Package ‘bapred’

June 3, 2016

Type Package

Title Batch Effect Removal and Addon Normalization (in Phenotype Prediction using Gene Data)

Version 1.0

Date 2016-06-02

Author Roman Hornung, David Causeur

Maintainer Roman Hornung <hornung@ibe.med.uni-muenchen.de>

Depends R (>= 3.1.0), glmnet, lme4, MASS, sva, affyPLM

Imports FNN, fuzzyRankTests, mnormt, affy, Biobase

Suggests ArrayExpress

Description Various tools dealing with batch effects, in particular enabling the removal of discrepancies between training and test sets in prediction scenarios. Moreover, addon quantile normalization and addon RMA normalization (Kostka & Spang, 2008) is implemented to enable integrating the quantile normalization step into prediction rules. The following batch effect removal methods are implemented: FAbatch, ComBat, (f)SVA, mean-centering, standardization, Ratio-A and Ratio-G. For each of these we provide an additional function which enables a posteriori ('addon') batch effect removal in independent batches ('test data'). Here, the (already batch effect adjusted) training data is not altered. For evaluating the success of batch effect adjustment several metrics are provided. Moreover, the package implements a plot for the visualization of batch effects using principal component analysis. The main functions of the package for batch effect adjustment are ba() and baaddon() which enable batch effect removal and addon batch effect removal, respectively, with one of the seven methods mentioned above. Another important function here is bametric() which is a wrapper function for all implemented methods for evaluating the success of batch effect removal. For (addon) quantile normalization and (addon) RMA normalization the functions qunormtrain(), qunormaddon(), rmatrain() and rmaaddon() can be used.

License GPL-2

Encoding UTF-8

NeedsCompilation no

Repository CRAN

Date/Publication 2016-06-03 19:12:25

R topics documented:

`bapred-package` *The bapred package*

Description

This is a short summary of the features of **bapred**, a package in R for the treatment and analysis of batch effects based in high-dimensional molecular data via batch effect adjustment and addon quantile normalization. Here, a special focus is set on phenotype prediction in the presence of batch effects.

Details

Various tools dealing with batch effects, in particular enabling the removal of discrepancies between training and test sets in prediction scenarios. Moreover, addon quantile normalization and addon RMA normalization (Kostka & Spang, 2008) is implemented to enable integrating the quantile normalization step into prediction rules. The following batch effect removal methods are implemented: FAbatch, ComBat, (f)SVA, mean-centering, standardization, Ratio-A and Ratio-G. For each of these we provide an additional function which enables a posteriori ('addon') batch effect removal in independent batches ('test data'). Here, the (already batch effect adjusted) training data is not altered. For evaluating the success of batch effect adjustment several metrics are provided. Moreover, the package implements a plot for the visualization of batch effects using principal component analysis. The main functions of the package for batch effect adjustment are `ba()` and `baaddon()` which enable batch effect removal and addon batch effect removal, respectively, with one of the seven methods mentioned above. Another important function here is `bametric()` which is a wrapper function for all implemented methods for evaluating the success of batch effect removal. For (addon) quantile normalization and (addon) RMA normalization the functions `qunormtrain()`, `qunormaddon()`, `rmatrain()` and `rmaaddon()` can be used.

Author(s)

Roman Hornung, David Causeur

Maintainer: Roman Hornung <hornung@ibe.med.uni-muenchen.de>

References

Hornung, R., Boulesteix, A.-L., Causeur, D. (2016a) Combining location-and-scale batch effect adjustment with data cleaning by latent factor adjustment. BMC Bioinformatics 17:27.

Hornung, R., Causeur, D., Bernau, C., Boulesteix, A.-L. (2016b). Improving cross-study prediction through addon batch effect adjustment and addon normalization. Technical Report, Department of Statistics, LMU.

Examples

```
# Load example dataset:

data(autism)
```

```
# Subset this example dataset to reduce the
# computational burden of the toy examples:

# Random subset of 150 variables:
set.seed(1234)
Xsub <- X[,sample(1:ncol(X), size=150)]

# In cases of batches with more than 20 observations
# select 20 observations at random:
subinds <- unlist(sapply(1:length(levels(batch)), function(x) {
  indbatch <- which(batch==x)
  if(length(indbatch) > 20)
    indbatch <- sort(sample(indbatch, size=20))
  indbatch
}))
Xsub <- Xsub[subinds,]
batchsub <- batch[subinds]
ysub <- y[subinds]



# Split into training and test sets:

trainind <- which(batchsub %in% c(1,2))

Xsubtrain <- Xsub[trainind,]
ysubtrain <- ysub[trainind]
batchsubtrain <- factor(as.numeric(batchsub[trainind]), levels=c(1,2))


testind <- which(batchsub %in% c(3,4))

Xsubtest <- Xsub[testind,]
ysubtest <- ysub[testind]

batchsubtest <- as.numeric(batchsub[testind])
batchsubtest[batchsubtest==3] <- 1
batchsubtest[batchsubtest==4] <- 2
batchsubtest <- factor(batchsubtest, levels=c(1,2))



# Batch effect adjustment:

combatparams <- ba(x=Xsubtrain, y=ysubtrain, batch=batchsubtrain,
  method = "combat")
Xsubtraincombat <- combatparams$xadj

meancenterparams <- ba(x=Xsubtrain, y=ysubtrain, batch=batchsubtrain,
  method = "meancenter")
Xsubtrainmeancenter <- meancenterparams$xadj
```

```
# Addon batch effect adjustment:

Xsubtestcombat <- baaddon(params=combatparams, x=Xsubtest,
  batch=batchsubtest)

Xsubtestmeancenter <- baaddon(params=meancenterparams, x=Xsubtest,
  batch=batchsubtest)



# Metrics for evaluating the success of batch effect adjustment:

bametric(xba=Xsubtrain, batch=batchsubtrain, metric = "sep")
bametric(xba=Xsubtrainmeancenter, batch=batchsubtrain, metric = "sep")

bametric(x=Xsubtrain, batch=batchsubtrain, y=ysubtrain,
  metric = "diffexpr", method = "meancenter")

bametric(xba=Xsubtrainmeancenter, x=Xsubtrain, metric = "cor")



# Principal component analysis plots for the visualization
# of batch effects:

par(mfrow=c(1,3))
pcplot(x=Xsub, batch=batchsub, y=ysub, alpha=0.25, main="alpha = 0.25")
pcplot(x=Xsub, batch=batchsub, y=ysub, alpha=0.75, main="alpha = 0.75")
pcplot(x=Xsub, batch=batchsub, y=ysub, col=1:length(unique(batchsub)),
  main="col = 1:length(unique(batchsub))")
par(mfrow=c(1,1))



# (Addon) quantile normalization:

qunormparams <- qunormtrain(x=Xsubtrain)

Xtrainnorm <- qunormparams$xnorm

Xtestaddonnorm <- qunormaddon(qunormparams, x=Xsubtest)
```

autism *Autism dataset*

Description

Total RNA obtained from lmyphoblast cell lines derived from 250 individuals, 137 of which suffer from autism and 113 are healthy. The dataset consists of four batches of sizes 101, 96, 45 and 8.

Usage

```
data(autism)
```

Format

1) `X` - the covariate matrix: a matrix of dimension 250 x 1000, containing the numerical transcript values

2) `batch` - the batch variable: a factor with levels '1', '2', '3' and '4'

3) `y` - the target variable: a factor with levels '1' corresponding to 'healthy' and '2' corresponding to 'autism'

Details

The RNA measurements were obtained by the Illumina HumanRef-8 v3.0 Expression BeadChip featuring 24,526 transcripts. To reduce computational burden of potential analyses performed using this dataset we randomly selected 1,000 of these 24,526 transcripts. Moreover, the original dataset consisted of five batches and contained measurements of 439 individuals. Again to reduce computational burden of potential analyses we excluded the biggest batch featuring 189 individuals resulting in the 250 individuals included in the dataset made available in `bapred`.

Source

ArrayExpress, accession number: E-GEOD-37772

References

Luo, R., Sanders, S. J., Tian, Y., Voineagu, I., Huang, N., Chu, S. H., Klei, L., Cai, C., Ou, J., Lowe, J. K., Hurles, M. E., Devlin, B., State, M. W., Geschwind, D. H. (2012) Genome-wide Transcriptome Profiling Reveals the Functional Impact of Rare De Novo and Recurrent CNVs in Autism Spectrum Disorders. The American Journal of Human Genetics, 91, 38-–55.

Examples

```
data(autism)
```

`avedist` *Average minimal distance between batches*

Description

This metric is concerned with the minimal distances between pairs of batches.

Usage

```
avedist(xba, batch)
```

Arguments

`xba`	matrix. The covariate matrix, raw or after batch effect adjustment. observations in rows, variables in columns.
`batch`	factor. Batch variable. Currently has to have levels: '1', '2', '3' and so on.

Details

For two batches j and j* (see next paragraph for the case with more batches): 1) for each observation in batch j the euclidean distance to the nearest observation in batch j* is calculated; 2) the roles of j and j* are switched and 1) is re-performed; 3) the average is taken over all n_j + n_j* minimal distances.

For more than two batches: 1) for all possible pairs of batches: calculate the metric as described above; 2) calculate the weighted average of the values in 1) with weights proportional to the sum of the sample sizes in the two respective batches.

The variables are standardized before the calculation to make the metric independent of scale.

Value

Value of the metric

Note

The smaller the values of this metric, the better.

