bias</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>The ICL-bias used in the EMt-iteration of the meta-clustering.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>alpha</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>A value between 0 and 1 used to balance the bhattacharrya probabilities calculated with either the full covariance matrices or using only the diagonal elements of it.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>min.class</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>The minimum number of clusters for the final model.</td>
</tr>
</tbody>
</table>

**Details**

This function is used internally by the meta-clustering procedures `meta.process` and `meta.Clustering` in `immunoClust`.

**Value**

The fitted meta-clusters information in an object of class `immunoClust`.

**Author(s)**

Till Sörensen <till-antoni.soerensen@charite.de>

**References**


**See Also**

`meta.process`, `meta.Clustering`

**Examples**

```r
data(dat.exp)
d <- meta.exprs(dat.exp)
r <- meta.ME(d$P, d$N, d$K, d$clsEvents, d$M, d$S, label=rep(1,sum(d$K)))```

**Description**

Performs a normalization via linear regression of the cell-cluster samples to the meta-clustering model.

**Usage**

```r
```

**Arguments**

- `P`: The number of observed parameters for the cell event clusters.
- `N`: The number of cell-clustering experiments.
- `K`: The \( N \)-dimensional vector with the numbers of cell event clusters in each experiment. The total number of clusters is \( \text{tot}K = \sum_{i=1}^{K} K_i \).
- `W`: The \( \text{tot}K \)-dimensional vector with weights, i.e. number of events, of all clusters.
- `M`: The \( \text{tot}K \times P \)-dimensional matrix of all cluster means.
- `S`: The \( \text{tot}K \times P \times P \)-dimensional matrix of all cluster covariance matrices.
- `G`: The number of meta-clusters.
- `Z`: The \( \text{tot}K \times G \)-dimensional matrix with the a-posteriori probabilities for a cell-cluster belonging to a meta-cluster.
- `method`: Alternative methods used for the normalization routine. Let \( Y \) denote the consensus meta-model build from all cell-event clusters of all experiments using the a-posteriori \( Z \) and \( X \) the cell-event clusters in each experiment.
  - \( 0 = \text{no normalization} \)
  - \( 1 = Y = a \times X \)
  - \( 2 = Y = a \times X + b \)
  - \( 3 = X = a \times Y \)
  - \( 4 = X = a \times Y + b \)

**Details**

The regression used the cell-cluster and meta-cluster means weighted by the probabilities for a cell-cluster belonging to the meta-cluster. It builds a consensus meta-model from all cell-clusters using the a-posteriori probabilities \( Z \).
Value

Returns the normalized cell-clusters means and co-variance matrices in a list-object with the following slots:

- **P** The number of observed parameters for the cell event clusters.
- **N** The number of cell-clustering experiments.
- **K** The \( N \)-dimensional vector with the numbers of cell event clusters in each experiment. The total number of clusters is \( \sum_{i=1}^{K} K_i \).
- **W** The \( \sum K \)-dimensional vector with weights, i.e. number of events, of all clusters.
- **M** The \( \sum K \times P \)-dimensional matrix of all cluster means.
- **S** The \( \sum K \times P \times P \)-dimensional matrix of all cluster covariance matrices.

Author(s)

Till Sörensen <till-antoni.soerensen@charite.de>

See Also

- `meta.process`
- `meta.Clustering`

Examples

```r
data(dat.meta)
#dat <- dat.meta$dat.clusters
res <- dat.meta$res.clusters

dat.norm <- meta.Normalize(npar(dat.meta), nsam(dat.meta),
    sam_ncls(dat.meta), sam_clsEvents(dat.meta), sam_clsMu(dat.meta),
    sam_clsSigma(dat.meta), ncls(res), aposteriori(res))
```

Description

This function performs iterative model based clustering on the clusters obtained by `cell.process` of several samples. Its input is a vector of the immunoClust-objects of the samples.

Usage

```r
meta.process(exp, dat.subset=c(), meta.iter=10, tol=1e-05, meta.bias=0.2,
    meta.alpha=.5, norm.method=0, norm.blur=2, norm.minG=10)
```
Arguments

exp
A vector of list objects, each list contains the cell-clustering result of a sample in the res field. Addition fields are name and fsc containing the cell-sample name and fcs-filename, which are used for data output and plot routines.

dat.subset
A numeric vector defining the used observed parameters for the meta-clustering. If unset, all parameters in the cell-clustering results are used.

meta.iter
The number of major iterations.

tol
The tolerance used to assess the convergence of the EM(t)-algorithms.

meta.bias
The ICL-bias used in the EMt-iteration of the meta-clustering.

meta.alpha
A value between 0 and 1 used to balance the bhattacharrya probabilities calculated with either the full covariance matrices or using only the diagonal elements of it. When working with uncompensated FC data, very high correlations between parameters may be observed due to spill over. This leads to a very low bhattacharrya probability for two clusters even if they are located nearby. Using a mixture of the probabilities calculated with the complete covariance matrices and the variance information of each parameter avoids this problem. With a value of alpha=1, only the probabilities with complete covariance matrices are applied. A reasonable value for alpha is 0.5.

norm.method
A numeric selector for the normalization step to be performed during the major iteration.

norm.blur
The bluring constant by which the cell-clusters co-variance matrices are increased within the normalization step.

norm.minG
Minimum number of meta-clusters required before processing the normalization step.

Value

The function returns a immunoMeta with the following components:

dat.clusters
A dat list-object of the cell event clusters used for meta-clustering.

res.clusters
The immunoClust-object of the fitted meta-clustering mixture model.

dat.scatter
A dat list-object of the scatter parameters for the cell event clusters used for scatter clustering.

res.scatter
The immunoClust-object of the fitted scatter-clustering mixture model.

gating
A list-object containing the hierarchical gating-tree.

Author(s)

Till Sörensen <till-antoni.soerensen@charite.de>

References

See Also

`immunoMeta-object, immunoClust-object, meta.Clustering, meta.export, cell.process`

Examples

