Package: gemR (via r-universe)

October 19, 2026

Encoding UTF-8

Type Package

Title General Effect Modelling

Version 1.2.2

Date 2025-09-04

Description Two-step modeling with separation of sources of variation through analysis of variance and subsequent multivariate modeling through a range of unsupervised and supervised statistical methods. Separation can focus on removal of interfering effects or isolation of effects of interest. EF Mosleth et al. (2021) <doi:10.1038/s41598-021-82388-w> and EF Mosleth et al. (2020) <doi:10.1016/B978-0-12-409547-2.14882-6>.

Depends R (>= 3.5.0)

Imports ggplot2, scales, gridExtra, mixlm, pls, plsVarSel, HDANOVA (>= 0.8.4), lme4, pracma

Suggests glmnet, neuralnet

License GPL

LazyData TRUE

RoxygenNote 7.3.2

Config/pak/sysreqs cmake make libicu-dev

Repository https://khliland.r-universe.dev

Date/Publication 2025-09-05 12:42:40 UTC

RemoteUrl https://github.com/khliland/gemr

RemoteRef HEAD

RemoteSha e3ba228e5c6a4747f25bdccd151a4ef30f6f5719

Contents

confints 2
Diabetes 4

elastic 5
GEM 6
knock.in 9
Lactobacillus 10
MS 11
neuralnet 12
pca 13
plot.GEMpls 14
pls.GEM 15
sca 17

Index **18**

`confints` *Confidence Intervals of Effect Differences*

Description

Confidence Intervals of Effect Differences

Usage

```
confints(X1, ...)

## Default S3 method:
confints(X1, X2, confidence = 0.95, df.used = 0, ...)

## S3 method for class 'GEM'
confints(
  X1,
  factor = 1,
  levels = c(1, 2),
  confidence = 0.95,
  df.used = X1$df.used,
  ...
)

## S3 method for class 'confints'
plot(
  x,
  y,
  xlab = "",
  ylab = "values",
  sorted = TRUE,
  labels = FALSE,
  nonZero = FALSE,
  xlim = NULL,
  ylim = NULL,
```

```
  text.pt = 12,
  ...
)
```

Arguments

X1	data.frame containing first effect or GEM object for ER matrix based intervals.
...	Further arguments to qplot.
X2	data.frame containing second effect.
confidence	Level of confidence, default = 0.95.
df.used	Optional argument indicating how many degrees of freedom have been consumed during deflation. Default = 0.
factor	(character or numeric) indicating which factor to use in ER based intervals (defaul = 1).
levels	vector (character or numeric) indicating which factor levels to. use in ER based intervals (default = c(1,2)).
x	Object of class confint.
y	Not used.
xlab	X label (character)
ylab	Y label (character)
sorted	Logical indicating if intervals should be sorted according to their mean values, or a vector of indices/labels to sort by.
labels	Logical indicating if sample labels should be used on x axis.
nonZero	Logical indicating if intervals are required not to include zero.
xlim	Limits of the horizontal scale.
ylim	Limits of the vertical scale.
text.pt	Size scaling of text in the plot (default = 16).

Value

An object of class confints, which holds the information needed to perform statistics or plot the confidence intervals is returned from confints. The plotting routine returns a ggplot structure for plotting.

See Also

Analyses using GEM: elastic, pca, sca, neuralnet, pls.

Examples

```
data(MS)
# Subset to reduce runtime in example
MS$proteins <- MS$proteins[,20:70]

# Compare MS and non-MS patients within group 1
```

```
conf <- with(MS, confints(proteins[MS == "yes" & group == 1,],
                          proteins[MS == "no"  & group == 1,]))
p1 <- plot(conf)
p2 <- plot(conf, nonZero = TRUE) # Only intervals without 0.
grid.arrange(p1,p2)

# Comparison repeated but based on ER matrices
gem <- GEM(proteins ~ MS * group, data = MS)
print(effs <- colnames(gem$symbolicDesign)) # Inspect factor names
confGEM <- confints(gem, factor=effs[3], levels=c("yes.1","no.1"))
p1g <- plot(confGEM)
p2g <- plot(confGEM, nonZero = TRUE) # Only intervals without 0.
grid.arrange(p1g,p2g)

# Shorter plot with labels
confShort <- conf[1:10,]
p1 <- plot(confShort, labels = TRUE)
p2 <- plot(confShort, labels = TRUE, nonZero = TRUE)
grid.arrange(p1,p2)
```

`Diabetes` *Diabetes data*

Description

A `data.frame` with a design and transcriptomic data.