Author(s)

Roman Hornung

References

Lazar, C., Meganck, S., Taminau, J., Steenhoff, D., Coletta, A., Molter,C., Weiss-Solís, D. Y., Duque, R., Bersini, H., Nowé, A. (2012) Batch effect removal methods for microarray gene expression data integration: a survey. Briefings in Bioinformatics, 14(4), 469-490.

Hornung, R., Boulesteix, A.-L., Causeur, D. (2016) Combining location-and-scale batch effect adjustment with data cleaning by latent factor adjustment. BMC Bioinformatics 17:27.

Examples

```
data(autism)

avedist(xba=X, batch=batch)
```

ba *Batch effect adjustment using a method of choice*

Description

Performs batch effect adjustment using one of the following methods: FAbatch, ComBat, SVA, mean-centering, standardization, Ratio-A, Ratio-G or "no batch effect adjustment". Additionally returns information necessary for addon batch effect adjustment with the respective method. The latter can be done using baaddon.

Usage

```
ba(x, y, batch, method = c("fabatch", "combat", "sva", "meancenter",
  "standardize", "ratioa", "ratiog", "none"), ...)
```

Arguments

x	matrix. The covariate matrix. observations in rows, variables in columns.
y	factor. Binary target variable. Currently has to have levels '1' and '2'. Only used for method = "fabatch" and method = "sva".
batch	factor. Batch variable. Currently has to have levels: '1', '2', '3' and so on.
method	character. Batch effect adjustment method.
...	additional arguments to be passed to fabatch or svaba.

Details

This function is merely for convenience - a wrapper function for fabatch, combatba, svaba, meancenter, standardize, ratioa, ratiog and noba.

Value

The output of fabatch, combatba, svaba, meancenter, standardize, ratioa, ratiog or noba respectively.

Note

The following methods are NOT recommended in cross-study prediction settings: FAbatch (fabatch), frozen SVA (svaba), standardization (standardize) as well as no addon batch effect adjustment (noba).

Given a not too small test set, **the following methods are recommended** (Hornung et al., 2016b): ComBat (combatba), mean-centering (meancenter), Ratio-A (ratioa), Ratio-G (ratiog).

Author(s)

Roman Hornung

References

Hornung, R., Boulesteix, A.-L., Causeur, D. (2016a) Combining location-and-scale batch effect adjustment with data cleaning by latent factor adjustment. BMC Bioinformatics 17:27.

Hornung, R., Causeur, D., Bernau, C., Boulesteix, A.-L. (2016b). Improving cross-study prediction through addon batch effect adjustment and addon normalization. Technical Report, Department of Statistics, LMU.

Johnson, W. E., Rabinovic, A., Li, C. (2007) Adjusting batch effects in microarray expression data using empirical bayes methods. Biostatistics, 8, 118-127.

Leek, J. T., Storey, J. D. (2007) Capturing Heterogeneity in Gene Expression Studies by Surrogate Variable Analysis. PLoS Genetics, 3, 1724–1735.

Luo, J., Schumacher, M., Scherer, A., Sanoudou, D., Megherbi, D., Davison, T., Shi, T., Tong, W., Shi, L., Hong, H., Zhao, C., Elloumi, F., Shi, W., Thomas, R., Lin, S., Tillinghast, G., Liu, G., Zhou, Y., Herman, D., Li, Y., Deng, Y., Fang, H., Bushel, P., Woods, M., Zhang, J. (2010) A comparison of batch effect removal methods for enhancement of prediction performance using maqc-ii microarray gene expression data. The Pharmacogenomics Journal, 10, 278-291.

Parker, H. S., Bravo, H. C., Leek, J. T. (2014) Removing batch effects for prediction problems with frozen surrogate variable analysis. PeerJ, 2, e561.

Examples

```
data(autism)

# Random subset of 150 variables:
set.seed(1234)
Xsub <- X[,sample(1:ncol(X), size=150)]

# In cases of batches with more than 20 observations
# select 20 observations at random:
subinds <- unlist(sapply(1:length(levels(batch)), function(x) {
  indbatch <- which(batch==x)
  if(length(indbatch) > 20)
    indbatch <- sort(sample(indbatch, size=20))
  indbatch
}))
Xsub <- Xsub[subinds,]
batchsub <- batch[subinds]
ysub <- y[subinds]


somemethods <- c("fabatch", "combat", "meancenter", "none")

adjusteddata <- list()

for(i in seq(along=somemethods)) {
  cat(paste("Adjusting using method = \"", somemethods[i], "\"",
    sep=""), "\n")
  adjusteddata[[i]] <- ba(x=Xsub, y=ysub, batch=batchsub,
    method = somemethods[i])$xadj
}
```

baaddon — *Addon batch effect adjustment*

Description

Performs addon batch effect adjustment for a method of choice: takes the output of `ba` or that of one of the functions performing a specific batch effect adjustment method (e.g. `fabatch` or `svaba`) and new batch data. Then performs the respective batch effect adjustment method on the new batch data.

Usage

```
baaddon(params, x, batch)
```

Arguments

`params`	object of class `fabatch`, `combat`, `svatrain`, `meancenter`, `standardize`, `ratioa`, `ratiog` or `noba`. Contains parameters necessary for addon batch effect adjustment according to the respective method.
`x`	matrix. The covariate matrix of the new data. Observations in rows, variables in columns.
`batch`	factor. Batch variable of the new data. Currently has to have levels: '1', '2', '3' and so on.

Value

The adjusted covariate matrix of the test data.

Note

The following methods are NOT recommended in cross-study prediction settings: FAbatch (`fabatch`), frozen SVA (`svaba`), standardization (`standardize`) as well as no addon batch effect adjustment (`noba`).

Given a not too small test set, **the following methods are recommended** (Hornung et al., 2016b): ComBat (`combatba`), mean-centering (`meancenter`), Ratio-A (`ratioa`), Ratio-G (`ratiog`).

Author(s)

Roman Hornung

References

Hornung, R., Boulesteix, A.-L., Causeur, D. (2016a) Combining location-and-scale batch effect adjustment with data cleaning by latent factor adjustment. BMC Bioinformatics 17:27.

Hornung, R., Causeur, D., Bernau, C., Boulesteix, A.-L. (2016b). Improving cross-study prediction through addon batch effect adjustment and addon normalization. Technical Report, Department of Statistics, LMU.

Johnson, W. E., Rabinovic, A., Li, C. (2007) Adjusting batch effects in microarray expression data using empirical bayes methods. Biostatistics, 8, 118-127.

Leek, J. T., Storey, J. D. (2007) Capturing Heterogeneity in Gene Expression Studies by Surrogate Variable Analysis. PLoS Genetics, 3, 1724–1735.

Luo, J., Schumacher, M., Scherer, A., Sanoudou, D., Megherbi, D., Davison, T., Shi, T., Tong, W., Shi, L., Hong, H., Zhao, C., Elloumi, F., Shi, W., Thomas, R., Lin, S., Tillinghast, G., Liu, G., Zhou, Y., Herman, D., Li, Y., Deng, Y., Fang, H., Bushel, P., Woods, M., Zhang, J. (2010) A comparison of batch effect removal methods for enhancement of prediction performance using maqc-ii microarray gene expression data. The Pharmacogenomics Journal, 10, 278-291.

Parker, H. S., Bravo, H. C., Leek, J. T. (2014) Removing batch effects for prediction problems with frozen surrogate variable analysis. PeerJ, 2, e561.

Examples

```
data(autism)

# Random subset of 150 variables:
set.seed(1234)
Xsub <- X[,sample(1:ncol(X), size=150)]

# In cases of batches with more than 20 observations
# select 20 observations at random:
subinds <- unlist(sapply(1:length(levels(batch)), function(x) {
  indbatch <- which(batch==x)
  if(length(indbatch) > 20)
    indbatch <- sort(sample(indbatch, size=20))
  indbatch
}))
Xsub <- Xsub[subinds,]
batchsub <- batch[subinds]
ysub <- y[subinds]



trainind <- which(batchsub %in% c(1,2))

Xsubtrain <- Xsub[trainind,]
ysubtrain <- ysub[trainind]
batchsubtrain <- factor(as.numeric(batchsub[trainind]), levels=c(1,2))


testind <- which(batchsub %in% c(3,4))

Xsubtest <- Xsub[testind,]
ysubtest <- ysub[testind]

batchsubtest <- as.numeric(batchsub[testind])
batchsubtest[batchsubtest==3] <- 1
batchsubtest[batchsubtest==4] <- 2
batchsubtest <- factor(batchsubtest, levels=c(1,2))
```

```
somemethods <- c("fabatch", "combat", "meancenter", "none")

adjustedtestdata <- list()

for(i in seq(along=somemethods)) {
  cat(paste("Adjusting training data using method = \"", somemethods[i],
    "\"", sep=""), "\n")
  paramstemp <- ba(x=Xsubtrain, y=ysubtrain, batch=batchsubtrain,
    method = somemethods[i])
  cat(paste("Addon adjusting test data using method = \"",
    somemethods[i], "\"", sep=""), "\n")
  adjustedtestdata[[i]] <- baaddon(params=paramstemp, x=Xsubtest,
    batch=batchsubtest)
}
```

bametric *Diverse metrics for quality of (adjusted) batch data*

Description

Diverse metrics measuring the severeness of batch effects in (batch effect adjusted) data or the performance of certain analyses performed using the latter. This is a wrapper function for the following functions, where each of them calculates a certain metric: `sepscore`, `avedist`, `kldist`, `skewdiv`, `pvcam`, `diffexprm` and `corba`. For details see the documentation of the latter.

