Package ‘Harman’

April 25, 2017

Type Package

Title The removal of batch effects from datasets using a PCA and constrained optimisation based technique

Version 1.4.0

Date 2016-03-31

Description Harman is a PCA and constrained optimisation based technique that maximises the removal of batch effects from datasets, with the constraint that the probability of overcorrection (i.e. removing genuine biological signal along with batch noise) is kept to a fraction which is set by the end-user.

NeedsCompilation yes

Suggests HarmanData, BiocGenerics, BiocStyle, knitr, rmarkdown, RUnit, RColorBrewer, bladderbatch, limma, minfi, lumi, msmsEDA, affydata, minfiData, sva

Depends R (>= 3.3)

Imports Rcpp (>= 0.11.2), graphics, stats

LinkingTo Rcpp

License GPL-3 + file LICENCE

LazyData true

biocViews BatchEffect, Microarray, MultipleComparison, PrincipalComponent, Normalization, Preprocessing, DNAMethylation, Transcription, Software, StatisticalMethod

VignetteBuilder knitr

URL http://www.bioinformatics.csiro.au/harman/

BugReports https://github.com/JasonR055/Harman/issues

RoxygenNote 5.0.1

Author Josh Bowden [aut], Jason Ross [aut, cre], Yalchin Oytam [aut]

Maintainer Jason Ross <jason.ross@csiro.au>
Description

Generates an arrow plot for an instance of `harmanresults`. The tail of the arrow is the starting point (original) in principle coordinates, while the arrow head is the new point (corrected) in principle coordinates. It can be observed that on principle components that have undergone correction (`codeharmanresults$stats$correction < 1.0`), the samples within a batch will be coordinately moved towards 0 on that principle component.

Usage

```r
arrowPlot(harmanresults, pc_x = 1, pc_y = 2, colBy = "batch",
          palette = "rainbow", col, length = 0.1, legend = TRUE, ...)
```

Arguments

- `harmanresults` : an instance of `harmanresults`.
- `pc_x` : integer, principle component for the plot x dimension.
- `pc_y` : integer, principle component for the plot y dimension.
- `colBy` : string, colour the points by the experimental or batch variable; legal values are `expt` and `batch`. The palette function specified in `palette` is used. This parameter is overridden by `col`.
- `palette` : string, the function to call to create a vector of contiguous colours with the levels of factor in `colBy` steps.
- `col`, : colour vector for the points. This parameter overrides `palette`.
- `length` : length of the arrow heads, default is 0.1.
- `legend` : logical, whether to display a legend on the plot
- `...` : further arguments passed to or from other methods.
callHarman

Details

Generates a Principle Component plot for an instance of harmanresults. If a vector of colours is supplied via the col argument, then a legend will not be drawn.

Value

None

See Also

harmanresults plot.harmanresults

Examples

library(HarmanData)
data(OLF)
expt <- olf.info$Treatment
batch <- olf.info$Batch
olf.harman <- harman(olf.data, expt, batch)
arrowPlot(olf.harman, pc_x=2, pc_y=3, length=0.2)

callHarman

Wrapper function to call the shared C/C++ library code

Description

This wrapper should probably not be addressed directly except for debugging. Instead use harman. Input of PCA scores and the experiment structure (treatments and batches) and returns a batch corrected version of the PCA scores matrix

Usage

.callHarman(pc_data_scores, group, limit, numrepeats, randseed, forceRand, printInfo)

Arguments

pc_data_scores 2D NumericMatrix of PCA scores data (from the prcomp$x slot), rows = samples, cols = PC scores
group The structure of the experiment, consisting of batch numbers and treatment numbers forming 2 rows or columns (HarmanMain works out which). Each entry for a sample describes what batch it came from and what treatment it was given. Has to be integer formatted data.
limit A double precision value indicating the limit of confidence in which to stop removing a batch effect
numrepeats The number of repeats in which to run the simulated batch mean distribution estimator. Probably should be greater than 100,000.
randseed Random seed to pass to the random number generator (0 for use default from system time)
forceRand Force algorithm
printInfo Print update information to screen
Value

SEXP R list: scores.corrected = harman_res_list["corrected_scores"]
correction = harman_res_list["correction"]
confidence = harman_res_list["confidence"]

Note

A data matrix with samples in columns must be transposed before PCA analysis and these scores in
turn are tweaked a little before handing over to .callHarman. See the example below.

detachHarman

Detach the Harman package and its shared C/C++ library code

Description

A helper function that can be called if harman had to be aborted.

