

# Package ‘funtooNorm’

September 30, 2025

**Type** Package

**Title** Normalization Procedure for Infinium HumanMethylation450  
BeadChip Kit

**Version** 1.33.0

**Date** 2017-09-19

**Author** Celia Greenwood <celia.greenwood@mcgill.ca>, Stepan Grinek  
<stepan.grinek@ladydavis.ca>, Maxime Turgeon <maxime.turgeon@mail.mcgill.ca>,  
Kathleen Klein <kathleen.klein@mail.mcgill.ca>

**Maintainer** Kathleen Klein <kathleen.klein@mail.mcgill.ca>

**Description** Provides a function to normalize Illumina Infinium Human  
Methylation 450 BeadChip (Illumina 450K), correcting for tissue and/or cell  
type.

**License** GPL-3

**Imports** pls, matrixStats, minfi, methods,  
IlluminaHumanMethylation450kmanifest,  
IlluminaHumanMethylation450kanno.ilmn12.hg19, GenomeInfoDb,  
grDevices, graphics, stats

**Suggests** prettydoc, minfiData, knitr, rmarkdown

**Depends** R(>= 3.4)

**LazyData** true

**VignetteBuilder** knitr

**biocViews** DNAMethylation, Preprocessing, Normalization

**RoxygenNote** 6.0.1

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

**git\_branch** devel

**git\_last\_commit** e1d5655

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

**Repository** Bioconductor 3.22

**Date/Publication** 2025-09-29

Contents

|                                 |           |
|---------------------------------|-----------|
| funtooNorm-package . . . . .    | 2         |
| agreement . . . . .             | 3         |
| fromGenStudFiles . . . . .      | 3         |
| fromRGChannelSet . . . . .      | 4         |
| funtooNorm . . . . .            | 4         |
| getGRanges . . . . .            | 5         |
| getNormBeta . . . . .           | 6         |
| getNormM . . . . .              | 7         |
| getRawBeta . . . . .            | 7         |
| getSnpM . . . . .               | 8         |
| plotValidationGraph . . . . .   | 9         |
| SampleSet-class . . . . .       | 9         |
| show,SampleSet-method . . . . . | 10        |
| <b>Index</b>                    | <b>11</b> |

---

|                    |                   |
|--------------------|-------------------|
| funtooNorm-package | <i>funtooNorm</i> |
|--------------------|-------------------|

---

Description

The funtooNorm Package provides a normalization method for data arising from the Illumina Infinium Human Methylation 450 BeadChip (Illumina 450K), including explicit considerations of differences between tissues or cell types. This method should only be used when the data set contains samples fom multiple different tissues or cell types.

Details

Package: funtooNorm  
Type: Package  
License: GPL-3

Author(s)

Celia Greenwood, Stepan Grinek, Raphael Poujol, Maxime Turgeon, Kathleen Oros Klein

---

|           |                                                                            |
|-----------|----------------------------------------------------------------------------|
| agreement | <i>Function to measure intra-replicate agreement for methylation data.</i> |
|-----------|----------------------------------------------------------------------------|

---

**Description**

Function to measure intra-replicate agreement for methylation data.

**Usage**

```
agreement(Beta, individualID)
```

**Arguments**

**Beta** : Matrix with beta-values, rows corresponding to probes, columns corresponding to samples.

**individualID** : a vector where 2 replicates have the exact same value for two technical replicates. Order of samples should nmatch the samples (columns) in Beta

**Details**

We expect that the values returned by the agreement function after normalization by funtooNorm to be smaller than before.

**Value**

The average value of the square distance between replicates: a measure of agreement between replicates in methylation data.