Usage

```
bametric(xba, batch, y, x, metric = c("sep", "avedist", "kldist",
  "skew", "pvca", "diffexpr", "cor"), method, ...)
```

Arguments

`xba`	matrix. The covariate matrix, raw or after batch effect adjustment. observations in rows, variables in columns.
`batch`	factor. Batch variable. Currently has to have levels: '1', '2', '3' and so on.
`y`	factor. Binary target variable. Currently has to have levels '1' and '2'. Only used for `metric = "pvca"` and `metric = "diffexpr"`.
`x`	matrix. The covariate matrix before batch effect adjustment. observations in rows, variables in columns.
`metric`	character. The metric to use.
`method`	character. The batch effect adjustment method to use for `metric = "diffexpr"`.
`...`	additional arguments to be passed to `sepscore` or `pvcam`.

Value

Value of the respective metric

Author(s)

Roman Hornung

References

Boltz, S., Debreuve, E., Barlaud, M. (2009) High-dimensional statistical measure for region-of-interest tracking. Transactions in Image Processing, 18(6), 1266-1283.

Geyer, C. J., Meeden, G., D. (2005) Fuzzy and randomized confidence intervals and p-values (with discussion). Statistical Science, 20(4), 358-387.

Hornung, R., Boulesteix, A.-L., Causeur, D. (2016) Combining location-and-scale batch effect adjustment with data cleaning by latent factor adjustment. BMC Bioinformatics 17:27.

Lazar, C., Meganck, S., Taminau, J., Steenhoff, D., Coletta, A., Molter,C., Weiss-Solís, D. Y., Duque, R., Bersini, H., Nowé, A. (2012) Batch effect removal methods for microarray gene expression data integration: a survey. Briefings in Bioinformatics, 14(4), 469-490.

Lee, J. A., Dobbin, K. K., Ahn, J. (2014) Covariance adjustment for batch effect in gene expression data. Statistics in Medicine, 33, 2681-2695.

Li, J., Bushel, P., Chu, T.-M., Wolfinger, R.D. (2009) Principal variance components analysis: Estimating batch effects in microarray gene expression data. In: Scherer, A. (ed) Batch Effects and Noise in Microarray Experiments: Sources and Solutions, John Wiley & Sons, Chichester, UK.

Shabalin, A. A., Tjelmeland, H., Fan, C., Perou, C. M., Nobel, A. B. (2008) Merging two gene-expression studies via cross-platform normalization. Bioinformatics, 24(9), 1154-1160.

Examples

```
data(autism)

# Random subset of 150 variables:
set.seed(1234)
Xsub <- X[,sample(1:ncol(X), size=150)]

# In cases of batches with more than 20 observations
# select 20 observations at random:
subinds <- unlist(sapply(1:length(levels(batch)), function(x) {
  indbatch <- which(batch==x)
  if(length(indbatch) > 20)
    indbatch <- sort(sample(indbatch, size=20))
  indbatch
}))
Xsub <- Xsub[subinds,]
batchsub <- batch[subinds]
ysub <- y[subinds]




Xsubadj <- ba(x=Xsub, y=ysub, batch=batchsub, method = "combat")$xadj


bametric(xba=Xsub, batch=batchsub, metric = "sep")
bametric(xba=Xsubadj, batch=batchsub, metric = "sep")

bametric(xba=Xsub, batch=batchsub, metric = "avedist")
bametric(xba=Xsubadj, batch=batchsub, metric = "avedist")

bametric(xba=Xsub, batch=batchsub, metric = "kldist")
bametric(xba=Xsubadj, batch=batchsub, metric = "kldist")

bametric(xba=Xsub, batch=batchsub, metric = "skew")
bametric(xba=Xsubadj, batch=batchsub, metric = "skew")

bametric(xba=Xsub, batch=batchsub, y=ysub, metric = "pvca")
bametric(xba=Xsubadj, batch=batchsub, y=ysub, metric = "pvca")

bametric(x=Xsub, batch=batchsub, y=ysub, metric = "diffexpr",
  method = "combat")

bametric(xba=Xsubadj, x=Xsub, metric = "cor")
```

batch — *batch variable of dataset* autism

Description

See dataset autism

combatba — *Batch effect adjustment using ComBat*

Description

Performs batch effect adjustment using the parametric version of ComBat and additionally returns information necessary for addon batch effect adjustment with ComBat.

Usage

```
combatba(x, batch)
```

Arguments

x	matrix. The covariate matrix. Observations in rows, variables in columns.
batch	factor. Batch variable. Currently has to have levels: '1', '2', '3' and so on.

Value

`combatba` returns an object of class `combat`. An object of class "`combat`" is a list containing the following components:

`xadj`	matrix of adjusted (training) data
`meanoverall`	vector containing the overall means of the variables. Used in addon adjustment.
`var.pooled`	vector containing the pooled variances of the variables. Used in addon adjustment.
`batch`	batch variable
`nbatches`	number of batches

Note

The original ComBat-code is used in `combatba`: `http://www.bu.edu/jlab/wp-assets/ComBat/Download.html` (Access date: 2015/06/19)

Author(s)

Roman Hornung

References

Johnson, W. E., Rabinovic, A., Li, C. (2007) Adjusting batch effects in microarray expression data using empirical bayes methods. Biostatistics, 8, 118-127.

Luo, J., Schumacher, M., Scherer, A., Sanoudou, D., Megherbi, D., Davison, T., Shi, T., Tong, W., Shi, L., Hong, H., Zhao, C., Elloumi, F., Shi, W., Thomas, R., Lin, S., Tillinghast, G., Liu, G., Zhou, Y., Herman, D., Li, Y., Deng, Y., Fang, H., Bushel, P., Woods, M., Zhang, J. (2010) A comparison of batch effect removal methods for enhancement of prediction performance using maqc-ii microarray gene expression data. The Pharmacogenomics Journal, 10, 278-291.

Examples

```
data(autism)

combatba(x=X, batch=batch)
```

`combatbaaddon` *Addon batch effect adjustment using ComBat*

Description

Performs addon batch effect adjustment using ComBat. Takes the output of performing `combatba` on a training data set and new batch data and correspondingly adjusts the test data to better match the adjusted training data according to the ComBat model

Usage

```
combatbaaddon(params, x, batch)
```

Arguments

params	object of class combat. Contains parameters necessary for addon batch effect adjustment.
x	matrix. The covariate matrix of the new data. Observations in rows, variables in columns.
batch	factor. Batch variable of the new data. Currently has to have levels: '1', '2', '3' and so on.

Value

The adjusted covariate matrix of the test data.

Note

The original ComBat-code is used in combatbaaddon: http://www.bu.edu/jlab/wp-assets/ComBat/Download.html (Access date: 2015/06/19)

Author(s)

Roman Hornung

References

Johnson, W. E., Rabinovic, A., Li, C. (2007) Adjusting batch effects in microarray expression data using empirical bayes methods. Biostatistics, 8, 118-127.

Luo, J., Schumacher, M., Scherer, A., Sanoudou, D., Megherbi, D., Davison, T., Shi, T., Tong, W., Shi, L., Hong, H., Zhao, C., Elloumi, F., Shi, W., Thomas, R., Lin, S., Tillinghast, G., Liu, G., Zhou, Y., Herman, D., Li, Y., Deng, Y., Fang, H., Bushel, P., Woods, M., Zhang, J. (2010) A comparison of batch effect removal methods for enhancement of prediction performance using maqc-ii microarray gene expression data. The Pharmacogenomics Journal, 10, 278-291.

Examples

```
data(autism)

trainind <- which(batch %in% c(1,2))

Xtrain <- X[trainind,]
ytrain <- y[trainind]
batchtrain <- factor(as.numeric(batch[trainind]), levels=c(1,2))


testind <- which(batch %in% c(3,4))

Xtest <- X[testind,]
```

```
ytest <- y[testind]

batchtest <- as.numeric(batch[testind])
batchtest[batchtest==3] <- 1
batchtest[batchtest==4] <- 2
batchtest <- factor(batchtest, levels=c(1,2))


params <- combatba(x=Xtrain, batch=batchtrain)

Xtestaddon <- combatbaaddon(params, x=Xtest, batch=batchtest)
```

---

corba *Mean correlation before and after batch effect adjustment*

Description

For each variable Pearson's correlation of the values before and after batch effect adjustment is calculated. Then the mean is taken over all these correlations.

Usage

```
corba(xba, x)
```

Arguments

`xba`	matrix. The covariate matrix after batch effect adjustment. observations in rows, variables in columns.
`x`	matrix. The covariate matrix before batch effect adjustment. observations in rows, variables in columns.

Value

Value of the metric

Author(s)

Roman Hornung

References

Lazar, C., Meganck, S., Taminau, J., Steenhoff, D., Coletta, A., Molter,C., Weiss-Solís, D. Y., Duque, R., Bersini, H., Nowé, A. (2012) Batch effect removal methods for microarray gene expression data integration: a survey. Briefings in Bioinformatics, 14(4), 469-490.

