

# Package ‘flowBin’

April 17, 2025

**Type** Package

**Title** Combining multitube flow cytometry data by binning

**Version** 1.45.0

**Date** 2013-5-11

**Author** Kieran O'Neill

**Maintainer** Kieran O'Neill <koneill@bccrc.ca>

**Description** Software to combine flow cytometry data that has been multiplexed into multiple tubes with common markers between them, by establishing common bins across tubes in terms of the common markers, then determining expression within each tube for each bin in terms of the tube-specific markers.

**License** Artistic-2.0

**LazyLoad** yes

**Imports** class, limma, snow, BiocGenerics

**Depends** methods, flowCore, flowFP, R (>= 2.10)

**Suggests** parallel

**biocViews** ImmunoOncology, CellBasedAssays, FlowCytometry

**Collate** 'AllClasses.R' 'AllGenerics.R' 'BinnedFlowSample-accessors.R'  
'checkQNorm.R' 'flowBin.R' 'flowFPBin.R'  
'FlowSample-accessors.R' 'getBinExpr.R' 'kMeansBin.R'  
'mapClustersKNN.R' 'quantileNormalise.R' 'removeSparseBins.R'

**git\_url** <https://git.bioconductor.org/packages/flowBin>

**git\_branch** devel

**git\_last\_commit** f833cbb

**git\_last\_commit\_date** 2025-04-15

**Repository** Bioconductor 3.22

**Date/Publication** 2025-04-17

## Contents

amlsample . . . . .	2
BinnedFlowSample . . . . .	2
checkQNorm . . . . .	3

eventsInBins . . . . .	3
flowBin . . . . .	3
flowFPBin . . . . .	5
FlowSample . . . . .	5
getBinExpr . . . . .	6
kMeansBin . . . . .	6
mapBinsKNN . . . . .	7
quantileNormalise . . . . .	8
removeSparseBins . . . . .	8
show-methods . . . . .	9

<b>Index</b>	<b>10</b>
--------------	-----------

---

amlsample	<i>Multitube AML sample as example data for flowBin</i>
-----------	---

---

### Description

Multitube AML sample as example data for flowBin

### Format

a flowSample containing 7 tubes with 3 common parameters and 4 measure parameters per tube.

### Source

FlowRepository.org accession FR-FCM-ZZYA

---

BinnedFlowSample	<i>A FlowSample, but with binning information for each tube</i>
------------------	---

---

### Description

**clust.labels:** list of cluster label vectors, one for each tube  
 Note: all slots can be get and set using accessor methods, for example `bin.pars(myFlowSet) <- c(1,2,5)`

---

checkQNorm	<i>Function to check the quantile normalisation of a FlowSample using flowFP</i>
------------	--

---

### Description

Function to check the quantile normalisation of a FlowSample using flowFP

### Details

object and normed.object are compared using flowFP binning, to assess the deviation in bin counts between the two.

### Value

list containing two matrices of standard deviations across bins (rows) vs tubes (columns) for before (sd.before) and after (sd.after).

### Examples

```
data(aml.sample)
normed.sample <- quantileNormalise(aml.sample)
qnorm.check <- checkQNorm(aml.sample, normed.sample, do.plot=FALSE)
show(qnorm.check)
```

---

eventsInBins	<i>Count number of events for each tube in each bin</i>
--------------	---

---

### Description

Useful for QA of bin mapping

---

flowBin	<i>function to run the entire flowBin pipeline</i>
---------	--

---

### Description

Takes a list of flowFrames representing tubes from a single flow cytometry sample, and combines them using binning of events in terms of common markers across tubes.

### Usage

```
flowBin(tube.list, bin.pars, control.tubes = vector(),
  measure.pars = NULL,
  sample.name = "Unnamed Flow Expr Set",
  bin.method = "kmeans", expr.method = "medianFI",
  sparse.bin.thresh = 0.001, dequantize = T,
  snow.cluster = NULL, n.bins = 128, scale.expr = F,
  do.qnorm = T, return.bins = F)
```

**Arguments**

<code>tube.list</code>	a list of flowFrames, one for each tube to combine
<code>bin.pars</code>	a numerical vector indicating which flow parameters in the each flowFrame to use for combining tubes. These should be the same markers assayed across every tube.
<code>control.tubes</code>	a vector indicating which tubes in tube.list to use for negative controls. May be empty.
<code>measure.pars</code>	a list of which parameters to measure expression for, with one vector for each tube. If left NULL, this defaults to all parameters other than those specified as bin.pars
<code>sample.name</code>	name of this flowSample, for convenience (defaults to 'Unnamed Flow Expr Set')
<code>bin.method</code>	The method to use for creating bins. The two options are "kmeans" for k-means clustering and nearest-neighbour mapping of bins. or "flowFP" for flowFP binning and direct mapping of bin boundaries across tubes.
<code>expr.method</code>	The method to use to compute bin expression across tubes. This defaults to MFI of the cells belonging to that bin in each tube. Other options are
<code>sparse.bin.thresh</code>	Bins which contain fewer than this proportion of total events in any tube will be excluded as outliers. Defaults to 0.001
<code>dequantize</code>	If TRUE, adds a small (region of 1e-8) value to flow data to help break ties when binning.
<code>snow.cluster</code>	A cluster created using the snow package, which flowBin will use to speed up computation. If NULL, flowBin will execute in serial mode.
<code>n.bins</code>	Number of bins to use. Note that this must be a power of 2 if flowFP is selected as binning method.
<code>scale.expr</code>	If TRUE, the resulting expression values will be scaled to (0,1) using the ranges specified in the flowFrames in tube.list.
<code>do.qnorm</code>	If TRUE, the binning markers will be quantile normalized prior to binning.
<code>return.bins</code>	If TRUE, return a BinnedFlowExprSet containing the bins themselves as well as the expression for each bin.