Usage

```
data(Diabetes)
```

Details

Clinical study on humans was performed as a 2-way factorial design with two factors both on two levels: bariatric surgery on two levels (before and after the bariatric surgery) and type 2 diabetes (T2D) on two levels (with and without T2D). There were 8 patients without T2D and 7 with T2D. It was discovered that the patients with T2D would be separated in two groups: 3 patients in the group called T2D1 and 4 patients in the group called T2D2. The experiment can therefore also analysed as 2 way factorial design where the disease factor is on three levels. All patients were obese before bariatric surgery (BMI >45). Transcriptome in the subcutaneous adipose tissue were obtained before and one year after bariatric surgery.

Author(s)

Ellen Færgestad Mosleth

References

Dankel et al. 2010. Switch from Stress Response to Homeobox Transcription Factors in Adipose Tissue After Profound Fat Loss. Plos One 5.

Examples

```
data(Diabetes)
str(Diabetes)
```

`elastic` *Elastic-net modeling of GEM objects.*

Description

Elastic-net modeling of GEM objects.

Usage

```
elastic(gem, ...)

## S3 method for class 'GEM'
elastic(
  gem,
  effect,
  alpha = 0.5,
  newdata = NULL,
  validation,
  segments = NULL,
  measure = measure,
  family = family,
  ...
)
```

Arguments

`gem`	Object of class `GEM`.
`...`	Additional arguments for `glmnet`.
`effect`	The effect to be used as response.
`alpha`	The elasticnet mixing parameter.
`newdata`	Optional new data matrix for prediction.
`validation`	Optional validation parameters.
`segments`	number of segments or list of segments (optional)
`measure`	Type of performance summary, default = 'class' (see `glmnet`)
`family`	Type of model response, default = 'multinomial'.

Value

An object of class `GEMglmnet, cv.glmnet, list` containing the fitted Elastic-net model, classifications/predictions and data.

See Also

Analyses using GEM: pca, sca, neuralnet, pls. Confidence interval plots: confints. Convenience knock-in and knock-out of effects: knock.in.

Examples

```
## Multiple Sclerosis data
data(MS, package = "gemR")
# Subset to reduce runtime in example
MS$proteins <- MS$proteins[,20:70]

gem <- GEM(proteins ~ MS * group, data = MS)
elasticMod <- elastic(gem, 'MS', validation = "CV")
sum(elasticMod$classes == MS$MS)
plot(elasticMod)            # Model fit
plot(elasticMod$glmnet.fit) # Coefficient trajectories

# Select all proteins with non-zeros coefficients
coefs     <- coef(elasticMod)
(selected <- names(which(coefs[,1] != 0)))

 # Time consuming due to many variables
  ## Diabetes data
  data(Diabetes, package = "gemR")
  gem.Dia <- GEM(transcriptome ~ surgery * T2D, data = Diabetes)
  elasticMod <- elastic(gem.Dia, 'T2D', validation = "LOO")
```

GEM *General Effect Modelling*

Description

General Effect Modelling

Usage

```
GEM(formula, data, contrasts = "contr.sum", add_residuals = TRUE, ...)

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

## S3 method for class 'GEM'
plot(
  x,
  y = 1,
  what = "raw",
  col = NULL,
```

```
  pch = NULL,
  model.line = (what %in% c("raw")),
  ylim = NULL,
  ylab = "",
  xlab = "",
  main = NULL,
  ...
)

tableGEM(object, variable)

## S3 method for class 'GEM'
summary(object, extended = TRUE, df = FALSE, ...)

## S3 method for class 'summary.GEM'
print(x, digits = 2, ...)
```