**Examples**

```
agreement(cbind(rnorm(n = 10),rnorm(n = 10),rnorm(n = 10)),c(1,1,1))
```

---

|                  |                                                                         |
|------------------|-------------------------------------------------------------------------|
| fromGenStudFiles | <i>Creates a S4 object of class 'SampleSet' from GenomeStudio files</i> |
|------------------|-------------------------------------------------------------------------|

---

**Description**

Creates a S4 object of class 'SampleSet' from GenomeStudio files

**Usage**

```
fromGenStudFiles(controlProbeFile, signalFile, cell_type)
```

**Arguments**

**controlProbeFile** : The control probe file exported from GenomeStudio

**signalFile** : The signals exported from GenomeStudio samples must be in same order as the control probe File

**cell\_type** : A vector of cell types, names must match control probes and signal files.

**Value**

An object of class 'SampleSet'.

---

|                  |                                                                       |
|------------------|-----------------------------------------------------------------------|
| fromRGChannelSet | <i>Creates an object of class SampleSet from a RGChannelSet minfi</i> |
|------------------|-----------------------------------------------------------------------|

---

**Description**

Creates a object of class SampleSet from the raw unprocessed data in RGChannelSet

**Usage**

```
fromRGChannelSet(myRGChannelSet)
```

**Arguments**

myRGChannelSet : RGChannelSet, from minfi package, should contain a cell\_type vector in pData

**Value**

An object of class 'SampleSet'

**Examples**

```
require(minfiData)
pData(RGsetEx)$cell_type <- rep(c("type1","type2"),3)
mySampleSet=fromRGChannelSet(RGsetEx)
```

---

|            |                                              |
|------------|----------------------------------------------|
| funtooNorm | <i>The funtooNorm normalization function</i> |
|------------|----------------------------------------------|

---

**Description**

funtooNorm Returns the normalized signals to the SampleSet object

**Usage**

```
funtooNorm(object, type.fits = "PCR", ncmp = 4, force = FALSE,
  sex = NULL)

## S4 method for signature 'SampleSet'
funtooNorm(object, type.fits = "PCR", ncmp = 4,
  force = FALSE, sex = NULL)
```

**Arguments**

|           |                                                                                    |
|-----------|------------------------------------------------------------------------------------|
| object    | Object of class SampleSet                                                          |
| type.fits | Choice between "PCR" or "PLS" (default="PCR")                                      |
| ncmp      | Number of components included in the analysis (default=4)                          |
| force     | If set to TRUE, forces the normalization procedure to re-compute                   |
| sex       | Boolean vector if male. if NULL Beta values from ChrY are used for classification. |

**Details**

This is a generic function which applies to autosomes and the X chromosome. Chromosome Y requires separate analysis as there are few probes on Y. We use a straightforward quantile normalization applied to males only.

**Value**

a S4 object of class SampleSet containing the normalized signal

**Methods (by class)**

- SampleSet: The funtooNorm normalization function

**Examples**

```
require(minfiData)
pData(RGsetEx)$cell_type <- rep(c("type1", "type2"), 3)
mySampleSet=fromRGChannelSet(RGsetEx)
mySampleSet=funtooNorm(mySampleSet)
```

---

getGRanges

*Build GRange object of methylation probes*


---

**Description**

Build GRange object of methylation probes

**Usage**

```
getGRanges(object)

## S4 method for signature 'SampleSet'
getGRanges(object)
```

**Arguments**

|        |                            |
|--------|----------------------------|
| object | Object of class SampleSet. |
|--------|----------------------------|

**Value**

A GRange object of the positions of each cpg.