Usage

detachHarman()

Value

None

harman

Harman batch correction method

Description

Harman is a PCA and constrained optimisation based technique that maximises the removal of
batch effects from datasets, with the constraint that the probability of overcorrection (i.e. removing
genuine biological signal along with batch noise) is kept to a fraction which is set by the end-user

Harman expects unbounded data, so for example, with HumanMethylation450 arrays do not use the
Beta statistic (with values constrained between 0 and 1), instead use the logit transformed M-values.

Usage

harman(datamatrix, expt, batch, limit = 0.95, numrepeats = 100000L,
randseed, forceRand = FALSE, printInfo = FALSE)
Arguments

datamatrix  matrix or data.frame, the data values to correct with samples in columns and data values in rows. Internally, a data.frame will be coerced to a matrix. Matrices need to be of type integer or double.

expt  vector or factor with the experimental variable of interest (variance to be kept).

batch  vector or factor with the batch variable (variance to be removed).

limit  numeric, confidence limit. Indicates the limit of confidence in which to stop removing a batch effect. Must be between 0 and 1.

numrepeats  integer, the number of repeats in which to run the simulated batch mean distribution estimator using the random selection algorithm. (N.B. 32 bit Windows versions may have an upper limit of 300000 before catastrophic failure)

randseed  integer, the seed for random number generation.

forceRand  logical, to enforce Harman to use a random selection algorithm to compute corrections. Force the simulated mean code to use random selection of scores to create the simulated batch mean (rather than full explicit calculation from all permutations).

printInfo  logical, whether to print information during computation or not.

Details

The datamatrix needs to be of type integer or numeric, or alternatively a data.frame that can be coerced into one using as.matrix. The matrix is to be constructed with data values (typically microarray probes or sequencing counts) in rows and samples in columns, much like the ‘assayData’ slot in the canonical Bioconductor eSet object, or any object which inherits from it. The data should have normalisation and any other global adjustment for noise reduction (such as background correction) applied prior to using Harman. For converge, the number of simulations, numrepeats parameter should probably should be at least 100,000. The underlying principle of Harman rests upon PCA, which is a parametric technique. This implies Harman should be optimal when the data is normally distributed. However, PCA is known to be rather robust to very non-normal data.

Value

A harmanresults S3 object.

References


See Also

harman, reconstructData, pcaPlot, arrowPlot

Examples

library(HarmanData)
data(OLF)
expt <- olf.info$Treatment
batch <- olf.info$Batch
olf.harman <- harman(olf.data, expt, batch)
plot(olf.harman)
olf.data.corrected <- reconstructData(olf.harman)
## Reading from a csv file

datafile <- system.file("extdata", "NPM_data_first_1000_rows.csv.gz", package="Harman")
infofile <- system.file("extdata", "NPM_info.csv.gz", package="Harman")
datamatrix <- read.table(datafile, header=TRUE, sep="", row.names="probeID")
batches <- read.table(infofile, header=TRUE, sep="", row.names="Sample")
res <- harman(datamatrix, expt=batches$Treatment, batch=batches$Batch)
arrowPlot(res, 1, 3)

### harmanresults

#### Harman results object

**Description**

The S3 object returned after running `harman`.

**Details**

`harmanresults` is the S3 object used to store the results from `harman`. This object may be presented to summary and data exploration functions such as `plot.harmanresults` and `summary.harmanresults` as well as the `reconstructData` function which creates a corrected matrix of data with the batch effect removed.

**Slots**

- `factors`: A data.frame of the `expt` and `batch` vectors.
- `parameters`: The `harman` runtime parameters. See `harman` for details.
- `stats`: Confidence intervals and the degree of correction for each principal component.
- `center`: The centering vector returned by `prcomp` with `center=TRUE`.
- `rotation`: The matrix of eigenvectors (by column) returned from `prcomp`.
- `original`: The original PC scores returned by `prcomp`.
- `corrected`: The `harman` corrected PC scores.