```r
data(dat.exp)
meta <- meta.process(dat.exp)
summary(meta)
tbl <- meta.numEvents(meta)
```

### meta.regnorm

**immunoClust normalization procedure**

**Description**

Performs a normalization via linear regression of the sample clusters in x to the clusters in y.

**Usage**

```r
meta.Regnorm(y, x, method=1, alpha=0.5)
```

**Arguments**

- **y**: `immunoClust-object` with the destination clusters.
- **x**: `immunoClust-object` with the cluster to normalize.
- **method**: Alternative methods used for the normalization routine.
  1. $X = a \times Y$
  2. $X = a \times Y + b$
- **alpha**: A value between 0 and 1 used to balance the bhat-attacharya probabilities calculated with either the full covariance matrices or using only the diagonal elements of it.

**Value**

Returns the normalized cell-clusters means and co-variance matrices in a list-object with the following slots:

- **P**: The number of observed parameters for the cell event clusters.
- **N**: The number of cell-clustering experiments.
- **K**: The $N$-dimensional vector with the numbers of cell event clusters in each experiment. The total number of clusters is $\text{tot}K = \sum_{i=1}^{K} K_i$.
- **M**: The $\text{tot}K \times P$-dimensional matrix of all cluster means.
- **S**: The $\text{tot}K \times P \times P$-dimensional matrix of all cluster covariance matrices.
Author(s)
Till Sörensen <till-antoni.soerensen@charite.de>

Examples
```r
data(dat.meta)
data(dat.exp)
dat.norm <- meta.RegNorm(dat.meta$res.clusters, dat.exp[[1]])
```

Description
The meta.SON.clustering is an extension of the meta-clustering process co-clustering several samples cluster results. It integrates a SON normalization step between the meta-clustering iterations.

Usage
```r
meta.SON.clustering(
  meta,
  cycles=6, alpha=0.5, scale.factor=2, scale.steps=0,
  meta.iter=2, meta.bias=0.3, meta.tol=1e-5,
  SON.cycles=1, SON.rlen=100, SON.deltas=c(1/SON.rlen,1/SON.rlen),
  SON.blurring=c(2,0.1),
  verbose=FALSE
)
```

Arguments
- `meta`: an immunoMeta-object for which the clustering should be refined.
- `cycles`: number of major iteration steps.
- `alpha`: The alpha value for calculation the bhattacharyya probabilities.
- `scale.factor`: scale factor for the internal model scaling step.
- `scale.steps`: scale steps for the internal model scaling step. 0 means no model scaling.
- `meta.iter`: number of iterations for meta-clustering step
- `meta.bias`: ICL bias for meta-clustering step
- `meta.tol`: maximal tolerance for meta-clustering step
- `SON.cycles`: number of cycles in SON normalization step.
- `SON.rlen`: runlength in SON normalization step
- `SON.deltas`: deltas parameter in SON normalization step
- `SON.blurring`: bluring parameter in SON normalisation step
- `verbose`: detailed messages during process
**Details**

For the refined meta.SON.clustering process a simple meta.process should be performed first. The resulting immunoMeta-object then serves as input data for the meta.SON.clustering.

Within the meta.SON.clustering between two meta.Clustering steps a SON normalization step is performed, which shifts the clusters of each sample towards the meta-clusters. The SON normalization for a sample consists of an optional first step to scale the model build by meta clusters best possible to the sample clusters. Afterwards, the meta clusters are moved to towards the sample clusters. This is done in a similar way to SOM clustering mapping. Finally, the sample clusters are retracted to the meta-clusters distribution. For this purpose the Bhattacharyya probabilities of sample and meta clusters are used.

**Value**

An immunoMeta-object for the co-clustering result.

**Author(s)**

Till Sörensen <till.soerensen@bioretis.com>

**References**

pre-print

**See Also**

meta.Clustering

**Examples**

```r
data(dat.meta)
meta <- meta.SON.clustering(dat.meta, cycles=2)
```

---

**meta.SON.combineClustering**

Transfer the annotation of an immunoMeta-object to an immunoClust-object.

**Description**

An immunoMeta-object is co-clustered with an immunoClust-object of the same parameter structure. Co-clustering includes SON normalization steps. The returned immunoCLust-object contains the meta-clusters unchanged in order and numeration.
Usage

```r
meta.SON.combineClustering(
  meta, res, par=seq_len(npar(meta)),
  map.cluster=seq_len(ncls(meta)),
  use.cluster=seq_len(ncls(res)),
  meta.alpha=0.5, meta.bias=0.1, meta.iter=100, meta.tol=1e-5,
  SON.method=1, SON.cycles=4, SON.rlen=10,
  SON.deltas=c(1/SON.rlen,1/SON.rlen), SON.blurring=c(2,1),
  traceG=c(), traceK=c())
```

Arguments

- `meta`: The annotated immunoMeta-object.
- `res`: An immunoClust-object as results from cell-event clustering for a sample.
- `par`: An integer array with the parameters to be used for SON mapping.
- `map.cluster`: The model clusters to be used for SON mapping.
- `use.cluster`: The sample clusters to be used for SON mapping.
- `meta.alpha`: The alpha value in calculation the bhattacharyya probabilities.
- `meta.bias`: The ICL bias for meta co-clustering step.
- `meta.iter`: Maximal iterations in the meta co-clustering step.
- `meta.tol`: Maximal tolerance for meta co-clustering step.
- `SON.method`: Method selection for SON normalization step.
- `SON.cycles`: Number cycles in the SON normalization step.
- `SON.rlen`: runlength in the SON normalization step.
- `SON.deltas`: delta parameter in the SON normalization step.
- `SON.blurring`: blurring parameter in the SON normalization step.
- `traceG`: An array of model cluster to trace in the process.
- `traceK`: An array of sample cluster to trace in the process.

Details

The co-clustering consists of a normalization and meta-clustering step. A sample cluster is than labeled according to its corresponding meta cluster. The SON-normalization and meta-clustering steps are parameterised by the SON and meta arguments.

Value

An immunoClust-object from meta-clusters and combined observation from meta- and samples-cluster. The first G elements of the label coresponds to the meta-clusters, afterwards the labelling of the samples-clusters indicates the nearest meta-cluster for the sample-cluster.

Author(s)

Till Sörensen <till.soerensen@bioretis.com>
**meta.SubClustering**

**References**

in progress

**See Also**

`meta.Clustering`

**Examples**