Examples

```
data(autism)

params <- ba(x=X, y=y, batch=batch, method = "combat")
Xadj <- params$xadj

corba(xba=Xadj, x=X)
```

---

diffexprm — *Measure for performance of differential expression analysis (after batch effect adjustment)*

Description

This metric is similar to the idea presented in Lazar et al (2012) which consists in comparing the list of the most differentially expressed genes obtained using a batch effect adjusted dataset to the list obtained using an independent dataset. For each batch the following is done by `diffexprm`: 1) the respective batch is left out and batch effect adjustment is performed using the remaining batches; 2) differential expression analysis is performed once using the left-out batch and once using the remaining batch-effect adjusted data; 3) the overlap between the two lists of genes found differentially expressed in the two subsets is measured. See below for further details.

Usage

```
diffexprm(x, batch, y, method = c("fabatch", "combat", "sva",
  "meancenter", "standardize", "ratioa", "ratiog", "none"))
```

Arguments

`x`	matrix. The covariate matrix. Observations in rows, variables in columns.
`batch`	factor. Batch variable. Currently has to have levels: '1', '2', '3' and so on.
`y`	factor. Binary target variable. Currently has to have levels '1' and '2'.
`method`	character. Method for batch effect adjustment. The following are supported: `fabatch`, `combat`, `fsva`, `meancenter`, `standardize`, `ratioa`, `ratiog` and `none`

Details

The following procedure is performed: 1) For each batch j leave this batch out and perform batch effect adjustment on the rest of the dataset. Derive two lists of the 5 percent of variables which are most differentially expressed (see next paragraph): one using the batch effect adjusted dataset - where batch j was left out - and one using the data from batch j. Calculate the number of variables appearing in both lists and divide this number by the common length of the lists. 2) Calculate a weighted average of the values obtained in 1) with weights proportional to the number of observations in the corresponding left-out batches.

Differential expression is measured as follows. For each variable a randomized p-value out of the Whitney-Wilcoxon rank sum test is drawn, see Geyer and Meeden (2005) for details. Then those 5 percent variables are considered differentially expressed, which are associated with the smallest p-values.

Value

Value of the metric

Note

The larger the values of this metric, the better.

Author(s)

Roman Hornung

References

Hornung, R., Boulesteix, A.-L., Causeur, D. (2016) Combining location-and-scale batch effect adjustment with data cleaning by latent factor adjustment. BMC Bioinformatics 17:27.

Lazar, C., Meganck, S., Taminau, J., Steenhoff, D., Coletta, A., Molter,C., Weiss-Solís, D. Y., Duque, R., Bersini, H., Nowé, A. (2012) Batch effect removal methods for microarray gene expression data integration: a survey. Briefings in Bioinformatics, 14(4), 469-490.

Geyer, C. J., Meeden, G., D. (2005) Fuzzy and randomized confidence intervals and p-values (with discussion). Statistical Science, 20(4), 358-387.

Examples

```
data(autism)

# Random subset of 150 variables:
set.seed(1234)
Xsub <- X[,sample(1:ncol(X), size=150)]

# In cases of batches with more than 20 observations
# select 20 observations at random:
subinds <- unlist(sapply(1:length(levels(batch)), function(x) {
  indbatch <- which(batch==x)
  if(length(indbatch) > 20)
    indbatch <- sort(sample(indbatch, size=20))
  indbatch
}))
Xsub <- Xsub[subinds,]
batchsub <- batch[subinds]
ysub <- y[subinds]



diffexprm(x=Xsub, batch=batchsub, y=ysub, method = "ratiog")
diffexprm(x=Xsub, batch=batchsub, y=ysub, method = "none")
```

fabatch *Batch effect adjustment using FAbatch*

Description

Performs batch effect adjustment using the FAbatch-method described in Hornung et al. (2016) and additionally returns information necessary for addon batch effect adjustment with FAbatch.

Usage

```
fabatch(x, y, batch, nbf = NULL, minerr = 1e-06,
  probcrossbatch = TRUE, maxiter = 100, maxnbf = 12)
```

Arguments

`x`	matrix. The covariate matrix. Observations in rows, variables in columns.
`y`	factor. Binary target variable. Currently has to have levels '1' and '2'.
`batch`	factor. Batch variable. Currently has to have levels: '1', '2', '3' and so on.
`nbf`	integer. Number of factors to estimate in all batches. If not given the number of factors is estimated automatically for each batch. Recommended to leave unspecified.
`minerr`	numeric. Maximal mean quadratic deviations between the estimated residual variances from two consecutive iterations. The iteration stops when this value is undercut.
`probcrossbatch`	logical. Default is `TRUE`. If `TRUE` the preliminary probabilities are estimated through leave-one-batch-out cross-validation. If set to `FALSE` ordinary cross-validation is used for estimating the preliminary probabilities. This might result in an artificially increased class signal in comparison to that in the data in independent batches. Is automatically set to `FALSE`, when only one batch is present in the training data.
`maxiter`	integer. Maximal number of iterations in the estimation of the latent factors by Maximum Likelihood.
`maxnbf`	integer. Maximal number of factors if `nbf` is not given. Default is the largest integer smaller than half the number of observations in the corresponding batch.

Value

`fabatch` returns an object of class `fabatch`. An object of class `"fabatch"` is a list containing the following components:

`xadj`	matrix of adjusted (training) data
`m1`	means of the standardized variables in class '1'
`m2`	means of the standardized variables in class '2'
`b0`	intercept out of the L2-penalized logistic regression performed for estimation of the class probabilities

`b`	variable coefficients out of the L2-penalized logistic regression performed for estimation of the class probabilities
`pooledsds`	vector containing the pooled standard deviations of the variables
`meanoverall`	vector containing the variable means
`minerr`	maximal mean quadratic deviations between the estimated residual variances from two consecutive iterations
`nbfinput`	user-specified number of latent factors `nbf` in all batches. `NULL` if `nbf` was not specified.
`badvariables`	indices of those variables which are constant in at least one batch
`nbatches`	number of batches
`batch`	batch variable
`nbfvec`	vector containing the numbers of factors in the individual batches

Author(s)

Roman Hornung

References

Hornung, R., Boulesteix, A.-L., Causeur, D. (2016) Combining location-and-scale batch effect adjustment with data cleaning by latent factor adjustment. BMC Bioinformatics 17:27.

Examples

```
data(autism)

# Random subset of 150 variables:
set.seed(1234)
Xsub <- X[,sample(1:ncol(X), size=150)]

# In cases of batches with more than 20 observations
# select 20 observations at random:
subinds <- unlist(sapply(1:length(levels(batch)), function(x) {
  indbatch <- which(batch==x)
  if(length(indbatch) > 20)
    indbatch <- sort(sample(indbatch, size=20))
  indbatch
}))
Xsub <- Xsub[subinds,]
batchsub <- batch[subinds]
ysub <- y[subinds]



fabatch(x=Xsub, y=ysub, batch=batchsub)
```

`fabatchaddon` *Addon batch effect adjustment using FAbatch*

Description

Performs addon batch effect adjustment using FAbatch. Takes the output of performing `fabatch` on a training data set and new batch data and correspondingly adjusts the test data to better match the adjusted training data according to the FAbatch model.

Usage

```
fabatchaddon(params, x, batch)
```

Arguments

`params`	object of class `fabatch`. Contains parameters necessary for addon batch effect adjustment.
`x`	matrix. The covariate matrix of the new data. Observations in rows, variables in columns.
`batch`	factor. Batch variable of the new data. Currently has to have levels: '1', '2', '3' and so on.

Value

The adjusted covariate matrix of the test data.

Note

It is **not recommended** to perform FAbatch in cross-study prediction settings, because it has been observed to (strongly) impair prediction performance. Given a not too small test set, the following methods are recommended (Hornung et al., 2016b): `combatba`, `meancenter`, `ratioa`, `ratiog`.

Author(s)

Roman Hornung

References

Hornung, R., Boulesteix, A.-L., Causeur, D. (2016a) Combining location-and-scale batch effect adjustment with data cleaning by latent factor adjustment. BMC Bioinformatics 17:27.

Hornung, R., Causeur, D., Bernau, C., Boulesteix, A.-L. (2016b). Improving cross-study prediction through addon batch effect adjustment and addon normalization. Technical Report, Department of Statistics, LMU.

Examples

```
data(autism)

# Random subset of 150 variables:
set.seed(1234)
Xsub <- X[,sample(1:ncol(X), size=150)]

# In cases of batches with more than 20 observations
# select 20 observations at random:
subinds <- unlist(sapply(1:length(levels(batch)), function(x) {
  indbatch <- which(batch==x)
  if(length(indbatch) > 20)
    indbatch <- sort(sample(indbatch, size=20))
  indbatch
}))
Xsub <- Xsub[subinds,]
batchsub <- batch[subinds]
ysub <- y[subinds]



trainind <- which(batchsub %in% c(1,2))

Xsubtrain <- Xsub[trainind,]
ysubtrain <- ysub[trainind]
batchsubtrain <- factor(as.numeric(batchsub[trainind]), levels=c(1,2))


testind <- which(batchsub %in% c(3,4))

Xsubtest <- Xsub[testind,]
ysubtest <- ysub[testind]

batchsubtest <- as.numeric(batchsub[testind])
batchsubtest[batchsubtest==3] <- 1
batchsubtest[batchsubtest==4] <- 2
batchsubtest <- factor(batchsubtest, levels=c(1,2))



params <- fabatch(x=Xsubtrain, y=ysubtrain, batch=batchsubtrain)

Xsubtestaddon <- fabatchaddon(params, x=Xsubtest,
  batch=batchsubtest)
```

`kldist` *Kullback-Leibler divergence between density of within and between batch pairwise distances*

Description

This metric estimates the Kullback-Leibler divergences between the distributions of the within and that of the between batch euclidian distances of pairs of observations. These distributions should be similar in the abscence of stronger batch effects.