Arguments

formula	a model formula specifying features and effects.
data	a data.frame containing response variables (features) and design factors or other groupings/continuous variables.
contrasts	a character containing the primary contrasts for use with mixlm::lm (default = "contr.sum").
add_residuals	Logical indicating if residuals should be added to the ER values (default = TRUE).
...	Additional arguments to plot
x	Object of class GEM.
y	Response name or number.
what	What part of GEM to plot; raw data (default), fits, residuals or a named model effect (can be combined with 'effect', see Examples).
col	Color of points, defaults to grouping. Usually set to a factor name or a column name in the input data with custom colours.
pch	Plot character of points, defaults to 1. Usually set to a factor name or a column name in the input data with custom symbols
model.line	Include line indicating estimates, default = TRUE. Can be an effect name.
ylim	Y axis limits (numeric, but defaults to NULL)
ylab	Y label (character)
xlab	X label (character)
main	Main title, defaults to y with description from what.
object	GEM object.
variable	Numeric for selecting a variable for extraction.
extended	Extended output in summary (default = TRUE).
df	Show degrees of freedom in summary (default = FALSE).
digits	integer number of digits for printing.

Value

GEM returns an object of class GEM containing effects, ER values (effect + residuals), fitted values, residuals, features, coefficients, dummy design, symbolic design, dimensions, highest level interaction and feature names.

References

* Mosleth et al. (2021) Cerebrospinal fluid proteome shows disrupted neuronal development in multiple sclerosis. Scientific Report, 11,4087. <doi:10.1038/s41598-021-82388-w>

* E.F. Mosleth et al. (2020). Comprehensive Chemometrics, 2nd edition; Brown, S., Tauler, R., & Walczak, B. (Eds.). Chapter 4.22. Analysis of Megavariate Data in Functional Omics. Elsevier. <doi:10.1016/B978-0-12-409547-2.14882-6>

See Also

Analyses using GEM: elastic, pca, sca, neuralnet, pls. Confidence interval plots: confints. Convenience knock-in and knock-out of effects: knock.in.

Examples

```
## Multiple Sclerosis
data(MS, package = "gemR")
# Subset to reduce runtime in example
MS$proteins <- MS$proteins[,20:70]

gem <- GEM(proteins ~ group * MS, data = MS)
print(gem)
summary(gem)                                    # Summary of GEM
plot(gem)                                       # Raw data, first feature
plot(gem,2)                                     # Raw data, numbered feature
plot(gem,'Q76L83', col='MS', pch='group')       # Selected colour and plot character
plot(gem,'Q76L83', what='effect MS',
     model.line='effect group')                # Comparison of factors (points and lines)
print(effs <- colnames(gem$symbolicDesign))     # Inspect factor names
eeffs <- paste0("effect ", effs)
# Example compound plot
old.par <- par(mfrow = c(3,3), mar = c(2,4,4,1))
plot(gem,'Q76L83')                          # Raw data, named feature
plot(gem,'Q76L83', what='fits')             # Fitted values
plot(gem,'Q76L83', what='residuals')        # Residuals
plot(gem,'Q76L83', what=eeffs[1])           # Effect levels
plot(gem,'Q76L83', what=eeffs[2])           # ----||----
plot(gem,'Q76L83', what=eeffs[3])           # ----||----
plot(gem,'Q76L83', what=effs[1])            # ER values
plot(gem,'Q76L83', what=effs[2])            # --------||---------
plot(gem,'Q76L83', what=effs[3])            # --------||---------
par(old.par)

# Complete overview of GEM
tab <- tableGEM(gem, 1)
```

```
# In general there can be more than two, effects, more than two levels, and continuous effects:
MS$three <- factor(c(rep(1:3,33),1:2))
gem3    <- GEM(proteins ~ MS * group + three, data = MS)


## Candy assessment
data(candies, package = "HDANOVA")
gemC <- GEM(assessment ~ assessor*candy, data=candies)

# Permutation testing
gemC <- permutation(gemC)
summary(gemC)

# GEM-SCA with ellipsoids in score plots
gemSCA <- sca(gemC)
scoreplot(gemSCA, factor="candy", ellipsoids="confidence")

# GEM-PCA with group colours
gemPCA <- pca(gemC)
scoreplot(gemPCA, factor="candy",
  gr.col=gemPCA$symbolicDesign$candy)


## Lactobacillus
data(Lactobacillus, package = "gemR")
# Subset to reduce runtime in example
Lactobacillus$proteome <- Lactobacillus$proteome[,50:100]

gemLac <- GEM(proteome ~ strain * growthrate, data = Lactobacillus)
print(gemLac)
plot(gemLac)                            # Raw data, first feature
plot(gemLac,2)                          # Raw data, numbered feature
plot(gemLac,'P.LSA0316', col='strain',
    pch='growthrate')                   # Selected colour and plot character
plot(gemLac,'P.LSA0316', what='strain',
    model.line='growthrate')            # Selected model.line


 # Don't run this example, it takes too long
  ## Diabetes
  data(Diabetes, package = "gemR")
  gemDia <- GEM(transcriptome ~ surgery * T2D, data = Diabetes)
  print(gemDia)
  plot(gemDia)                            # Raw data, first feature
  plot(gemDia,2)                          # Raw data, numbered feature
  plot(gemDia,'ILMN_1720829', col='surgery',
      pch='T2D')                          # Selected colour and plot character
```