```r
data(dat.exp)
data(dat.meta)
res <- meta.SON.combineClustering(dat.meta, dat.exp[[1]], SON.cycles=2)
```

**Description**

These functions test each meta-cluster of a model for refining it into more sub-clusters and return the refined cluster memberships in an integer array.

**Usage**

```r
meta.SubClustering(x, P, N, W, M, S, tol=1e-5, bias=0.25, thres=bias,
                   alpha=1.0, EM.method=20, verbose=FALSE)
```

```r
meta.TestSubCluster(x, P, N, W, M, S, J=8, B=500, tol=1e-5, bias=0.5,
                     alpha=1.0, EM.method=2, HC.samples=2000)
```

**Arguments**

- `x` An immunoClust object with the initial model parameter \((K, label)\).
- `P` The number of parameters.
- `N` The number of clusters.
- `W` The \(N\)-dimensional vector with cluster weights, i.e. numbers of events in a cluster.
- `M` The \(N \times P\)-dimensional vector with cluster means.
- `S` The \(N \times P \times P\)-dimensional vector with the cluster covariance matrices.
- `tol` The tolerance used to assess the convergence of the EM(t)-algorithms in SubClustering.
- `bias` The ICL-bias used in the EMt-algorithm.
- `thres` Defines the threshold, below which an ICL-increase is meaningless. The threshold is given as the multiple (or fraction) of the costs for a single cluster.
alpha  A value between 0 and 1 used to balance the bhattacharrya probabilities calculated with either the full covariance matrices or using only the diagonal elements of it.

J   The number of sub-models to be built and tested for a particular cluster.

B   The maximum number of EM(t)-iterations in Sub-Clustering.

EM.method
0 = KL-minimization not weighted
1 = BC-maximization not weighted
10 = BC-maximization weighted
2 = EMt-classification not weighted
20 = EMt-classification weighted

HC.samples  The number of samples used for initial hierarchical clustering.

verbose  detailed messages during process

Details
These function are used internally by the meta-clustering procedures meta.process and meta.Clustering in immunoClust and are not intended to be used directly.

Value
An integer array of length \(N\) containing the cell-clusters meta-cluster memberships of the refined model.

Author(s)
Till Sörensen <till-antoni.soerensen@charite.de>

References

See Also
meta.process, meta.Clustering, meta.hclust

Examples
```r
data(dat.exp)
d <- meta.exprs(dat.exp)
#label <- rep(1,sum(d$K))
#label <- meta.SubClustering(d$P, sum(d$K), d$clsEvents, d$M, d$S, label=label)

r0 <- new("immunoClust", K=sum(d$K), label=rep(1,sum(d$K)))
label <- meta.SubClustering(r0, d$P, sum(d$K), d$clsEvents, d$M, d$S)

r1 <- meta.ME(d$P, d$N, d$K, d$clsEvents, d$M, d$S, label)
```
**Methods.immunoClust**

Acessors and Methods for immunoClust Objects

**Description**

Documentation of the accessors and methods for immunoClust-objects

**Arguments**

- object, immunoClust
  - an object of class `immunoClust` as return by `cell.process`.
- cls
  - cluster subset for retrieved slot values.
- par
  - parameter subset for retrieved slot values.

**Accessors**

- **nobs** the number of cell events clustered
  
  *Usage:*
  
  `nobs(immunoClust)`

- **ncls** the number of clusters.
  
  *Usage:*
  
  `ncls(immunoClust)`

- **npar** the number of parameters measured, cell-clustered
  
  *Usage:*
  
  `npar(immunoClust)`

- **parameters, parameters<-** extracts or replaces the names of measured, cell-clustered parameters
  
  *Usage:*
  
  `parameters(immunoClust)`
  
  `parameters(immunoClust) <- value`

- **label** the clustering label, that is the assignment of the cell-events to the clusters.
  
  *Usage:*
  
  `label(immunoClust)`

- **weights** the clustering weights for the cluster selection (all cluster by default)
  
  *Usage:*
  
  `weights(immunoClust, cls=seq_len(ncls(immunoClust)))`

- **mu** the cluster mean values for the cluster and parameter selection (all cluster and all parameter by default)
  
  *Usage:*
  
  `mu(immunoClust, cls=seq_len(ncls(immunoClust)), par=seq_len(npar(immunoClust)))`

- **sigma** the cluster co-variance values for the cluster and parameter selection (all cluster and all parameter by default)
  
  *Usage:*
  
  `sigma(immunoClust, cls=seq_len(ncls(immunoClust)), par=seq_len(npar(immunoClust)))`
**aposteriori**  the a-posteriori probabilities of cluster membership for each event

*Usage:*

```r
aposteriori(immunoClust)
```

**events**  the cell-event numbers for the cluster selection (all cluster by default)

*Usage:*

```r
events(immunoClust, ncls=seq_len(ncls(immunoClust)))
```

**cells**  the cell-events indices in the FCS-file for the cluster selection (all cluster by default). If `na.rm` is `TRUE` the removed events are omitted and the indices fits to the a-posteriori matrix \( z \) in the `immunoClust`-object

*Usage:*

```r
cells(immunoClust, ncls=seq_len(ncls(immunoClust)), na.rm=FALSE)
```

**Methods**

**subset**  Builds the `immunoClust`-object for a parameter subset

*Usage:*

```r
res <- subset(immunoClust, par)
```

**transformParams**  Scales and translates the cluster means of the `immunoClust`-object in each parameter

*Usage:*

```r
res <- transformParams(immunoClust, scale=c(), offset=c())
```

**Author(s)**

Till Sörensen <till-antoni.soerensen@charite.de>

**See Also**

`immunoClust-object`

**Examples**

