

Package ‘convert’

February 20, 2025

Version 1.83.0

Title Convert Microarray Data Objects

Author Gordon Smyth <smyth@wehi.edu.au>,
James Wettenhall <wettenhall@wehi.edu.au>,
Yee Hwa (Jean Yang) <jean@biostat.ucsf.edu>,
Martin Morgan <Martin.Morgan@RoswellPark.org>

Maintainer Yee Hwa (Jean) Yang <jean@biostat.ucsf.edu>

Depends R (>= 2.6.0), Biobase (>= 1.15.33), limma (>= 1.7.0), marray,
utils, methods

Description Define coerce methods for microarray data objects.

License LGPL

URL <http://bioinf.wehi.edu.au/limma/convert.html>

biocViews Infrastructure, Microarray, TwoChannel

git_url <https://git.bioconductor.org/packages/convert>

git_branch devel

git_last_commit 9922d10

git_last_commit_date 2024-10-29

Repository Bioconductor 3.21

Date/Publication 2025-02-20

Contents

coerce	2
Index	4

 coerce

Convert Data Objects

Description

Convert between limma, marray and Biobase data objects.

Details

Objects can be converted (coerced) from one class to another using `as(object, Class)` where `object` is an object to convert and `Class` is the name of the class to convert to. The following conversions are provided:

From:	To:
RGList	marrayRaw
marrayRaw	RGList
MAList	marrayNorm
marrayNorm	MAList
RGList	NChannelSet
marrayRaw	NChannelSet
MAList	ExpressionSet
marrayNorm	ExpressionSet

RGList and marrayRaw are coerced to NChannelSet. Channel values are not transformed.

MAList and marrayNorm are coerced so that the ExpressionSet slot contains log-ratios (M-values) and the ExpressionSet object has the same number of columns as the original object. In this case, information on the A-values is lost.

There is intentionally no conversion from RGList or marrayRaw to ExpressionSet, as ExpressionSet is intended for expression values, not intensities.

Author(s)

Gordon Smyth and others

See Also

[as](#) in the methods package.

Examples

```
##first set up some fake intensity matrices
testRed <- matrix(rnorm(5*2),5,2,
  dimnames=list(paste("gene",1:5, sep=""), c("S1", "S2")))
testGreen <- matrix(rnorm(5*2),5,2,
  dimnames=list(paste("gene",1:5, sep=""), c("S1", "S2")))
```

```
##some sample/target info
testTarget <- data.frame(slide=c("S1", "S2"), Cy3=c("T", "C"),
  Cy5=c("C", "T"), row.names=c("S1", "S2"))

maT <- new("marrayInfo", maLabels=c("S1", "S2"),
  maInfo= testTarget)

##now create instances and convert
x <- new("RGList")
x$R <- testRed
x$G <- testGreen
y <- as(x,"marrayRaw")
z <- as(x, "NChannelSet")

x <- new("marrayRaw")
x@maGf <- testGreen
x@maRf <- testRed
x@maTargets = maT
y <- as(x,"RGList")
z <- as(x, "NChannelSet")

x <- new("MArrayList")
y <- as(x,"marrayNorm")

##we construct a reasonably complete fake, small
##instance of the marrayNorm class
x <- new("marrayNorm")
x@maM <- testRed
x@maA <- testGreen
maTargets(x) = maT
y <- as(x,"MArrayList")
y <- as(x, "ExpressionSet")

x <- new("MArrayList")
x$M <- testRed
x$A <- testGreen
x$targets <- testTarget
y <- as(x,"ExpressionSet")
```

Index

* **classes**

coerce, [2](#)

* **data**

coerce, [2](#)

as, [2](#)

coerce, [2](#)

coerce, MAlis, ExpressionSet-method
(coerce), [2](#)

coerce, MAlis, marrayNorm-method
(coerce), [2](#)

coerce, marrayNorm, ExpressionSet-method
(coerce), [2](#)

coerce, marrayNorm, MAlis-method
(coerce), [2](#)

coerce, marrayRaw, NChannelSet-method
(coerce), [2](#)

coerce, marrayRaw, RGLis-method
(coerce), [2](#)

coerce, RGLis, marrayRaw-method
(coerce), [2](#)

coerce, RGLis, NChannelSet-method
(coerce), [2](#)

convert (coerce), [2](#)