---

knock.in *Knock-in and knock-out of effect(s)*

Description

Convenience functions to apply to GEM objects to isolate/extract (knock-in) one or more effects (and possibly residuals) or to remove (knock-out) one or more effects.

Usage

```
knock.in(object, effect, residuals = TRUE)

knock.out(object, effect, residuals = TRUE)
```

Arguments

object	GEM object.
effect	Name or number of effect (character or numeric vector).
residuals	Logical indicating if residuals should be added (default = TRUE).

Value

A data.frame of ER values where effects have been knocked in (isolated/extracted from data) or knocked out (removed from data).

Examples

```
data(MS, package = "gemR")

# Subset to reduce runtime in example
MS$proteins <- MS$proteins[,20:70]

gem <- GEM(proteins ~ MS * group, data = MS)

# Extract interaction between 'MS' and 'group
ER.isolated <- knock.in(gem, 'MS:group')

# Remove main effect of 'group'
ER.cleaned <- knock.out(gem, 'group')
```

Lactobacillus *Lactobacillus data*

Description

A data.frame with a design and proteomic data, transcriptomic data and phenotypic data.

Usage

```
data(Lactobacillus)
```

Details

Experiment on Lactobacillus sakei was performed as a 2-way factorial design with two factors both on two levels: strain (L. sakei strains LS25 and 23K) (factor A) and growth condition (high and low glucose availability) (factor B) both on two levels, and their interaction term (factor AB). There were three biological replicates within each group. Transcriptome, proteome and end product profile (lactate, formate, acetate and ethanol) were observed.

Author(s)

Ellen Færgestad Mosleth

References

McLeod et al. 2017. Effects of glucose availability in Lactobacillus sakei; metabolic change and regulation of the proteome and transcriptome. Plos One 12, e0187542.

Examples

```
data(Lactobacillus)
str(Lactobacillus)
```

MS *Multiple Sclerosis data*

Description

A `data.frame` with a design and proteomic data.

Usage

```
data(MS)
```

Details

Data from biobank are analysed a study population of 101 patients, 37 were diagnosed with multiple sclerosis, and 64 without multiple sclerosis. Of the patients without multiple sclerosis, 50 were diagnosed with other neurological disorders and 14 were neurologically healthy patients who had undergone spinal anaesthesia for orthopaedic surgery on the knee or ankle, i.e. neurologically healthy controls. Unless otherwise stated, all the patients without multiple sclerosis were considered as controls for this study. All patients with multiple sclerosis had relapsing remitting multiple sclerosis. The proteome were obtained on cerebrospinal fluid samples from all patients prior medical treatment for multiple sclerosis. It was discovered the patients separated into two clusters, called group 1 and group 2. This is utilised in the data analysis by considering the data as 2-way factorial design with the two factors: MS and group both on two levels.

Author(s)

Ellen Færgestad Mosleth

References

* Opsahl, J.A. et al. Label-free analysis of human cerebrospinal fluid addressing various normalization strategies and revealing protein groups affected by multiple sclerosis. Proteomics 16, 1154-1165 (2016).

* Ellen Færgestad Mosleth, Christian Alexander Vedeler, Kristian Hovde Liland, Anette McLeod, Gerd Haga Bringland, Liesbeth Kroondijk, Frode Berven, Artem Lysenko, Christopher J. Rawlings, Karim El-Hajj Eid, Jill Anette Opsahl, Bjørn Tore Gjertsen, Kjell-Morten Myhr and Sonia Gavasso, Cerebrospinal fluid proteome shows disrupted neuronal development in multiple sclerosis. Scientific Reports – Nature 11(4087), (2021).