```r
###
data(dat.exp)
## cell.clustering result for dat.fcs
res <- dat.exp[[1]]
nobs(res)
ncls(res)
```
**Description**

Documentation of the accessors and methods for immunoMeta-objects

**Arguments**

- **object**, **immunoMeta**
  - an object of class `immunoMeta` as returned by `meta.process`.
- **cls**
  - cluster subset for retrieved slot values.
- **par**
  - parameter subset for retrieved slot values.
- **pos**
  - Gives the position in the immunoMeta-hierarchy. `pos` is an array of indices which addresses the level of interest. Each level in the immunoMeta-hierarchy consists of a name (`desc`), meta-cluster subset (array of cluster indices) and a vector of sub-levels. `pos` is the sequence of indices into these sub-levels beginning at root level.

**Accessors**

- **nsam**
  - the number of immunoClust-objects (samples) which are co-clustered.
    - *Usage:*
      ```r
      nsam(immunoMeta)
      ```
- **sam_ncls**
  - the number of cell event clusters in the immunoClust-objects (samples) which are co-clustered.
    - *Usage:*
      ```r
      sam_ncls(immunoMeta, for.samples=seq_len(nsam(meta))
      ```
- **sam_clsWeights**
  - the weights of all cell event clusters which are collected for co-clustering.
    - *Usage:*
      ```r
      sam_clsWeights(immunoMeta)
      ```
- **sam_clsMu**
  - the means of all cell event clusters which are collected for co-clustering.
    - *Usage:*
      ```r
      sam_clsMu(immunoMeta)
      ```
- **sam_clsSigma**
  - the co-variance matrices of all cell event clusters which are collected for co-clustering.
    - *Usage:*
      ```r
      sam_clsSigma(immunoMeta)
      ```
- **sam_clsEvents**
  - the event numbers of all cell event clusters which are collected for co-clustering.
    - *Usage:*
      ```r
      sam_clsEvents(immunoMeta)
      ```
- **nobj**
  - the number of cell events clusters from sample cell-clustering which are co-clustered.
    - *Usage:*
      ```r
      nobj(immunoMeta)
      ```
ncls  the number of meta-clusters.
   Usage:
   ncls(immunoMeta)

npar  the number of parameters measured, cell-clustered and meta-clustered
   Usage:
   npar(immunoMeta)

parameters, parameters<->  extracts or replaces the names of measured, cell-clustered and meta-clustered parameters
   Usage:
   parameters(immunoMeta)
   parameters(immunoMeta) <- value

label  the meta-clustering label, that is the assignment of the cell-clusters to the meta-clusters.
   Usage:
   label(immunoMeta, for.sample=NA)
   If for.sample is specified, the label part for this sample only.

weights  the meta-clustering weights for the cluster selection (all meta-cluster by default)
   Usage:
   weights(immunoMets,cls=seq_len(ncls(immunoMeta)))

mu  the meta-cluster mean values for the cluster and parameter selection (all meta-cluster and all parameter by default)
   Usage:
   mu(immunoMeta, cls=seq_len(ncls(immunoMeta)), par=seq_len(npar(immunoMeta)))

sigma  the meta-cluster co-variance values for the cluster and parameter selection (all meta-cluster and all parameter by default)
   Usage:
   sigma(immunoMeta, cls=seq_len(ncls(immunoMeta)), par=seq_len(npar(immunoMeta)))

aposteriori  the a-posteriori probabilities of cluster membership for each cell-cluster
   Usage:
   aposteriori(immunoMeta)

events  the cell-event numbers for each sample for the cluster selection (all meta-cluster by default)
   Usage:
   events(immunoMeta, ncls=seq_len(ncls(immunoMeta)), for.sample=NA)
   If for.sample is specified, the cell-event numbers for this sample only.

prop, prop<->  get or a property value in the hierarchy level given by pos and named name
   Usage:
   prop(immunoMeta, name, pos=c())
   prop(immunoMeta, name, pos, for.level=TRUE, for.sublevels=FALSE) <- value
   If the option for.sublevels is set, the property value will be setted deep for all sub-levels of the by pos specified level.

The prop interface is very basic and no checks for meaningful properties and values are performed. It could be used for everything at any time. Nevertheless, there are some property keys which are used internally mainly to control the plot routine for the levels.
desc the name of this level.
M the mean of all clusters in this level
S the co-variance matrix of all clusters in this level
pscales a list of npar entries for the limits and ticks information.Normally, only set on
root-level and then used for all sub-levels. But could set and altered at any level.
plot.subset an array of parameter indices used as default for the plot of this level.
plot.color an index in the palette or other specified color used for plots of this level in its
parent level.
plot.childs to be renamed in plot.levels.
plot.parent when set, additionally all cluster of the parent level are plotted in light gray.

desc, desc<- Get or set the desc property in the by pos specified level.
Usage:
desc(immunoMeta, pos)
desc(immunoMeta, pos) <- value
descFull Gives the full description path for the level given by pos, i.e. the concatinate desc values
of this all parent levels.
Usage:
descFull(immunoMeta, pos)
level, level<- Get or replace the level object at specified pos,
Usage:
value <- level(immunoMeta, pos)
level(immunoMeta, pos ) <- value
findLevel Find the level pos value for a specific cluster cls
Usage:
pos <- findLevel(immunoMeta, cls)
clusters Retrieves the cluster subset for the level at pos.
Usage:
cls <- clusters(immunoMeta, pos)
classified Retrieves the cluster subset for the level at pos which are classified in sub-levels.
Usage:
cls <- classified(immunoMeta, pos)
unclassified Retrieves the cluster subset for the level at pos which are not classified in sub-levels.
Usage:
cls <- unclassified(immunoMeta, pos)

Manipulators

addLevel<- Adds a level at a specified hierarchy position pos. A level consists of a name (desc)
and a cluster subset cls.
Usage:
addLevel(immunoMeta, pos, desc="new level") <- cls
move<-  Moves a cluster subset to a specific immunoMeta level. Clusters in cls are added to parent levels if nessesary and removed from other levels.

Usage:
move(immunoMeta, pos) <- cls

remove<- removes a cluster subset from a specific immunoMeta level.

Usage:
remove(immunoMeta, pos) <- cls

parent<-  sets the parent for this level, or this level as parent for all its sub-levels

Usage:
parent(immunoMeta, pos) <- c()
parent(immunoMeta, pos) <- level

transfer<-  Overtakes the annotation of an immunoMeta-object to this immunoMeta-object

Usage:
transfer(immunoMeta) <- annotatedMeta

Methods

finalize  After manipulations of a immunoMeta-object finalize restructure all levels and returns the finalized object, where the parent relations are solved and the mean and co-variances of all levels are build.

Usage:
immunoMeta <- finalize(immunoMeta)

subset  Builds the immunoMeta-object for a cluster and/or parameter subset

Usage:
subsetMeta <- subset(immunoMeta, cls=seq_len(ncls(meta)), par=seq_len(npar(meta)))

transformParams  Scales and translates the cluster means of the immunoMeta-object in each parameter

Usage:
transformedMeta <- transformParams(immunoMeta, scale=c(), offset=c())

clusterCoeff  Calculates the bhattacharrya coefficients of clusters cls for a level lvl in the immunoMeta-object

Usage:
ret <- clustersCoeff(immunoMeta, cls, lvl, par=seq_len(npar(immunoMeta)))

clusterDist  Calculates the bhattacharrya distances of clusters cls for a level lvl in the immunoMeta-object

Usage:
ret <- clustersDist(immunoMeta, cls, lvl, par=seq_len(npar(immunoMeta)))

clusterProb  Calculates the bhattacharrya probabilities of clusters cls for a level lvl in the immunoMeta-object

Usage:
ret <- clustersProb(immunoMeta, cls, lvl, par=seq_len(npar(immunoMeta)))
plot.immunoClust

Author(s)
Till Sörensen <till-antoni.soerensen@charite.de>

See Also
immunoMeta-object

Examples

###
data(dat.meta)
npar(dat.meta)
ncls(dat.meta)
cls <- clusters(dat.meta,c(1))
mooth(dat.meta,c(2)) <- cls

---

plot.immunoClust  Scatterplot of immunoClust Clustering Results

Description
This method generates scatterplot revealing the cluster assignment.

Usage