**Value**

A matrix containing expression values for each bin in terms of each marker across all tubes. If `return.bins` is set TRUE, then a list containing a BinnedFlowExprSet followed by the expression matrix is returned.

**Examples**

```
data(amlsample)
tube.combined <- flowBin(aml.sample@tube.set,
  bin.pars=aml.sample@bin.pars,
  bin.method='flowFP',
  control.tubes=aml.sample@control.tubes,
  expr.method='medianFIDist', scale.expr=TRUE)
heatmap(tube.combined, scale='none')
```

flowFPBin

*Bin sample using flowFP binning***Description**

Bin sample using flowFP binning

**Arguments**

`object`                `flowSample` to bin

`n.bins=128`            number of bins to use. This should be a power of 2, and will be rounded down to the nearest power of 2 if not.

`snow.cluster=NULL`    Optional snow cluster to use for parallel execution.

`dequantize=T`          If TRUE, adds a small (region of 1e-8) value to flow data to help break ties when binning.

**Value**

a BinnedFlowSample

**Examples**

```
data(amlsample)
normed.sample <- quantileNormalise(aml.sample)
res <- flowFPBin(normed.sample)
```

FlowSample

*A class similar to flowSet, but with extra information needed by flow-Bin***Description**

`name`: character string - name of the object

`tube.set`: list of `flowFrames` containing raw flow data.

`control.tubes`: Integer vector indicating which tubes in the list (if any) to use as negative controls. May be empty.

`bin.pars`: Integer vector indicating which parameters to use for binning. These must be in the same position in all tubes.

`measure.pars`: list of integer vectors indicating which parameters to use for measurement. These must be specified per tube. Note: all slots can be get and set using accessor methods, for example `bin.pars(myFlowSet) <- c(1,2,5)`

---

getBinExpr	<i>Calculate the expression of each bin in a BinnedFlowSample in terms of the measurement markers</i>
------------	---

---

## Description

getBinExpr main function definition

## Arguments

method	Method to use to compute expression, passed as a string. Defaults to medianFI, which takes the simple median of each bin, and does not require control tubes. Other options available are medianFIDist, which uses medians with the median of the negative control subtracted out, and propPos which sets a threshold at the 98th percentile of the negative control and determines what proportion of cells lie above that.
include.bin.medians	logical, specifies whether to compute the medians of each bin in terms of the binning markers and include them in the result or not. Defaults to T.
scale	logical specifying whether to scale the results to the interval (0,1). If T (default), then all medians will be divided by the range for that marker as specified in the flowFrame.

## Value

A numeric matrix containing expression values, with bins as rows and markers as columns

## Examples

```
data(aml.sample)
normed.sample <- quantileNormalise(aml.sample)
binned.sample <- flowFPBin(normed.sample)
binned.sample <- removeSparseBins(binned.sample, 0.001)
bin.expr <- getBinExpr(binned.sample)
heatmap(bin.expr, scale='none')
```

---

kMeansBin	<i>Bin sample using K-means binning</i>
-----------	---

---

## Description

Bin sample using K-means binning

**Arguments**

object	flowSample to bin
n.bins=128	number of bins to use. This should be a power of 2, and will be rounded down to the nearest power of 2 if not.
n.neighbours=1	number of neighbours to use for KNN mapping of bins from clustered tube
snow.cluster=NULL	Optional snow cluster to use for parallel execution.
random.seed=101	Random seed to set to make K-means clustering deterministic.
dequantize=T	If TRUE, adds a small (region of 1e-8) value to flow data to help break ties when binning.

**Details**

Runs K-means clustering on the binning markers in the first tube of the data set. These clusters are then mapped to the other tubes using K-nearest neighbours.

**Value**

a BinnedFlowSample

**Examples**

```
data(aml.sample)
normed.sample <- quantileNormalise(aml.sample)
res <- kMeansBin(normed.sample)
```

---

mapBinsKNN

*Internal function to map bins by KNN*


---

**Description**

Internal function to map bins by KNN

**Arguments**

object	flowSample to map the bins of
tube.1.labels	integer vector of bin labels for the events in tube 1
n.neighbours=1	number of neighbours to use for KNN mapping of bins from clustered tube
snow.cluster=NULL	Optional snow cluster to use for parallel execution.
dequant=T	If TRUE, adds a small (region of 1e-8) value to flow data to help break ties when binning.