Examples

```
data(MS)
str(MS)
```

---

neuralnet *Neural Network by Multilayer Perceptron*

Description

Neural Network by Multilayer Perceptron

Usage

```
neuralnet(
  object,
  formula,
  factor = 1,
  hidden = c(2),
  linear.output = FALSE,
  ...
)
```

Arguments

`object`	Object of class `GEM`.
`formula`	A formula specifying the model to be fitted. If not provided, the response variable is taken from the `GEM` object.
`factor`	The factor to be used as response. If `formula` is provided, this is ignored.
`hidden`	Vector with numbers of neurons in the hidden layers.
`linear.output`	Logical. If `TRUE` the output layer is linear, otherwise it is logistic.
`...`	Additional arguments passed to `neuralnet`.

Value

A neuralnet object that can be inspected and plotted.

See Also

Analyses using GEM: elastic, pca, sca, neuralnet, pls. Confidence interval plots: confints. Convenience knock-in and knock-out of effects: knock.in.

Examples

```
data(candies, package = "HDANOVA")
gemC <- GEM(assessment ~ assessor*candy, data=candies)

# Neural network model
nn <- neuralnet(gemC, factor = "candy", hidden = c(2))
plot(nn, rep="best")

# Network weights (input and hidden layers)
nn$weights
```

---

pca *Principal Component Analysis*

Description

This function performs Principal Component Analysis (SCA) on a GEM/hdanova object.

Usage

```
pca(object)
```

Arguments

object	A GEM/hdanova object.

Value

An updated GEM/hdanova object with PCA results.

See Also

Analyses using GEM: elastic, pca, sca, neuralnet, pls. Confidence interval plots: confints. Convenience knock-in and knock-out of effects: knock.in.

Examples

```
# Load candies data
data(candies, package="HDANOVA")

# Basic HDANOVA model with two factors
mod <- GEM(assessment ~ candy + assessor, data=candies)
mod <- pca(mod)
scoreplot(mod)
```

`plot.GEMpls` *Plot function and extraction method for GEM-based PLS*

Description

Plot function and extraction method for GEM-based PLS

Usage

```
## S3 method for class 'GEMpls'
plot(x, y, ylab = "error", xlab = "nvar", main = "Shaving", ...)

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

## S3 method for class 'GEMpls'
summary(object, ...)

scores(object, ...)

scoreplot(object, ...)

loadings(object, ...)

loadingplot(object, ...)

corrplot(object, ...)

R2(object, ...)

mvrValstats(object, ...)

explvar(object, ...)
```

Arguments

`x, object`	`GEMpls` object

y	Not used
ylab	character label for Y axis.
xlab	character label for X axis.
main	character main header.
...	additional arguments for plot().

Value

A plot of the PLS model's cross-validated accuracy or the Shaving results if available.

See Also

pls has examples of how to use this function.

---

pls.GEM *Partial Least Squares modelling of GEM objects.*

Description

The output of GEM is used as input to a PLS classification with the selected effect as response. It is possible to compare two models using the gem2 argument. Variable selection is available through Jackknifing (from package pls) and Shaving (from package plsVarSel).

Usage

```
## S3 method for class 'GEM'
pls(
  object,
  effect,
  ncomp,
  newdata = NULL,
  gem2,
  validation,
  jackknife = NULL,
  shave = NULL,
  df.used = object$df.used,
  ...
)
```

Arguments

object	Object of class GEM.
effect	The effect to be used as response.
ncomp	Number of PLS components.
newdata	Optional new data matrix for prediction.

gem2	Second object of class GEM for comparison.
validation	Optional validation parameters for plsr.
jackknife	Optional argument specifying if jackknifing should be applied.
shave	Optional argument indicating if variable shaving should be used. shave should be a list with two elements: the PLS filter method and the proportion to remove. shave = TRUE uses defaults: list("sMC", 0.2).
df.used	Optional argument indicating how many degrees of freedom have been consumed during deflation. Default value from input object.
...	Additional arguments for plsr.

Details

If using the shave options, the segment type is given as type instead of segment.type (see examples).

Value

An object of class GEMpls, mvr, list containing the fitted PLS model, classifications/predictions, data and optionally Jackknife or Shaving results.