```r
## S4 method for signature 'immunoClust'
plot(x, data, subset=c(1,2), ellipse=T,
     show.rm=F, include=1:(x@K), main=NULL,
     col=include+1, pch=".", cex=0.6,
     col.rm=1, pch.rm=1, cex.rm=0.6, ecol=col, elty=1,
npoints=501, add=F, ...)
```

Arguments

- `x` An object of class `immunoClust` as return by `cell.process`.
- `data` A matrix, data frame of observations, or object of class `flowFrame`. This is the object of observations on which `cell.process` was performed or the matrix of cell-cluster centers for the `meta.process`.
- `subset` A numeric vector of length two indicating which two parameters are selected for the scatterplot. Alternatively, a character vector containing the names of the two parameters is allowed if `x@parameters` is not `NULL`.
- `ellipse` A logical value indicating whether the cluster 90% percentil boundary is to be drawn or not.
- `show.rm` A logical value indicating whether filtered observations will be shown or not.
- `include` A numeric vector specifying which clusters will be shown on the plot. By default, all clusters are included.
main

Title of the plot.

col

Color(s) of the plotting points. May specify a different color for each cluster.

pch

Plotting character(s) of the plotting points. May specify a different character for each cluster.

cex

Size of the plotting characters. May specify a different size for each cluster.

col.rm

Color of the plotting characters denoting filtered observations.

pch.rm

Plotting character used to denote filtered observations.

cex.rm

Size of the plotting character used to denote filtered observations.

ecol

Color(s) of the lines representing the cluster boundaries. May specify a different color for each cluster.

elty

Line type(s) drawing the cluster boundaries. May specify a different line type for each cluster.

npoints

The number of points used to draw each cluster boundary.

add

A logical value. If TRUE, add to the current plot.

... Further graphical parameters passed to the generic function plot.

Value

Plots the clustering assignment on an appropriate plotting device.

Author(s)

Till Sörensen <till-antoni.soerensen@charite.de>

References


See Also

immunoClust-object

Examples

data(dat.fcs)
data(dat.exp)
dat.res <- dat.exp[[1]]
dat.trans <- trans.ApplyToData(dat.res, dat.fcs)
plot(dat.res, dat=dat.trans,N=1000)
Description

This method generates scatterplot revealing the cluster assignment.

Usage