**Methods (by class)**

- SampleSet: Build GRange object of methylation probes

**Examples**

```
require(minfiData)
pData(RGsetEx)$cell_type <- rep(c("type1", "type2"), 3)
mySampleSet=fromRGChannelSet(RGsetEx)
gr=getGRanges(mySampleSet)
```

---

|             |                                                     |
|-------------|-----------------------------------------------------|
| getNormBeta | <i>Computes Beta values from normalized signals</i> |
|-------------|-----------------------------------------------------|

---

**Description**

Computes Beta values from normalized signals

**Usage**

```
getNormBeta(object, offset = 100)

## S4 method for signature 'SampleSet'
getNormBeta(object, offset = 100)
```

**Arguments**

|        |                                     |
|--------|-------------------------------------|
| object | of type SampleSet                   |
| offset | default is 100 as Illumina standard |

**Value**

a matrix containing beta after normalization value for each CpG position and each samples

**Methods (by class)**

- SampleSet: Computes Beta values from normalized signals

**Examples**

```
require(minfiData)
pData(RGsetEx)$cell_type <- rep(c("type1", "type2"), 3)
mySampleSet=fromRGChannelSet(RGsetEx)
b=getNormBeta(funtooNorm(mySampleSet))
```

---

|          |                                                                      |
|----------|----------------------------------------------------------------------|
| getNormM | <i>Computes M values, log2(Meth/Unmeth), from normalized signals</i> |
|----------|----------------------------------------------------------------------|

---

**Description**

Computes M values, log2(Meth/Unmeth), from normalized signals

**Usage**

```
getNormM(object)

## S4 method for signature 'SampleSet'
getNormM(object)
```

**Arguments**

object            An object of class SampleSet

**Value**

a matrix containing M values, log2(Meth/Unmeth), after normalization

**Methods (by class)**

- SampleSet: Computes M values, log2(Meth/Unmeth), from normalized signals

**Examples**

```
require(minfiData)
pData(RGsetEx)$cell_type <- rep(c("type1", "type2"), 3)
mySampleSet=fromRGChannelSet(RGsetEx)
m=getNormM(funtooNorm(mySampleSet))
```

---

|            |                                             |
|------------|---------------------------------------------|
| getRawBeta | <i>Computes Beta value from raw signals</i> |
|------------|---------------------------------------------|

---

**Description**

Computes Beta value from raw signals

**Usage**

```
getRawBeta(object, offset = 100)

## S4 method for signature 'SampleSet'
getRawBeta(object, offset = 100)
```

**Arguments**

|        |                                     |
|--------|-------------------------------------|
| object | object of class SampleSet           |
| offset | default is 100 as Illumina standard |

**Value**

a matrix containing the raw beta value for each position and each samples

**Methods (by class)**

- SampleSet: Computes Beta value from raw signals

**Examples**

```
require(minfiData)
pData(RGsetEx)$cell_type <- rep(c("type1", "type2"), 3)
mySampleSet=fromRGChannelSet(RGsetEx)
r=getRawBeta(mySampleSet)
```

---

|         |                                                           |
|---------|-----------------------------------------------------------|
| getSnpM | <i>Computes M values after normalization of SNP data.</i> |
|---------|-----------------------------------------------------------|

---

**Description**

Computes M values after normalization of SNP data.

**Usage**

```
getSnpM(object)

## S4 method for signature 'SampleSet'
getSnpM(object)
```

**Arguments**

|        |                    |
|--------|--------------------|
| object | of class SampleSet |
|--------|--------------------|

**Value**

a matrix containing M values,  $\log_2(\text{Meth}/\text{Unmeth})$ , after normalization for SNP data

**Methods (by class)**

- SampleSet: Computes M values,  $\log_2(\text{Meth}/\text{Unmeth})$ , for normalized SNP data

**Examples**

```
require(minfiData)
pData(RGsetEx)$cell_type <- rep(c("type1", "type2"), 3)
mySampleSet=fromRGChannelSet(RGsetEx)
snp=getSnpM(funtooNorm(mySampleSet))
```



---

|                     |                                                                      |
|---------------------|----------------------------------------------------------------------|
| plotValidationGraph | <i>plot of Validation Graph for determining number of components</i> |
|---------------------|----------------------------------------------------------------------|

---

### Description

Plots a series of graphs for each signal type, to determine the number of components to include in the normalization procedure.