Usage

```
kldist(xba, batch)
```

Arguments

`xba`	matrix. The covariate matrix, raw or after batch effect adjustment. observations in rows, variables in columns.
`batch`	factor. Batch variable. Currently has to have levels: '1', '2', '3' and so on.

Details

For two batches j and j* (see next paragraph for the case with more batches): 1) the distances between all pairs of observations in batch j - denoted as {dist_j} - and the distances between all such pairs in batch j* - denoted as {dist_j*} - are calculated; 2) for each observation in j the distances to all observations in j* are calculated, resulting in n_j x n_j* distances denoted as {dist_jj*}; calculate the Kullback-Leibler divergence between the densities of {dist_j} and {dist_jj*} and that between the densities of {dist_j*} and {dist_jj*} - using the k-nearest neighbours based method by Boltz et al (2009) with k=5; 3) take the weighted mean of the values of these two divergences with weights proportional to n_j and n_j*.

For more than two batches: 1) for all possible pairs of batches: calculate the metric as described above; 2) calculate the weighted average of the values in 1) with weights proportional to the sum of the sample sizes in the two respective batches.

The variables are standardized before the calculation to make the metric independent of scale.

Value

Value of the metric

Note

The smaller the values of this metric, the better.

Author(s)

Roman Hornung

References

Lee, J. A., Dobbin, K. K., Ahn, J. (2014) Covariance adjustment for batch effect in gene expression data. Statistics in Medicine, 33, 2681-2695.

Boltz, S., Debreuve, E., Barlaud, M. (2009) High-dimensional statistical measure for region-of-interest tracking. Transactions in Image Processing, 18(6), 1266-1283.

Hornung, R., Boulesteix, A.-L., Causeur, D. (2016) Combining location-and-scale batch effect adjustment with data cleaning by latent factor adjustment. BMC Bioinformatics 17:27.

Examples

```
data(autism)

kldist(xba=X, batch=batch)
```

---

meancenter *Batch effect adjustment by mean-centering*

Description

Performs batch effect adjustment by centering the variables within batches to have zero mean.

Usage

```
meancenter(x, batch)
```

Arguments

x	matrix. The covariate matrix of the new data. Observations in rows, variables in columns.
batch	factor. Batch variable. Currently has to have levels: ’1’, ’2’, ’3’ and so on.

Value

meancenter returns an object of class meancenter. An object of class "meancenter" is a list containing the following components:

xadj	matrix of adjusted (training) data
nbatches	number of batches
batch	batch variable

Author(s)

Roman Hornung

Examples

```
data(autism)

params <- meancenter(x=X, batch=batch)
```

meancenteraddon *Addon batch effect adjustment for mean-centering*

Description

Performs addon batch effect adjustment for mean-centering: 1) takes the output of `meancenter` applied to a training data set together with new batch data; 2) checks whether the training data was also adjusted using mean-centering and whether the same number of variables is present in training and new data; 3) performs mean-centering on the new batch data.

Usage

```
meancenteraddon(params, x, batch)
```

Arguments

`params`	object of class `meancenter`.
`x`	matrix. The covariate matrix of the new data. Observations in rows, variables in columns.
`batch`	factor. Batch variable of the new data. Currently has to have levels: ’1’, ’2’, ’3’ and so on.

Value

The adjusted covariate matrix of the test data.

Note

Because mean-centering is performed "batch by batch" the "addon procedure" for mean-centering consists of plain mean-centering on the new test batches.

Author(s)

Roman Hornung

References

Hornung, R., Boulesteix, A.-L., Causeur, D. (2016) Combining location-and-scale batch effect adjustment with data cleaning by latent factor adjustment. BMC Bioinformatics 17:27.

Examples

```
data(autism)

trainind <- which(batch %in% c(1,2))

Xtrain <- X[trainind,]
ytrain <- y[trainind]
```

```
batchtrain <- factor(as.numeric(batch[trainind]), levels=c(1,2))


testind <- which(batch %in% c(3,4))

Xtest <- X[testind,]
ytest <- y[testind]

batchtest <- as.numeric(batch[testind])
batchtest[batchtest==3] <- 1
batchtest[batchtest==4] <- 2
batchtest <- factor(batchtest, levels=c(1,2))


params <- meancenter(x=Xtrain, batch=batchtrain)

Xtestaddon <- meancenteraddon(params=params, x=Xtest, batch=batchtest)
```

---

noba *No batch effect adjustment*

Description

This function is merely included for consistency. It returns the raw dataset not adjusted for batch effects.

Usage

```
noba(x, batch)
```

Arguments

`x`	matrix. The covariate matrix. Observations in rows, variables in columns.
`batch`	factor. Batch variable. Currently has to have levels: '1', '2', '3' and so on.

Value

`noba` returns an object of class `noba`. An object of class "`noba`" is a list containing the following components:

`xadj`	matrix of (training) data
`nbatches`	number of batches
`batch`	batch variable

Author(s)

Roman Hornung

Examples

```
data(autism)

Xadj <- noba(x=X, batch=batch)$adj

all(as.vector(Xadj)==as.vector(X))
```

`nobaaddon` *No addon batch effect adjustment*

Description

This function is merely included for consistency. It does the following: 1) takes the output of `noba` applied to a training data set together with new batch data; 2) checks whether the training data has also not been adjusted using a batch effect adjustment method and whether the same number of variables is present in training and new data; 3) returns the new batch data not adjusted for batch effects.

Usage

```
nobaaddon(params, x, batch)
```

Arguments

`params`	object of class `noba`.
`x`	matrix. The covariate matrix of the new data. Observations in rows, variables in columns.
`batch`	factor. Batch variable of the new data. Currently has to have levels: '1', '2', '3' and so on.

Value

The unadjusted covariate matrix `x` of the test data.

Note

It is **not recommended** to perform no addon batch effect adjustment in cross-study prediction settings. Given a not too small test set, the following methods are recommended (Hornung et al., 2016): `combatba`, `meancenter`, `ratioa`, `ratiog`.

Author(s)

Roman Hornung

References

Hornung, R., Causeur, D., Bernau, C., Boulesteix, A.-L. (2016). Improving cross-study prediction through addon batch effect adjustment and addon normalization. Technical Report, Department of Statistics, LMU.

Examples

```
data(autism)

trainind <- which(batch %in% c(1,2))

Xtrain <- X[trainind,]
ytrain <- y[trainind]
batchtrain <- factor(as.numeric(batch[trainind]), levels=c(1,2))


testind <- which(batch %in% c(3,4))

Xtest <- X[testind,]
ytest <- y[testind]

batchtest <- as.numeric(batch[testind])
batchtest[batchtest==3] <- 1
batchtest[batchtest==4] <- 2
batchtest <- factor(batchtest, levels=c(1,2))


params <- noba(x=Xtrain, batch=batchtrain)

Xtestaddon <- nobaaddon(params=params, x=Xtest, batch=batchtest)


all(as.vector(Xtestaddon)==as.vector(Xtest))
```

`pcplot` *Visualization of batch effects using Principal Component Analysis*

Description

This function performs principal component analysis on the covariate matrix and plots the first two principal components against each other. Different batches are distinguished by different colors and (optionally) the two classes of the target variable by different plot symbols.

Usage

```
pcplot(x, batch, y, alpha = 0.35, ...)
```

Arguments

`x`	matrix. The covariate matrix. Observations in rows, variables in columns.
`batch`	factor. Batch variable. Currently has to have levels: '1', '2', '3' and so on.
`y`	optional factor. Binary target variable. Currently has to have levels '1' and '2'.
`alpha`	optional numeric between 0 and 1. Alpha transparency of the contour lines of the batch-specific two-dimensional density estimates. Only applicable when default color scheme (`rainbow`) is used.
`...`	additional arguments to be passed to `plot`.

Details

For the data corresponding to each batch a two-dimensional kernel density estimate is obtained using the function `kde2d()` from the **MASS**-package. These estimates are depicted through contour lines (using `contour`).

Value

`NULL`

Author(s)

Roman Hornung

Examples

```
data(autism)

par(mfrow=c(1,3))
pcplot(x=X, batch=batch, y=y, alpha=0.25, main="alpha = 0.25")
pcplot(x=X, batch=batch, y=y, alpha=0.75, main="alpha = 0.75")
pcplot(x=X, batch=batch, y=y, col=1:length(unique(batch)),
  main="col = 1:length(unique(batch))")
par(mfrow=c(1,1))
```

`pvcam` *Proportion of variation induced by class signal estimated by Principal Variance Component Analysis*

Description

Principal Variance Component Analysis (PVCA) (Li et al, 2009) allows the estimation of the contribution of several sources of variability. `pvcam` uses it to estimate the proportion of variance in the data explained by the class signal. See below for a more detailed explanation of what the function does.

Usage

```
pvcam(xba, batch, y, threshold = 0.6)
```

Arguments

`xba`	matrix. The covariate matrix, raw or after batch effect adjustment. observations in rows, variables in columns.
`batch`	factor. Batch variable. Currently has to have levels: '1', '2', '3' and so on.
`y`	factor. Binary target variable. Currently has to have levels '1' and '2'.
`threshold`	numeric. Minimal proportion of variance explained by the principal components used.