```r
## S3 method for class 'immunoMeta'
plot(x, pos=c(), main="", plot.childs=TRUE,
     plot.unclassified=FALSE, plot.subset=c(), inc.childs=c(), plot.ellipse=TRUE,
     plot.all=FALSE, ...)
```

Arguments

- `x` An object of class `immunoMeta` as return by `meta.process`.
- `pos` gives the position in the immunoMeta-hierarchy to plot (default=c() plots the root level). `pos` is an array of indices, which addresses the level of interest. Each level in the immunoMeta-hierarchy has an array of sub-levels and `pos` is the sequences of indices into these sub-levels.
- `main` additional title which is concatenated with the position and description path of the plotted level.
- `plot.subset` an array of indices for the parameter selection to be plotted.
- `plot.unclassified` if set, the unclassified clusters,i.e clusters not assigned into a sub-level, are plotted rather than the classified clusters.
- `plot.childs` colours the clusters by the sub-level rather than the clusters themselves. By default colours are assigned by sub-level index repeated in red, green,blue,cyan,magenta,yellow,gray,black.
- `inc.childs` optionally, to restrict to a particular selection of sub-levels to plot.
- `plot.ellipse` surrounds the cell-cluster center by an ellipse reflecting the meta-cluster deviation
- `plot.all` plots all sub-levels. Usefull for a full annotation documentation with a pdf file.
- `...` Further graphical parameters passed to the generic function `plot`.

Value

Plots the clustering assignment on an appropriated plotting device.

Author(s)

Till Sørensen <till-antoni.soerensen@charite.de>
**splom.immunoClust**

---

**S4 method for signature 'immunoClust,missing'**

splom(x, data, include=seq_len(x@K), ...)

**S4 method for signature 'immunoClust,flowFrame'**

splom(x, data, include=seq_len(x@K),
      subset=seq_len(length(attributes(x)$param)), N=NULL, label=NULL, desc=NULL,
      add.param=c(), ...)

**S4 method for signature 'immunoClust,matrix'**

splom(x, data, include=seq_len(x@K),
      subset=seq_len(length(attributes(x)$param)), N=NULL, label=NULL,
      desc=NULL, ...)

datSplom(label, data, subset=seq_len(ncol(data)),
         include=seq_len(nrow(data)), ...)

---

**Arguments**

- `x` An object of class `immunoClust` as return by `cell.process` or `meta.process`.
- `data` Missing, a matrix, or object of class `flowFrame`. This is the object of observations on which `cell.process` was performed.
- `include` A numeric vector specifying which clusters will be shown on the plot. By default, all clusters are included.
- `subset` A numeric vector indicating which parameters are selected for the scatterplot matrix.
- `N` An integer for the maximum number of observations to be plotted. By default all observations are plotted.
- `label` A integer vector for the cluster membership of the observations. By default this is `x@label`.

---

**Description**

This method generates scatterplot matrix revealing the cluster assignment.

---

**See Also**

- `immunoMeta-object`

---

**Examples**

```r
data(dat.meta)
plot(dat.meta)
```

---

**Examples**

```r
splom.immunoClust Scatterplot Matrix of immunoClust Clustering Results
```
trans.ApplyToData

| desc     | A character vector for the parameter description. |
| add.param | A list of additional parameters to plot, which are not used for clustering. |
| ...      | Further graphical parameters passed to the generic function splom. |

Value

An object of class trellis as returned by the generic splom function of the lattice-package. The print method (called by default) will plot it on an appropriate plotting device.

Author(s)

Till Sörensen <till-antoni.soerensen@charite.de>

References


See Also

immunoClust-object

Examples

data(dat.fcs)
data(dat.exp)
# cell clustering results of dat.fcs
dat.res <- dat.exp[[1]]
dat.trans <- trans.ApplyToData(dat.res, dat.fcs)
splom(dat.res, data=dat.trans, N=1000)

Description

Applies the transformation information of the immunoClust object to the raw observed FC dataset.

Usage

trans.ApplyToData(x, data, add.param=c(), max.decade=attr(x,"trans.decade"), lin.scale=attr(x,"trans.scale") )
trans.ApplyToData

Arguments

- **x**  
  The immunoClust object containing the estimators for the transformation `trans.a` and `trans.b`.

- **data**  
  The numeric matrix, data frame of observations, or object of class `flowFrame`.

- **add.param**  
  A list of additional parameters in the `flowFrame`, which are not used for clustering but should be included in the final transformed resulting `flowFrame`.

- **max.decade**  
  A numeric scale for the maximum transformed observation value; if missing or below 0, no scaling of the transformed values is applied, which is the default in `immunoClust`.

- **lin.scale**  
  A numeric scaling factor for the linear, i.e. not transformed, parameters; if missing no scaling, i.e. `lin.scale = 1`, is applied, which is the default in `immunoClust`.

Details

In `immunoClust` an \( \text{asinh} \)-transformation \( h(y) = \text{asinh}(a \cdot y + b) \) is applied to the fluorescence parameter in the observed data. The scatter parameter are assumed to be linear.

Value

A matrix or `flowFrame` with replaced transformed observation values.

Author(s)

Till Sörensen <till-antoni.soerensen@charite.de>

References


See Also

`immunoClust`, `trans.FitToData`, `cell.process`

Examples

```r
data(dat.fcs)
data(dat.exp)
dat.trans <- trans.ApplyToData(dat.exp[[1]], dat.fcs)
#
#plot(dat.exp[[1]], data=dat.trans)
#```
trans.FitToData

**immunoClust asinh-Transformation Optimization**

**Description**

Performs variance stabilization transformation estimation on the fluorescence parameters of the observed cell events. It is integrated in the interactive cell event clustering approach of immunoClust when transformation estimation should be applied.

**Usage**

trans.FitToData(x, data, B=10, tol=1e-5, certainty=0.3, proc="vsHtransAw")

**Arguments**

- **x**: The immunoClust object of the fitted mixture model and initial estimators for the transformation.
- **data**: The numeric matrix, data frame of observations, or object of class flowFrame.
- **B**: The maximum number of BFG2 minimizer iterations.
- **tol**: The tolerance used to assess the convergence for the BFG2 minimizer.
- **certainty**: Minimum probability for cluster membership of an observation to be taken into account.
- **proc**: An experimental switch for alternative procedures; should be "vsHtransAw".

**Details**

In immunoClust an asinh-transformation \( h(y) = \text{asinh}(a \cdot y + b) \) is applied for all fluorescence parameter in the observed data.

The transformation optimization trans.FitToData requires a fitted model of cluster information together with suitable initial transformation estimation in an immunoClust object. It fits the transformation based on the initial scaling values trans.a and translation values trans.b to the observed data. It returns the optimized transformation parameter in a \( 2 \times P \)-dimensional matrix, first row for the scaling and second row for the translation values. A scaling value of \( a = 0 \) on input and output indicates, that a parameter should not be transformed.

The presented transformation optimization ("vsHtransAw") fits only the scaling value. An alternative procedure ("vsHtrans_w") fits both, the scaling and the translation value, but turns out to be less robust.

**Value**

Optimized transformation scaling and translation values in a \( 2 \times P \)-dimensional matrix, first row for the scaling and second row for the translation values.