**See Also**

`harman`, `reconstructData`, `pcaPlot`, `arrowPlot`

**Examples**

```r
## HarmanResults
library(HarmanData)
data(OLF)
expt <- olf.info$Treatment
batch <- olf.info$Batch
olf.harman <- harman(as.matrix(olf.data), expt, batch)
plot(olf.harman)
summary(olf.harman)
pcaPlot(olf.harman, pc_x=2, pc_y=3)
pcaPlot(olf.harman, pc_x=2, pc_y=3, colBy='expt', pch=1)
olf.data.corrected <- reconstructData(olf.harman)
```
**harmanScores**

**A Principal components prcomp function tweaked for Harman**

**Description**

A tweaking of `stats::prcomp` such that for the svd, the transpose of u is used instead of v when the number of assays is less than the number of samples.

**Usage**

```
harmanScores(x)
```

**Arguments**

- `x` matrix, data matrix of values to perform PCA on.

**Value**

scores, a `prcomp`-like object with rotation, scores and the center values. The scores are corrected, but all three are needed later to reconstruct the data.

---

**pcaPlot**

**PCA plot for harman results**

**Description**

Generates a Principle Component plot for an instance of `harmanresults`.

**Usage**

```
pcaPlot(harmanresults, pc_x = 1, pc_y = 2, this = "corrected",
        colBy = "batch", pchBy = "expt", palette = "rainbow", legend = TRUE,
        col, pch, ...)
```

**Arguments**

- `harmanresults` An instance of `harmanresults`.
- `pc_x` integer, principle component for the plot x dimension.
- `pc_y` integer, principle component for the plot y dimension.
- `this` string, colour the points by the experimental or batch variable; legal values are `original` or `corrected`.
- `colBy` string, point-type by the experimental or batch variable; legal values are `expt` and `batch`. The palette function specified in `palette` is used. This parameter is overridden by `col`.
- `pchBy` string, point-type by the experimental or batch variable; legal values are `expt` and `batch`. This parameter is overridden by `pch`.
- `palette` string, the function to call to create a vector of contiguous colours with the levels of factor in `colBy` steps.
- `legend` logical, whether to display a legend on the plot.
plot.harmanresults

co1, colour vector for the points. This parameter overrides colBy and palette.
pch, integer vector giving the point type. This parameter overrides pchBy.
... further arguments passed to or from other methods.

Details
If a vector of colours is supplied via the co1 argument, then a legend will not be drawn.

Value
None

See Also
harmanresults plot.harmanresults

Examples
library(HarmanData)
data(OLF)
expt <- olf.info$Treatment
batch <- olf.info$Batch
olf.harman <- harman(as.matrix(olf.data), expt, batch)
pcaPlot(olf.harman)
pcaPlot(olf.harman, colBy='Var
expt
Var
batch')
pcaPlot(olf.harman, pc_x=2, pc_y=3, this='original', pch=17)

plot.harmanresults  Plot method for harman

Description
Plot method for instances of harmanresults.

Usage
## S3 method for class 'harmanresults'
plot(x, ...)

Arguments
x An instance of harmanresults.
... further plotting parameters.

Value
None

See Also
harmanresults pcaPlot
**Examples**

```r
library(HarmanData)
data(OLF)
expt <- olf.info$Treatment
batch <- olf.info$Batch
olf.harman <- harman(olf.data, expt, batch)
plot(olf.harman)
```

**Description**

Generates a Principle Component plot for data.frames, matrices, or a pre-made `prcomp` object.

**Usage**

```r
prcompPlot(object, pc_x = 1, pc_y = 2, scale = FALSE, colFactor = NULL, pchFactor = NULL, palette = "rainbow", legend = TRUE, ...)
```

**Arguments**

- `object`: data.frame, matrix or `prcomp` object.
- `pc_x`: integer, principle component for the plot x dimension.
- `pc_y`: integer, principle component for the plot y dimension.
- `scale`: logical, whether to scale to unit variance before PCA.
- `colFactor`: factor or vector, colour the points by this factor, default is `NULL`.
- `pchFactor`: factor or vector, point-type by this factor, default is `NULL`.
- `palette`: string, the function to call to create a vector of contiguous colours with `levels(colFactor)` steps.
- `legend`: logical, whether to display a legend on the plot.
- `...`: further arguments passed to or from other methods.