Details

In PVCA, first principal component analysis is performed on the `n x n` covariance matrix between the observations. Then, using a random effects model the principal components are regressed on arbitrary factors of variability, such as "batch" and "(phenotype) class". Ultimately, estimated proportions of variance induced by each factor and that of the residual variance are obtained. In `pvcam` the factors included into the model are: "batch", "class" and the interaction of these two into. The metric calculated by `pvcam` is the proportion of variance explained by "class".

`pvcam` uses a slightly altered version of the function `pvcaBatchAssess()` from the Bioconductor package `pvca`. The latter was altered to take the covariate data as a `matrix` instead of as an object of class `ExpressionSet`.

Value

Value of the metric

Note

Higher values of this metric indicate a better preservation or exposure, respectively, of the biological signal of interest.

Author(s)

Roman Hornung

References

Li, J., Bushel, P., Chu, T.-M., Wolfinger, R.D. (2009) Principal variance components analysis: Estimating batch effects in microarray gene expression data. In: Scherer, A. (ed) Batch Effects and Noise in Microarray Experiments: Sources and Solutions, John Wiley & Sons, Chichester, UK.

Examples

```
data(autism)

Xadj <- ba(x=X, y=y, batch=batch, method = "combat")$xadj

pvcam(xba = X, batch = batch, y = y)
pvcam(xba = Xadj, batch = batch, y = y)
```

---

qunormaddon — *Addon quantile normalization using “documentation by value” (Kostka & Spang, 2008)*

Description

Performs addon quantile normalization by using documentation by value (Kostka & Spang, 2008).

Usage

```
qunormaddon(params, x)
```

Arguments

`params`	object of class `qunormtrain`. Contains parameters necessary for addon quantile normalization.
`x`	matrix. The covariate matrix of the new data. Observations in rows, variables in columns.

Details

This function uses code from the off-CRAN package `docval`, version 1.0.

Value

The covariate matrix of the test data after addon quantile normalization. Observations in rows, variables in columns.

Author(s)

Roman Hornung

References

Kostka, D., Spang, R. (2008). Microarray based diagnosis profits from better documentation of gene expression signatures. PLoS Computational Biology, 4(2), e22.

Examples

```
data(autism)
Xtrain <- X[batch==1,]
Xtest <- X[batch==2,]

params <- qunormtrain(x=Xtrain)

Xtrainnorm <- params$xnorm

Xtestaddonnorm <- qunormaddon(params, x=Xtest)
```

`qunormtrain` *Quantile normalization with "documentation by value" (Kostka & Spang, 2008)*

Description

Performs quantile normalization for a covariate matrix and returns the normalized dataset together with information necessary for addon quantile normalization (Kostka & Spang, 2008) using `qunormaddon`.

Usage

```
qunormtrain(x)
```

Arguments

`x`	matrix. The covariate matrix. observations in rows, variables in columns.

Details

This function uses code from the off-CRAN package `docval`, version 1.0.

Value

`qunormtrain` returns an object of class `qunormtrain`. An object of class "`qunormtrain`" is a list containing the following components:

`xnorm`	matrix of quantile normalized (training) data. Observations in rows, variables in columns.
`mqnts`	Vector of length ncol(xnorm). Averages of sorted values, with averages taken across observations.

Author(s)

Roman Hornung

References

Kostka, D., Spang, R. (2008). Microarray based diagnosis profits from better documentation of gene expression signatures. PLoS Computational Biology, 4(2), e22.

Examples

```
data(autism)
Xtrain <- X[batch==1,]

params <- qunormtrain(x=Xtrain)

Xtrainnorm <- params$xnorm
```

`ratioa` *Batch effect adjustment using Ratio-A*

Description

Performs batch effect adjustment using Ratio-A. Here, the variable values are divided by the batch-specific arithmetic mean of the corresponding variable.

Usage

```
ratioa(x, batch)
```

Arguments

`x`	matrix. The covariate matrix. Observations in rows, variables in columns.
`batch`	factor. Batch variable. Currently has to have levels: '1', '2', '3' and so on.

Value

`ratioa` returns an object of class `ratioa`. An object of class "`ratioa`" is a list containing the following components:

`xadj`	matrix of adjusted (training) data
`nbatches`	number of batches
`batch`	batch variable

Author(s)

Roman Hornung

References

Luo, J., Schumacher, M., Scherer, A., Sanoudou, D., Megherbi, D., Davison, T., Shi, T., Tong, W., Shi, L., Hong, H., Zhao, C., Elloumi, F., Shi, W., Thomas, R., Lin, S., Tillinghast, G., Liu, G., Zhou, Y., Herman, D., Li, Y., Deng, Y., Fang, H., Bushel, P., Woods, M., Zhang, J. (2010) A comparison of batch effect removal methods for enhancement of prediction performance using maqc-ii microarray gene expression data. The Pharmacogenomics Journal, 10, 278-291.

Examples

```
data(autism)

params <- ratioa(x=X, batch=batch)
```

`ratioaaddon` *Addon batch effect adjustment for Ratio-A*

Description

Performs addon batch effect adjustment for Ratio-A: 1) takes the output of `ratioa` applied to a training data set together with new batch data; 2) checks whether the training data was also adjusted using Ratio-A and whether the same number of variables is present in training and new data; 3) performs Ratio-A on the new batch data.

Usage

```
ratioaaddon(params, x, batch)
```

Arguments

`params`	object of class `ratioa`.
`x`	matrix. The covariate matrix of the new data. Observations in rows, variables in columns.
`batch`	factor. Batch variable of the new data. Currently has to have levels: ’1’, ’2’, ’3’ and so on.

Value

The adjusted covariate matrix of the test data.

Note

Because Ratio-A is performed "batch by batch" the "addon procedure" for Ratio-A consists of plain Ratio-A on the new test batches.

Author(s)

Roman Hornung

References

Luo, J., Schumacher, M., Scherer, A., Sanoudou, D., Megherbi, D., Davison, T., Shi, T., Tong, W., Shi, L., Hong, H., Zhao, C., Elloumi, F., Shi, W., Thomas, R., Lin, S., Tillinghast, G., Liu, G., Zhou, Y., Herman, D., Li, Y., Deng, Y., Fang, H., Bushel, P., Woods, M., Zhang, J. (2010) A comparison of batch effect removal methods for enhancement of prediction performance using maqc-ii microarray gene expression data. The Pharmacogenomics Journal, 10, 278-291.

Hornung, R., Boulesteix, A.-L., Causeur, D. (2016) Combining location-and-scale batch effect adjustment with data cleaning by latent factor adjustment. BMC Bioinformatics 17:27.

Examples

```
data(autism)

trainind <- which(batch %in% c(1,2))

Xtrain <- X[trainind,]
ytrain <- y[trainind]
batchtrain <- factor(as.numeric(batch[trainind]), levels=c(1,2))


testind <- which(batch %in% c(3,4))

Xtest <- X[testind,]
ytest <- y[testind]

batchtest <- as.numeric(batch[testind])
batchtest[batchtest==3] <- 1
batchtest[batchtest==4] <- 2
batchtest <- factor(batchtest, levels=c(1,2))


params <- ratioa(x=Xtrain, batch=batchtrain)

Xtestaddon <- ratioaaddon(params=params, x=Xtest, batch=batchtest)
```

`ratiog` *Batch effect adjustment using Ratio-G*

Description

Performs batch effect adjustment using Ratio-G. Here, the variable values are divided by the batch-specific geometric mean of the corresponding variable.

Usage

```
ratiog(x, batch)
```

Arguments

`x`	matrix. The covariate matrix. Observations in rows, variables in columns.
`batch`	factor. Batch variable. Currently has to have levels: '1', '2', '3' and so on.

Value

`ratiog` returns an object of class `ratiog`. An object of class "`ratiog`" is a list containing the following components:

`xadj`	matrix of adjusted (training) data
`nbatches`	number of batches
`batch`	batch variable

Author(s)

Roman Hornung

References

Luo, J., Schumacher, M., Scherer, A., Sanoudou, D., Megherbi, D., Davison, T., Shi, T., Tong, W., Shi, L., Hong, H., Zhao, C., Elloumi, F., Shi, W., Thomas, R., Lin, S., Tillinghast, G., Liu, G., Zhou, Y., Herman, D., Li, Y., Deng, Y., Fang, H., Bushel, P., Woods, M., Zhang, J. (2010) A comparison of batch effect removal methods for enhancement of prediction performance using maqc-ii microarray gene expression data. The Pharmacogenomics Journal, 10, 278-291.

Examples

```
data(autism)

params <- ratiog(x=X, batch=batch)
```

`ratiogaddon` *Addon batch effect adjustment for Ratio-G*

Description

Performs addon batch effect adjustment for Ratio-G: 1) takes the output of `ratiog` applied to a training data set together with new batch data; 2) checks whether the training data was also adjusted using Ratio-G and whether the same number of variables is present in training and new data; 3) performs Ratio-G on the new batch data.

Usage

```
ratiogaddon(params, x, batch)
```

Arguments

`params`	object of class `ratiog`.
`x`	matrix. The covariate matrix of the new data. Observations in rows, variables in columns.
`batch`	factor. Batch variable of the new data. Currently has to have levels: '1', '2', '3' and so on.

Value

The adjusted covariate matrix of the test data.

Note

Because Ratio-G is performed "batch by batch" the "addon procedure" for Ratio-G consists of plain Ratio-G on the new test batches.