**Details**

Takes a FlowSample and labels for the events in tube 1, and maps these to all other tubes.

**Value**

a BinnedFlowSample

**Examples**

```
data(aml.sample)
tube1.expr <- exprs(tube.set(aml.sample)[[1]])
kmeans.res <- kmeans(tube1.expr, 100)
kmeans.labels <- kmeans.res$cluster

#Now create a binnedFlowExprSet using the cluster labels for tube 1
clustered.sample <- mapBinsKNN(aml.sample, kmeans.labels)
sort(table(bin.labels(clustered.sample)[[3]]))
```

---

quantileNormalise	<i>quantileNormalise normalise binning parameters across all tubes of a flowSample</i>
-------------------	--

---

**Description**

Since the binning parameters are the same across tubes, and samples each tube is an aliquot from the same sample, these should have the same underlying distribution. Hence, quantile normalisation can be used to force this to be so, removing technical variation.

**Examples**

```
data(aml.sample)
normed.sample <- quantileNormalise(aml.sample)
qnorm.check <- checkQNorm(aml.sample, normed.sample, do.plot=FALSE)
show(qnorm.check)
```

---

removeSparseBins	<i>Remove bins from a BinnedFlowSample with few events in them</i>
------------------	--

---

**Description**

Remove bins from a BinnedFlowSample with few events in them

**Arguments**

object	the BinnedFlowSample to act on
cutoff.prop=NULL	the minimum proportion that a bin must contain to be kept. If NULL, only bins with no events in at least one tube will be removed.

**Details**

This is important to do prior to calculating bin expression, as bins containing 2 or less events, for example, cannot have their median computed.

**Value**

a BinnedFlowSample with sparse bins removed



---

show-methods*Methods to view flowBin objects*

---

**Description**

Methods for function show in Package flowBin

**Methods**

signature(object = "BinnedFlowExprSet") Show number of bins and samples for a Binned-FlowExprSet.

signature(object = "CVResult") Show various statistics stored in a CVResult object.

# Index

- \* **datasets**
  - amlsample, 2
- \* **methods**
  - show-methods, 9
- aml.sample (amlsample), 2
- amlsample, 2
- bin.labels (BinnedFlowSample), 2
- bin.labels, BinnedFlowSample-method (BinnedFlowSample), 2
- bin.labels<- (BinnedFlowSample), 2
- bin.labels<- , BinnedFlowSample, list-method (BinnedFlowSample), 2
- bin.pars (FlowSample), 5
- bin.pars, FlowSample-method (FlowSample), 5
- bin.pars<- (FlowSample), 5
- bin.pars<- , FlowSample, vector-method (FlowSample), 5
- BinnedFlowSample, 2
- BinnedFlowSample-class (BinnedFlowSample), 2
- checkQNorm, 3
- checkQNorm, (checkQNorm), 3
- checkQNorm, FlowSample, FlowSample-method (checkQNorm), 3
- control.tubes (FlowSample), 5
- control.tubes, FlowSample-method (FlowSample), 5
- control.tubes<- (FlowSample), 5
- control.tubes<- , FlowSample, vector-method (FlowSample), 5
- eventsInBins, 3
- eventsInBins, BinnedFlowSample-method (eventsInBins), 3
- flowBin, 3
- flowbin-package (flowBin), 3
- flowFPBin, 5
- flowFPBin, FlowSample-method (flowFPBin), 5
- FlowSample, 5
- FlowSample-class (FlowSample), 5
- getBinExpr, 6
- getBinExpr, BinnedFlowSample-method (getBinExpr), 6
- kMeansBin, 6
- kMeansBin, FlowSample-method (kMeansBin), 6
- mapBinsKNN, 7
- mapBinsKNN, FlowSample-method (mapBinsKNN), 7
- measure.pars (FlowSample), 5
- measure.pars, FlowSample-method (FlowSample), 5
- measure.pars<- (FlowSample), 5
- measure.pars<- , FlowSample, list-method (FlowSample), 5
- name (FlowSample), 5
- name, FlowSample-method (FlowSample), 5
- name<- (FlowSample), 5
- name<- , FlowSample, character-method (FlowSample), 5
- quantileNormalise, 8
- quantileNormalise, FlowSample-method (quantileNormalise), 8
- removeSparseBins, 8
- removeSparseBins, BinnedFlowSample-method (removeSparseBins), 8
- show, BinnedFlowExprSet-method (show-methods), 9
- show, CVResult-method (show-methods), 9
- show, FlowSample-method (FlowSample), 5
- show-methods, 9
- tube.set (FlowSample), 5
- tube.set, FlowSample-method (FlowSample), 5
- tube.set<- (FlowSample), 5
- tube.set<- , FlowSample, list-method (FlowSample), 5