**Author(s)**

Till Sörensen <till-antoni.soerensen@charite.de>
References


See Also

trans.ApplyToData, cell.process

Examples

data(dat.fcs)
data(dat.exp)
## in dat.exp the z-matrices of the immunoClust-object are removed
## so we have to re-calculate it first ...
dat.trans <- trans.ApplyToData(dat.exp[[1]], dat.fcs)
res <- cell.Classify(dat.exp[[1]], dat.trans)
## ... now the transformation parameter can be optimized
trans.FitToData(res, dat.fcs)
Index

* cluster
  cell.ClustData, 4
  cell.EM, 6
  cell.hclust, 9
  cell.ME, 10
  cell.process, 12
  cell.SubClustering, 17
  meta.clustering, 26
  meta.hclust, 31
  meta.ME, 32
  meta.process, 35
  meta.SubClustering, 41

* datasets
  dat.exp, 19
  dat.fcs, 20
  dat.meta, 21

* generics
  generics.immunoclust, 21

* graphs
  plot.immunoClust, 49
  plot.immunoMeta, 51
  splom.immunoClust, 52

* immunoClust
  methods.immunoClust, 43

* immunoMeta
  methods.immunoMeta, 45

* package
  immunoClust-package, 2

* print
  immunoClust-object, 24
  immunoMeta-class, 25

addLevel<-(methods.immunoMeta), 45
aposteriori (generics.immunoclust), 21
aposteriori, immunoClust-method (methods.immunoClust), 43
aposteriori, immunoMeta-method (methods.immunoMeta), 45
aposteriori, immunoClust (methods.immunoClust), 43
aposteriori.immunoMeta (methods.immunoMeta), 45

bhattacharyya, 3

cell.Classify (cell.FitModel), 8
cell.classifyAll (cell.process), 12
cell.ClustData, 4
cell.EM, 6, 9, 11
cell.EMstep (cell.EM), 6
cell.EMt (cell.EM), 6
cell.Estep (cell.EM), 6
cell.Estimation, 9
cell.Estimation (cell.EM), 6
cell.FitModel, 7, 8, 15
cell.hclust, 5, 9, 18
cell.InitialModel (cell.process), 12
cell.MajorIterationLoop (cell.process), 12
cell.MajorIterationTrans (cell.process), 12
cell.ME, 7, 10
cell.MEstep (cell.ME), 10

cell.Mstep (cell.ME), 10
cell.process, 3, 8–10, 12, 18, 24, 25, 35, 37, 43, 49, 52, 54, 56
cell.removed, 16

cell.SubClustering, 15, 17
cell.TestSubCluster, 9, 10
cell.TestSubCluster (cell.SubClustering), 17
cells (methods.immunoClust), 43
cells, immunoClust-method (methods.immunoClust), 43
classified (methods.immunoMeta), 45
classified, immunoMeta-method (methods.immunoMeta), 45
classified.immunoMeta (methods.immunoMeta), 45
clusterCoeff (generics.immunoclust), 21
clusterCoeff,immunoMeta-method  
(methods.immunoMeta), 45
clusterDist (generics.immunoclust), 21
clusterDist,immunoMeta-method  
(methods.immunoMeta), 45
clusterProb (generics.immunoclust), 21
clusterProb,immunoMeta-method  
(methods.immunoMeta), 45
clusters (methods.immunoMeta), 45
clusters,immunoMeta-method  
(methods.immunoMeta), 45
clusters.immunoMeta  
(methods.immunoMeta), 45
dat.exp, 19, 21
dat.fcs, 20
dat.meta, 21
datSplom (splom.immunoClust), 52
desc (methods.immunoMeta), 45
desc,immunoMeta-method  
(methods.immunoMeta), 45
desc.immunoMeta (methods.immunoMeta), 45
desc<-,immunoMeta-method  
(methods.immunoMeta), 45
desc<-.immunoMeta  
(methods.immunoMeta), 45
descFull (methods.immunoMeta), 45
descFull,immunoMeta-method  
(methods.immunoMeta), 45
descFull.immunoMeta  
(methods.immunoMeta), 45
events (methods.immunoMeta), 45
events,immunoClust-method  
(methods.immunoClust), 43
events,immunoMeta-method  
(methods.immunoMeta), 45
events.immunoClust  
(methods.immunoClust), 43
events.immunoMeta (methods.immunoMeta), 45
finalize (methods.immunoMeta), 45
finalize,immunoMeta-method  
(methods.immunoMeta), 45
finalize.immunoMeta  
(methods.immunoMeta), 45
findLevel (methods.immunoMeta), 45
findLevel,immunoMeta-method  
(methods.immunoMeta), 45
generics.immunoclust, 21
immunoClust, 5–8, 10, 11, 15, 17, 18, 23, 28,  
31–33, 43, 49, 52, 54, 55
immunoClust (immunoClust-package), 2
immunoClust-class (immunoClust-object), 24
immunoClust-object, 24, 26, 36
immunoClust-package, 2
immunoClust.object  
(immunoClust-object), 24
immunoClust.package  
(immunoClust-package), 2
immunoMeta (immunoMeta-class), 25
immunoMeta-class, 25
immunoMeta-object (immunoMeta-class), 25
immunoMeta.class (immunoMeta-class), 25
label (generics.immunoclust), 21
label,immunoClust-method  
(methods.immunoClust), 43
label,immunoMeta-method  
(methods.immunoMeta), 45
label.immunoClust  
(methods.immunoClust), 43
label.immunoMeta (methods.immunoMeta), 45
level (methods.immunoMeta), 45
level,immunoMeta-method  
(methods.immunoMeta), 45
level<-,immunoMeta-method  
(methods.immunoMeta), 45
meta.Clustering, 33, 35, 37, 39, 41, 42
meta.Clustering (meta.clustering), 26
meta.clustering, 26
meta.export, 28, 37
meta.exprs, 30
meta.freqTable (meta.export), 28
meta.hclust, 31, 42
meta.majorEvents (meta.export), 28
INDEX