Author(s)

Roman Hornung

References

Luo, J., Schumacher, M., Scherer, A., Sanoudou, D., Megherbi, D., Davison, T., Shi, T., Tong, W., Shi, L., Hong, H., Zhao, C., Elloumi, F., Shi, W., Thomas, R., Lin, S., Tillinghast, G., Liu, G., Zhou, Y., Herman, D., Li, Y., Deng, Y., Fang, H., Bushel, P., Woods, M., Zhang, J. (2010) A comparison of batch effect removal methods for enhancement of prediction performance using maqc-ii microarray gene expression data. The Pharmacogenomics Journal, 10, 278-291.

Hornung, R., Boulesteix, A.-L., Causeur, D. (2016) Combining location-and-scale batch effect adjustment with data cleaning by latent factor adjustment. BMC Bioinformatics 17:27.

Examples

```
data(autism)

trainind <- which(batch %in% c(1,2))

Xtrain <- X[trainind,]
ytrain <- y[trainind]
batchtrain <- factor(as.numeric(batch[trainind]), levels=c(1,2))


testind <- which(batch %in% c(3,4))

Xtest <- X[testind,]
ytest <- y[testind]

batchtest <- as.numeric(batch[testind])
batchtest[batchtest==3] <- 1
batchtest[batchtest==4] <- 2
batchtest <- factor(batchtest, levels=c(1,2))
```

```
params <- ratiog(x=Xtrain, batch=batchtrain)

Xtestaddon <- ratiogaddon(params=params, x=Xtest, batch=batchtest)
```

`rmaaddon` *Addon RMA normalization using "documentation by value" (Kostka & Spang, 2008)*

Description

Performs RMA with addon quantile normalization by using documentation by value (Kostka & Spang, 2008).

Usage

```
rmaaddon(params, affybatchtest)
```

Arguments

`params`	object of class `rmatrain`. The normalized training data together with the information necesssary for addon normalization.
`affybatchtest`	object of class `AffyBatch`, that is Affymetrix GeneChip probe level data. Test data to be used for addon normalization.

Details

This function uses code from the off-CRAN package `docval`, version 1.0.

Value

The covariate matrix of the test data after addon normalization. Observations in rows, variables in columns.

Author(s)

Roman Hornung

References

Kostka, D., Spang, R. (2008). Microarray based diagnosis profits from better documentation of gene expression signatures. PLoS Computational Biology, 4(2), e22.

Examples

```
## Not run:

# Read in example data from ArrayExpress-webpage:

library("ArrayExpress")

expFiles <- getAE("E-GEOD-62837", path = tempdir(), type = "raw")

rawfiles <- file.path(tempdir(), expFiles$rawFiles)

library("affy")
# Training data:
affybatchtrain <- ReadAffy(filenames=rawfiles[1:3])
# Test data:
affybatchtest <- ReadAffy(filenames=rawfiles[4:5])

try(file.remove(file.path(tempdir(), expFiles$rawFiles)))
try(file.remove(file.path(tempdir(), expFiles$processedFiles)))
try(file.remove(file.path(tempdir(), expFiles$sdrf)))
try(file.remove(file.path(tempdir(), expFiles$idf)))
try(file.remove(file.path(tempdir(), expFiles$adf)))
try(file.remove(file.path(tempdir(), expFiles$rawArchive)))
try(file.remove(file.path(tempdir(), expFiles$processedArchive)))


# RMA normalization with documentation by value:
rmaparams <- rmatrain(affybatchtrain)
Xtrainnorm <- rmaparams$xnorm
dim(Xtrainnorm)

# RMA addon normalization:
Xtestaddonnorm <- rmaaddon(rmaparams, affybatchtest)
dim(Xtestaddonnorm)

## End(Not run)
```

`rmatrain` *RMA normalization with “documentation by value” (Kostka & Spang, 2008)*

Description

Performs RMA normalization and returns the normalized dataset together with information necessary for addon RMA normalization (Kostka & Spang, 2008) using `rmaaddon`.

Usage

```
rmatrain(affybatchtrain)
```

Arguments

`affybatchtrain`	object of class `AffyBatch`. Affymetrix GeneChip probe level data.

Details

This function uses code from the off-CRAN package `docval`, version 1.0.

Value

`rmatrain` returns an object of class `rmatrain`. An object of class "`rmatrain`" is a list containing the following components:

`xnorm`	matrix of RMA normalized (training) data. Observations in rows, variables in columns.
`rmadoc, sumdoc.rma, nfeature`	information necessary for addon RMA normalization

Author(s)

Roman Hornung

References

Kostka, D., Spang, R. (2008). Microarray based diagnosis profits from better documentation of gene expression signatures. PLoS Computational Biology, 4(2), e22.

Examples

```
## Not run:

# Read in example data from ArrayExpress-webpage:

library("ArrayExpress")

expFiles <- getAE("E-GEOD-62837", path = tempdir(), type = "raw")

rawfiles <- file.path(tempdir(), expFiles$rawFiles)

library("affy")
# Training data:
affybatchtrain <- ReadAffy(filenames=rawfiles[1:3])

try(file.remove(file.path(tempdir(), expFiles$rawFiles)))
try(file.remove(file.path(tempdir(), expFiles$processedFiles)))
try(file.remove(file.path(tempdir(), expFiles$sdrf)))
try(file.remove(file.path(tempdir(), expFiles$idf)))
try(file.remove(file.path(tempdir(), expFiles$adf)))
try(file.remove(file.path(tempdir(), expFiles$rawArchive)))
try(file.remove(file.path(tempdir(), expFiles$processedArchive)))
```

```
# RMA normalization with documentation by value:
rmaparams <- rmatrain(affybatchtrain)
Xtrainnorm <- rmaparams$xnorm
dim(Xtrainnorm)

## End(Not run)
```

---

sepscore — *Separation score as described in Hornung et al. (2016)*

Description

This metric described in Hornung et al. (2016) was derived from the mixture score presented in Lazar et al. (2012). In contrast to the mixture score the separation score does not measure the degree of mixing but the degree of separation between the batches. Moreover it is less dependent on the relative sizes of the involved batches.

Usage

```
sepscore(xba, batch, k = 10)
```

Arguments

`xba`	matrix. The covariate matrix, raw or after batch effect adjustment. observations in rows, variables in columns.
`batch`	factor. Batch variable. Currently has to have levels: '1', '2', '3' and so on.
`k`	integer. Number of nearest neighbors.

Details

For two batches j and j* (see next paragraph for the case with more batches): 1) for each observation in batch j its k nearest neighbours in both batches j and j* simultaneously with respect to the euclidean distance are determined. Here, the proportion of those of these nearest neighbours, which belong to batch j* is calculated; 2) the average - denoted as MS_j - is taken over the thus obtained n_j proportions. This value is the mixture score as in Lazar et al. (2012); 3) to obtain a measure for the separation of the two batches the absolute difference between MS_j and its value expected in the absence of batch effects is taken: |MS_j - n_j* /(n_j + n_j* - 1)|; 4) the separation score is defined as the simple average of the latter quantity and the corresponding quantity when the roles of j and j* are switched. If the supplied number `k` of nearest neighbours is larger than n_j + n_j*, `k` is set to n_j + n_j* - 1 internally.

For more than two batches: 1) for all possible pairs of batches: calculate the metric as described above; 2) calculate the weighted average of the values in 1) with weights proportional to the sum of the sample sizes in the two respective batches.

Value

Value of the metric

Note

The smaller the values of this metric, the better.

Author(s)

Roman Hornung

References

Hornung, R., Boulesteix, A.-L., Causeur, D. (2016) Combining location-and-scale batch effect adjustment with data cleaning by latent factor adjustment. BMC Bioinformatics 17:27.

Lazar, C., Meganck, S., Taminau, J., Steenhoff, D., Coletta, A., Molter,C., Weiss-Solís, D. Y., Duque, R., Bersini, H., Nowé, A. (2012) Batch effect removal methods for microarray gene expression data integration: a survey. Briefings in Bioinformatics, 14(4), 469-490.

Examples

```
data(autism)

# Random subset of 150 variables:
set.seed(1234)
Xsub <- X[,sample(1:ncol(X), size=150)]

# In cases of batches with more than 20 observations
# select 20 observations at random:
subinds <- unlist(sapply(1:length(levels(batch)), function(x) {
  indbatch <- which(batch==x)
  if(length(indbatch) > 20)
    indbatch <- sort(sample(indbatch, size=20))
  indbatch
}))
Xsub <- Xsub[subinds,]
batchsub <- batch[subinds]
ysub <- y[subinds]



sepscore(xba=Xsub, batch=batchsub, k=5)

params <- ba(x=Xsub, y=ysub, batch=batchsub, method = "ratiog")
Xsubadj <- params$xadj

sepscore(xba=Xsubadj, batch=batchsub, k=5)

params <- ba(x=Xsub, y=ysub, batch=batchsub, method = "combat")
Xsubadj <- params$xadj

sepscore(xba=Xsubadj, batch=batchsub, k=5)
```

skewdiv *Skewness divergence score*

Description

This metric presented in Shabalin et al. (2008) is concerned with the dissimilarity across batches of the skewnesses of the observation-wise empirical distributions of the data.

Usage

```
skewdiv(xba, batch)
```

Arguments

`xba`	matrix. The covariate matrix, raw or after batch effect adjustment. observations in rows, variables in columns.
`batch`	factor. Batch variable. Currently has to have levels: '1', '2', '3' and so on.