See Also

Analyses using GEM: elastic, pca, sca, neuralnet. Confidence interval plots: confints. Convenience knock-in and knock-out of effects: knock.in.

Examples

```
data(MS, package = "gemR")
# Subset to reduce runtime in example
MS$proteins <- MS$proteins[,20:70]

gem <- GEM(proteins ~ MS * group, data = MS[-1,])

# Simple PLS using interleaved cross-validation
plsMod <- pls(gem, 'MS', 6, validation = "CV",
              segment.type = "interleaved", length.seg = 25)
plot(plsMod)
scoreplot(plsMod, labels = "names")

# PLS with shaving of variables (mind different variable for cross-validation type)
plsModS <- pls(gem, 'MS', 6, validation = "CV",
              type = "interleaved", length.seg=25, shave = TRUE)
# Error as a function of remaining variables
plot(plsModS)
# Selected variables for minimum error
with(plsModS$shave, colnames(X)[variables[[min.red+1]]])

 # Time consuming due to leave-one-out cross-validation
  plsModJ <- pls(gem, 'MS', 5, validation = "LOO",
              jackknife = TRUE)
```

```
colSums(plsModJ$classes == as.numeric(MS$MS[-1]))
# Jackknifed coefficient P-values (sorted)
plot(sort(plsModJ$jack[,1,1]), pch = '.', ylab = 'P-value')
abline(h=c(0.01,0.05),col=2:3)

scoreplot(plsModJ)
scoreplot(plsModJ, comps=c(1,3))   # Selected components
# Use MS categories for colouring and clusters for plot characters.
scoreplot(plsModJ, col = gem$symbolicDesign[['MS']],
                pch = 20+as.numeric(gem$symbolicDesign[['group']]))
loadingplot(plsModJ, scatter=TRUE) # scatter=TRUE for scatter plot
```

---

sca *Simultaneous Component Analysis*

Description

This function performs Simultaneous Component Analysis (SCA) on a hdanova object.

Usage

```
sca(object)
```

Arguments

object	A hdanova object.

Value

An updated hdanova object with SCA results.

See Also

Analyses using GEM: elastic, pca, sca, neuralnet, pls. Confidence interval plots: confints. Convenience knock-in and knock-out of effects: knock.in.

Examples

```
# Load candies data
data(candies, package="HDANOVA")

# Basic HDANOVA model with two factors
mod <- GEM(assessment ~ candy + assessor, data=candies)
mod <- sca(mod)
scoreplot(mod)
```

Index

coef.GEMglmnet *(*elastic*)*, 5
confints, 2, *6*, *8*, *13*, *16, 17*
corrplot *(*plot.GEMpls*)*, 14

Diabetes, 4

elastic, *3*, 5, *8*, *13*, *16, 17*
explvar *(*plot.GEMpls*)*, 14

GEM, 6
gemR *(*GEM*)*, 6
glmnet, *5*

knock.in, *6*, *8*, 9, *13*, *16, 17*
knock.out *(*knock.in*)*, 10

Lactobacillus, 10
loadingplot *(*plot.GEMpls*)*, 14
loadings *(*plot.GEMpls*)*, 14

MS, 11
mvrValstats *(*plot.GEMpls*)*, 14

neuralnet, *3*, *6*, *8*, 12, *13*, *16, 17*

pca, *3*, *6*, *8*, 13, *13*, *16, 17*
plot.confints *(*confints*)*, 2
plot.GEM *(*GEM*)*, 6
plot.GEM.pls *(*plot.GEMpls*)*, 14
plot.GEMpls, 14
pls, *3*, *6*, *8*, *13*, *15*, *17*
pls.GEM, 15
print.GEM *(*GEM*)*, 6
print.GEMpls *(*plot.GEMpls*)*, 14
print.summary.GEM *(*GEM*)*, 6

R2 *(*plot.GEMpls*)*, 14

sca, *3*, *6*, *8*, *13*, *16*, 17, *17*
scoreplot *(*plot.GEMpls*)*, 14
scores *(*plot.GEMpls*)*, 14
summary.GEM *(*GEM*)*, 6
summary.GEMpls *(*plot.GEMpls*)*, 14

tableGEM *(*GEM*)*, 6