meta.ME, 32
meta.Normalize, 27
meta.Normalize (meta.normalize), 34
meta.normalize, 34
meta.numClusters (meta.export), 28
meta.numEvents (meta.export), 28
meta.parMFI (meta.export), 28
meta.process, 3, 21, 24–26, 28, 29, 32, 33, 35, 35, 42, 45, 51, 52
meta.RegNorm (meta.regnorm), 37
meta.regnorm, 37
meta.relEvents (meta.export), 28
meta.relEvents2 (meta.export), 28
meta.relEvents3 (meta.export), 28
meta.relParent (meta.export), 28
meta.SON.cluster, 38
meta.SON.combineClustering, 39
meta.SubClustering, 28, 41
meta.TestSubCluster, 31, 32
meta.TestSubCluster
(meta.SubClustering), 41
methods.immunoclust, 43
methods.immunoMeta, 45
move<- (methods.immunoMeta), 45
mu (generics.immunoclust), 21
mu,immunoClust-method
(methods.immunoClust), 43
mu,immunoMeta-method
(methods.immunoMeta), 45
mu.immunoClust (methods.immunoClust), 43
mu.immunoMeta (methods.immunoMeta), 45
ncls (generics.immunoclust), 21
ncls,immunoClust-method
(methods.immunoClust), 43
ncls,immunoMeta-method
(methods.immunoMeta), 45
ncls.immunoClust (methods.immunoClust), 43
ncls.immunoMeta (methods.immunoMeta), 45
nobs (generics.immunoclust), 21
nobs,immunoClust-method
(methods.immunoClust), 43
nobs,immunoMeta-method
(methods.immunoMeta), 45
nobs.immunoClust (methods.immunoClust), 43
nobs.immunoMeta (methods.immunoMeta), 45
npar (generics.immunoclust), 21
npar,immunoClust-method
(methods.immunoClust), 43
npar,immunoMeta-method
(methods.immunoMeta), 45
npar.immunoClust (methods.immunoClust), 43
npar.immunoMeta (methods.immunoMeta), 45
nsam (generics.immunoclust), 21
nsam,immunoMeta-method
(methods.immunoMeta), 45
nsam.immunoMeta (methods.immunoMeta), 45
parameters (generics.immunoclust), 21
parameters,immunoClust-method
(methods.immunoClust), 43
parameters,immunoMeta-method
(methods.immunoMeta), 45
parameters.immunoClust
(methods.immunoMeta), 43
parameters.immunoMeta
(methods.immunoMeta), 45
parameters<-(generics.immunoclust), 21
parameters<-,immunoClust,character-method
(methods.immunoClust), 43
parameters<-,immunoMeta,character-method
(methods.immunoMeta), 45
parameters<-.immunoClust
(methods.immunoMeta), 43
parameters<-.immunoMeta
(methods.immunoMeta), 45
parent<- (methods.immunoMeta), 45
plot, 15
plot,immunoClust,missing-method
(plot.immunoClust), 49
plot,immunoClust-method
(plot.immunoClust), 49
plot,immunoMeta,missing-method
(plot.immunoMeta), 51
plot,immunoMeta-method
(plot.immunoMeta), 51
plot.immunoClust, 49
plot.immunoMeta, 51
prop (methods.immunoMeta), 45
prop,immunoMeta-method
(methods.immunoMeta), 45
prop.immunoMeta (methods.immunoMeta), 45
prop<- (methods.immunoMeta), 45
prop<-,immunoMeta-method
(methods.immunoMeta), 45
prop<-.immunoMeta (methods.immunoMeta), 45
remove<-(methods.immunoMeta), 45
removed.above (cell.removed), 16
removed.below (cell.removed), 16

sam.clsEvents (generics.immunoclust), 21
sam.clsEvents,immunoMeta-method
(methods.immunoMeta), 45

sam.clsMu (generics.immunoclust), 21
sam.clsMu,immunoMeta-method
(methods.immunoMeta), 45

sam.clsSigma (generics.immunoclust), 21
sam.clsSigma,immunoMeta-method
(methods.immunoMeta), 45

sam.clsWeights (generics.immunoclust), 21
sam.clsWeights,immunoMeta-method
(methods.immunoMeta), 45

sam.ncls (generics.immunoclust), 21
sam.ncls,immunoMeta-method
(methods.immunoMeta), 45

sigma (generics.immunoclust), 21
sigma,immunoClust-method
(methods.immunoClust), 43
sigma,immunoMeta-method
(methods.immunoMeta), 45
sigma.immunoClust
(methods.immunoClust), 43

sigma.immunoMeta (methods.immunoMeta), 45
splom, 15
splom (splom.immunoClust), 52
splom.immunoClust,flowFrame-method
(splom.immunoClust), 52
splom.immunoClust,matrix-method
(splom.immunoClust), 52
splom.immunoClust,missing-method
(splom.immunoClust), 52

subset (generics.immunoclust), 21
subset.immunoClust-method
(methods.immunoClust), 43
subset.immunoMeta-method
(methods.immunoMeta), 45

trans.ApplyToData, 53, 56
trans.FitToData, 15, 54, 55
transfer<-(methods.immunoMeta), 45
transfer<-.immunoMeta,immunoMeta-method
(methods.immunoMeta), 45

transformParams (generics.immunoclust), 21
transformParams,immunoClust-method
(methods.immunoClust), 43
transformParams,immunoMeta-method
(methods.immunoMeta), 45
transformParams.immunoClust
(methods.immunoClust), 43
transformParams.immunoMeta
(methods.immunoMeta), 45

trans.immunoMeta (methods.immunoMeta), 45

transformParams.immunoClust
(methods.immunoClust), 43
unclassified.immunoMeta-method (methods.immunoMeta), 45
unclassified.immunoMeta (methods.immunoMeta), 45
weights(generics.immunoclust), 21
weights.immunoClust-method (methods.immunoClust), 43
weights.immunoMeta-method (methods.immunoMeta), 43
weights.immunoClust (methods.immunoClust), 43
weights.immunoMeta (methods.immunoMeta), 45