Details

For two batches j and j* (see next paragraph for the case with more batches): 1) for each observation calculate the difference between the mean and the median of the data as a measure for the skewness of the distribution of the variable values; 2) determine the area between the two batch-wise empirical cumulative density functions of the values out of 1). The value obtained in 2) can be regarded as a measure for the disparity of the batches with respect to the skewness of the observation-wise empirical distributions.

For more than two batches: 1) for all possible pairs of batches: calculate the metric as described above; 2) calculate the weighted average of the values in 1) with weights proportional to the sum of the sample sizes in the two respective batches.

The variables are standardized before the calculation to make the metric independent of scale.

Value

Value of the metric

Note

The smaller the values of this metric, the better.

Author(s)

Roman Hornung

References

Shabalin, A. A., Tjelmeland, H., Fan, C., Perou, C. M., Nobel, A. B. (2008) Merging two gene-expression studies via cross-platform normalization. Bioinformatics, 24(9), 1154-1160.

Examples

```
data(autism)

skewdiv(xba=X, batch=batch)

params <- ba(x=X, y=y, batch=batch, method = "ratiog")
Xadj <- params$xadj

skewdiv(xba=Xadj, batch=batch)

params <- ba(x=X, y=y, batch=batch, method = "combat")
Xadj <- params$xadj

skewdiv(xba=Xadj, batch=batch)
```

`standardize` *Batch effect adjustment by standardization*

Description

Performs batch effect adjustment by standardizing the variables within batches to have zero mean and variance one.

Usage

```
standardize(x, batch)
```

Arguments

`x`	matrix. The covariate matrix. Observations in rows, variables in columns.
`batch`	factor. Batch variable. Currently has to have levels: '1', '2', '3' and so on.

Value

`standardize` returns an object of class `standardize`. An object of class "`standardize`" is a list containing the following components:

`xadj`	matrix of adjusted (training) data
`nbatches`	number of batches
`batch`	batch variable

Author(s)

Roman Hornung

Examples

```
data(autism)

params <- standardize(x=X, batch=batch)
```

`standardizeaddon` *Addon batch effect adjustment for standardization*

Description

Performs addon batch effect adjustment for standardization: 1) takes the output of `standardize` applied to a training data set together with new batch data; 2) checks whether the training data was also adjusted using standardization and whether the same number of variables is present in training and new data; 3) performs standardization on the new batch data.

Usage

```
standardizeaddon(params, x, batch)
```

Arguments

`params`	object of class `standardize`.
`x`	matrix. The covariate matrix of the new data. Observations in rows, variables in columns.
`batch`	factor. Batch variable of the new data. Currently has to have levels: '1', '2', '3' and so on.

Value

The adjusted covariate matrix of the test data.

Note

It is **not recommended** to perform standardization in cross-study prediction settings, because for some classifiers the prediction performance can be strongly impaired by this method. Given a not too small test set, the following methods are recommended (Hornung et al., 2016a): `combatba`, `meancenter`, `ratioa`, `ratiog`.

Because standardization is performed "batch by batch" the "addon procedure" for standardization consists of plain standardization on the new test batches.

Author(s)

Roman Hornung

References

Hornung, R., Causeur, D., Bernau, C., Boulesteix, A.-L. (2016a). Improving cross-study prediction through addon batch effect adjustment and addon normalization. Technical Report, Department of Statistics, LMU.

Hornung, R., Boulesteix, A.-L., Causeur, D. (2016b) Combining location-and-scale batch effect adjustment with data cleaning by latent factor adjustment. BMC Bioinformatics 17:27.

Examples

```
data(autism)

trainind <- which(batch %in% c(1,2))

Xtrain <- X[trainind,]
ytrain <- y[trainind]
batchtrain <- factor(as.numeric(batch[trainind]), levels=c(1,2))


testind <- which(batch %in% c(3,4))

Xtest <- X[testind,]
ytest <- y[testind]

batchtest <- as.numeric(batch[testind])
batchtest[batchtest==3] <- 1
batchtest[batchtest==4] <- 2
batchtest <- factor(batchtest, levels=c(1,2))


params <- standardize(x=Xtrain, batch=batchtrain)

Xtestaddon <- standardizeaddon(params=params, x=Xtest, batch=batchtest)
```

`svaba` *Batch effect adjustment using SVA*

Description

Performs batch effect adjustment using Surrogate Variable Analysis (SVA) and additionally returns information necessary for addon batch effect adjustment with frozen SVA.

Usage

```
svaba(x, y, batch, nbf = NULL, algorithm = "fast")
```

Arguments

`x`	matrix. The covariate matrix. Observations in rows, variables in columns.
`y`	factor. Binary target variable. Currently has to have levels ’1’ and ’2’.
`batch`	factor. Batch variable. Currently has to have levels: ’1’, ’2’, ’3’ and so on.
`nbf`	integer. Number of latent factors to estimate.
`algorithm`	character. If `method = "fast"` the "approximate fSVA algorithm" will be used in frozen SVA. If `method = "exact"` the "exact fSVA algorithm" will be used. See Parker et al. (2014).

Details

This is essentially a wrapper function of the function `sva()` from the Bioconductor package of the same name.

Value

`svaba` returns an object of class `svatrain`. An object of class "`svatrain`" is a list containing the following components:

`xadj`	matrix of adjusted (training) data
`xtrain`	the unadjusted covariate matrix. Used in frozen SVA.
`ytrain`	binary target variable. Used in frozen SVA.
`svobj`	output of the function `sva()`. Used in frozen SVA.
`algorithm`	algorithm to use in frozen SVA
`nbatches`	number of batches
`batch`	batch variable

Author(s)

Roman Hornung

References

Leek, J. T., Storey, J. D. (2007) Capturing Heterogeneity in Gene Expression Studies by Surrogate Variable Analysis. PLoS Genetics, 3, 1724–1735.

Parker, H. S., Bravo, H. C., Leek, J. T. (2014) Removing batch effects for prediction problems with frozen surrogate variable analysis. PeerJ, 2, e561.

Examples

```
data(autism)

# Random subset of 150 variables:
set.seed(1234)
Xsub <- X[,sample(1:ncol(X), size=150)]

# In cases of batches with more than 20 observations
# select 20 observations at random:
subinds <- unlist(sapply(1:length(levels(batch)), function(x) {
  indbatch <- which(batch==x)
  if(length(indbatch) > 20)
    indbatch <- sort(sample(indbatch, size=20))
  indbatch
}))
Xsub <- Xsub[subinds,]
batchsub <- batch[subinds]
ysub <- y[subinds]
```

```
params <- svaba(x=Xsub, y=ysub, batch=batchsub)
```

`svabaaddon` *Addon batch effect adjustment using frozen SVA*

Description

Performs addon batch effect adjustment using frozen SVA. Takes the output of performing `svaba` on a training data set and new batch data and correspondingly adjusts the test data to better match the adjusted training data according to the SVA model.

Usage

```
svabaaddon(params, x)
```

Arguments

`params`	object of class `svatrain`. Contains parameters necessary for addon batch effect adjustment with frozen SVA.
`x`	matrix. The covariate matrix of the new data. Observations in rows, variables in columns.

Value

The adjusted covariate matrix of the test data.

Note

It is **not recommended** to perform frozen SVA in cross-study prediction settings, because it assumes similarity between training and test set and has been observed to (strongly) impair prediction performance in cases where this assumption is not given. Given a not too small test set, the following methods are recommended (Hornung et al., 2016): `combatba`, `meancenter`, `ratioa`, `ratiog`.

Author(s)

Roman Hornung

References

Leek, J. T., Storey, J. D. (2007) Capturing Heterogeneity in Gene Expression Studies by Surrogate Variable Analysis. PLoS Genetics, 3, 1724–1735.

Parker, H. S., Bravo, H. C., Leek, J. T. (2014) Removing batch effects for prediction problems with frozen surrogate variable analysis. PeerJ, 2, e561.

Hornung, R., Causeur, D., Bernau, C., Boulesteix, A.-L. (2016). Improving cross-study prediction through addon batch effect adjustment and addon normalization. Technical Report, Department of Statistics, LMU.

Examples

```
data(autism)

# Random subset of 150 variables:
set.seed(1234)
Xsub <- X[,sample(1:ncol(X), size=150)]

# In cases of batches with more than 20 observations
# select 20 observations at random:
subinds <- unlist(sapply(1:length(levels(batch)), function(x) {
  indbatch <- which(batch==x)
  if(length(indbatch) > 20)
    indbatch <- sort(sample(indbatch, size=20))
  indbatch
}))
Xsub <- Xsub[subinds,]
batchsub <- batch[subinds]
ysub <- y[subinds]



trainind <- which(batchsub %in% c(1,2))

Xsubtrain <- Xsub[trainind,]
ysubtrain <- ysub[trainind]
batchsubtrain <- factor(as.numeric(batchsub[trainind]), levels=c(1,2))


testind <- which(batchsub %in% c(3,4))

Xsubtest <- Xsub[testind,]
ysubtest <- ysub[testind]

batchsubtest <- as.numeric(batchsub[testind])
batchsubtest[batchsubtest==3] <- 1
batchsubtest[batchsubtest==4] <- 2
batchsubtest <- factor(batchsubtest, levels=c(1,2))



params <- svaba(x=Xsubtrain, y=ysubtrain, batch=batchsubtrain)

Xsubtestaddon <- svabaaddon(params, x=Xsubtest)
```

X *Covariate matrix of dataset* autism

Description

See dataset autism

`y` *Target variable of dataset* `autism`

Description

See dataset `autism`

Index