### Usage

```
plotValidationGraph(object, type.fits = "PCR", pdf.file = NULL)
```

```
## S4 method for signature 'SampleSet'
plotValidationGraph(object, type.fits = "PCR",
  pdf.file = NULL)
```

### Arguments

|           |                                                                                          |
|-----------|------------------------------------------------------------------------------------------|
| object    | of class SampleSet                                                                       |
| type.fits | can be "PCR" or "PLS" (default "PCR")                                                    |
| pdf.file  | if no file name is provided print pdf file plotValidationGraph.pdf in working directory. |

### Value

No value is returned. The function prints the plots to a pdf file.

### Methods (by class)

- SampleSet: Plots a series of graphs for each signal type, to determine the number of components to include in the normalization procedure.

### Examples

```
require(minfiData)
pData(RGsetEx)$cell_type <- rep(c("type1", "type2"), 3)
mySampleSet=fromRGChannelSet(RGsetEx)
plotValidationGraph(mySampleSet)
```

---

|                 |                                  |
|-----------------|----------------------------------|
| SampleSet-class | <i>S4 class object SampleSet</i> |
|-----------------|----------------------------------|

---

### Description

SampleSet is an S4 class defined for the purpose of running the funtooNorm algorithm. They are lists containing signal data and different variables useful for funtooNorm. The data is separated into the 3 probes types, each having 2 channels (methylated and unmethylated ie : A and B) We then define then the 6 (2\*3) labels: AIGrn BIGrn AIGrn BIRed AII BII

**Value**

a S4 object of class SampleSet

**Slots**

type Character: is 'minfi' or 'GenomeStudio'  
 sampleNames character vector: contain the list of sample names in order used  
 sampleSize numeric: the number of samples  
 nPos numeric: the number of positions in the ILLUMINA chip  
 annotation character: the annotation object from minfi package  
 cell\_type factor: vector of the cell type for each sample as factors  
 qntllist numeric: vector of ordered quantiles  
 quantiles list: list of 6 quantiles tables for the 6 signal types  
 ctl.covmat matrix: covariance matrix for the model fit  
 signal list: list of the values for all 6 probe types.  
 names list: list of probes for each type  
 predmat list: list of the normalized values for all 6 probe types.

**Examples**

```
showClass("SampleSet")
```

---

show, SampleSet-method    *Show Object SampleSet*

---

**Description**

Display informations about the SampleSet object

**Usage**

```
## S4 method for signature 'SampleSet'
show(object)
```

**Arguments**

object                    an object of class SampleSet  
 ...                        optional arguments passed to or from other methods.

**Value**

No value is returned. The function prints the summary of object of class SampleSet to screen

**Examples**

```
require(minfiData)
pData(RGsetEx)$cell_type <- rep(c("type1", "type2"), 3)
mySampleSet=fromRGChannelSet(RGsetEx)
mySampleSet
```

# Index

## \* Methylation, Preprocessing, PLS

funtooNorm-package, [2](#)

agreement, [3](#)

fromGenStudFiles, [3](#)

fromRGChannelSet, [4](#)

funtooNorm, [4](#)

funtooNorm, SampleSet-method  
(funtooNorm), [4](#)

funtooNorm-package, [2](#)

getGRanges, [5](#)

getGRanges, SampleSet-method  
(getGRanges), [5](#)

getNormBeta, [6](#)

getNormBeta, SampleSet-method  
(getNormBeta), [6](#)

getNormM, [7](#)

getNormM, SampleSet-method (getNormM), [7](#)

getRawBeta, [7](#)

getRawBeta, SampleSet-method  
(getRawBeta), [7](#)

getSnpM, [8](#)

getSnpM, SampleSet-method (getSnpM), [8](#)

plotValidationGraph, [9](#)

plotValidationGraph, SampleSet-method  
(plotValidationGraph), [9](#)

SampleSet-class, [9](#)

show, SampleSet-method, [10](#)