**Details**

A data.frame object will be coerced internally to a matrix. Matrices must be of type double or integer. The `prcompPlot` function will then perform a principle component analysis on the data prior to plotting. The function is call is `prcomp(t(object), retx=TRUE, center=TRUE, scale.=scale)`. Instead of specifying a data.frame or matrix, a pre-made `prcomp` object can be given to `prcompPlot`. In this case, care should be taken in setting the appropriate value of `scale`. If a vector is given to `colFactor` or `pchFactor`, they will be coerced internally to factors.

For the default `NULL` values of `colFactor` and `pchFactor`, all colours will be black and circles the point type, respectively.

**Value**

None
See Also

prcomp rainbow

Examples

library(HarmanData)
data(IMR90)
expt <- imr90.info$Treatment
batch <- imr90.info$Batch
prcompPlot(imr90.data, colFactor=expt)
pca <- prcomp(t(imr90.data), scale.=TRUE)
prcompPlot(pca, 1, 3, colFactor=batch, pchFactor=expt, palette='topo.colors',
main='IMR90 PCA plot of Dim 1 and 3')

print.summary.harmanresults
  Printing Harmanresults summaries.

Description

Print method for summary.harmanresults.

Usage

## S3 method for class 'summary.harmanresults'
print(x, ...)

Arguments

x  an object of class summary.harmanresults, usually, a result of a call to summary.harmanresults.
...
   further parameters.

Value

Prints summary information from an object of class summary.harmanresults.

reconstructData  Reconstruct corrected data from Harman results

Description

Method which reverts the PCA factorisation for instances of harmanresults. This allows
the original or corrected data to be returned back from the PCA domain into the original data domain.

Usage

reconstructData(object, this = "corrected")
shiftBetas

Arguments

object       An instance of harmanresults.
this         string, legal values are original or corrected.

Value

matrix of data

See Also

harman harmanresults

Examples

library(HarmanData)
data(OLF)
expt <- olf.info$Treatment
batch <- olf.info$Batch
olf.harman <- harman(olf.data, expt, batch)
olf.data.corrected <- reconstructData(olf.harman)

shiftBetas  Shift beta values from 0 and 1 to avoid infinite M values

Description

A convenience function for methylation data.

Usage

shiftBetas(betas, shiftBy = 1e-04)

Arguments

betas       matrix, beta values.
shiftBy     numeric, the amount to shift values of 0 and 1 by.

Value

None

Examples

betas <- seq(0, 1, by=0.05)
range(betas)
newBetas <- shiftBetas(betas, shiftBy=1e-4)
newBetas
range(newBetas)
Summary method for class \texttt{harmanresults}.

**Usage**

```r
## S3 method for class 'harmanresults'
summary(object, ...)
```

**Arguments**

- `object` An object of class \texttt{harmanresults}.
- `...` further parameters.

**Value**

Returns an object of class \texttt{summary.harmanresults}.

**See Also**

\texttt{harmanresults}

**Examples**

```r
library(HarmanData)
data(OLF)
expt <- olf.info$Treatment
batch <- olf.info$Batch
olf.harman <- harman(olf.data, expt, batch)
summary(olf.harman)
```
Index

.callHarman (callHarman), 3
arrow, 2
arrowPlot, 2, 5, 6
as.matrix, 5
callHarman, 3
detachHarman, 4
harman, 3, 4, 4, 5, 6, 11
harmanResults, 2, 3, 5, 6, 7, 8, 10–12
harmanScores, 7
 pcaPlot, 5, 6, 7, 8
plot.harmanresults, 3, 6, 8, 8
prcomp, 6, 9, 10
prcompPlot, 9
print.summary.harmanresults, 10
rainbow, 10
reconstructData, 5, 6, 10
shiftBetas, 11
summary.harmanresults, 6, 12