# Package ‘GPCMlasso’

May 7, 2026

**Type** Package

**Title** Differential Item Functioning in Generalized Partial Credit Models

**Version** 0.1-8

**Date** 2025-07-23

**Maintainer** Gunther Schauberger <gunther.schauberger@tum.de>

**Description** Provides a framework to detect Differential Item Functioning (DIF) in Generalized Partial Credit Models (GPCM) and special cases of the GPCM as proposed by Schauberger and Mair (2019) <doi:10.3758/s13428-019-01224-2>. A joint model is set up where DIF is explicitly parametrized and penalized likelihood estimation is used for parameter selection. The big advantage of the method called GPCMlasso is that several variables can be treated simultaneously and that both continuous and categorical variables can be used to detect DIF.

**License** GPL (>= 2)

**Imports** Rcpp (>= 0.12.4), TeachingDemos, cubature, caret, statmod, mvtnorm, mirt, methods

**Depends** ltm

**LinkingTo** Rcpp, RcppArmadillo

**RoxygenNote** 7.3.2

**Encoding** UTF-8

**NeedsCompilation** yes

**Author** Gunther Schauberger [aut, cre]

**Repository** CRAN

**Date/Publication** 2025-07-23 10:50:07 UTC

## Contents

---

`GPCMlasso-package` *Find DIF in Generalized Partial Credit Models*

---

### Description

Performs GPCMlasso, a method to identify DIF in Generalized Partial Credit Models. A joint parametric model is set up based on an IRT model chosen by the user. Several variables can be considered simultaneously. For each pair between variable and item, a parametric DIF effect is introduced which indicates DIF if the respective parameter is selected (estimated to be unequal zero). Parameter selection is done using a lasso-type penalization term.

### Author(s)

**Maintainer**: Gunther Schauberger `<gunther.schauberger@tum.de>`

Gunther Schauberger
`<gunther.schauberger@tum.de>`

### References

Schauberger, Gunther and Mair, Patrick (2019): A Regularization Approach for the Detection of Differential Item Functioning in Generalized Partial Credit Models, *Behavior Research Methods*, https://link.springer.com/article/10.3758/s13428-019-01224-2

### See Also

`GPCMlasso`

### Examples

```
data(tenseness_small)

## formula for simple model without covariates
form.0 <- as.formula(paste("cbind(",paste(colnames(tenseness_small)[1:5],collapse=","),")~0"))

######
## fit simple RSM where loglikelihood and score function are evaluated parallel on 2 cores
rsm.0 <- GPCMlasso(form.0, tenseness_small, model = "RSM",
control= ctrl_GPCMlasso(cores=2))
rsm.0

## Not run:
```

```
## formula for model with covariates (and DIF detection)
form <- as.formula(paste("cbind(",paste(colnames(tenseness_small)[1:5],collapse=","),")~."))

######
## fit GPCM model with 10 different tuning parameters
gpcm <- GPCMlasso(form, tenseness_small, model = "GPCM",
                  control = ctrl_GPCMlasso(l.lambda = 10))
gpcm
plot(gpcm)
pred.gpcm <- predict(gpcm)
trait.gpcm <- trait.posterior(gpcm)

######
## fit RSM, detect differential step functioning (DSF)
rsm.DSF <- GPCMlasso(form, tenseness_small, model = "RSM", DSF = TRUE,
                     control = ctrl_GPCMlasso(l.lambda = 10))
rsm.DSF
plot(rsm.DSF)

## create binary data set
tenseness_small_binary <- tenseness_small
tenseness_small_binary[,1:5][tenseness_small[,1:5]>1] <- 2

######
## fit and cross-validate Rasch model
set.seed(1860)
rm.cv <- GPCMlasso(form, tenseness_small_binary, model = "RM", cv = TRUE,
                   control = ctrl_GPCMlasso(l.lambda = 10))
rm.cv
plot(rm.cv)

## End(Not run)
```

---

`ctrl_GPCMlasso` *Control function for GPCMlasso*

---

## Description

Control parameters for penalty terms and for tuning the fitting algorithm.

## Usage

```
ctrl_GPCMlasso(
  log.lambda = TRUE,
  lambda = NULL,
  l.lambda = 50,
  lambda.min = 0.1,
  penalize.main.effects = FALSE,
  fuse.per.variable = FALSE,
  adaptive = TRUE,
```

```
  weight.penalties = TRUE,
  ada.lambda = 1e-04,
  ada.power = 1,
  Q = 15,
  lambda2 = 1e-04,
  cvalue = 1e-05,
  trace = TRUE,
  folds = 10,
  cores = 25,
  null_thresh = 0.01,
  gradtol = 1e-06,
  steptol = 1e-06,
  iterlim = 500,
  precision = 3,
  all.dummies = FALSE,
  ctrl.gpcm = list()
)
```

## Arguments

`log.lambda` Should the grid of tuning parameters be created on a log scale?

`lambda` Optional argument to specify a vector of tuning parameters. If `lambda = NULL`, a vector of length `l.lambda` is created automatically.

`l.lambda` Specifies the length of the grid of tuning parameters.

`lambda.min` Minimal value used if the grid of tuning parameters is created automatically.

`penalize.main.effects` Should also main covariate effects be penalized? Default is `FALSE`.

`fuse.per.variable` Should fusion be applied per variable? This option creates clusters of items with equal effects for a variable.

`adaptive` Should adaptive lasso be used? Default is `TRUE`.

`weight.penalties` Should penalties be weighted according to the number of penalty term and the number of parameters corresponding to one pair between item and covariate. Only relevant if both `DSF = TRUE` and the number of response categories differs across items (because only then these values can differ).

`ada.lambda` Size of tuning parameter for Ridge-regularized estimation of parameters used for adaptive weights.

`ada.power` By default, 1st power of absolute values of Ridge-regularized estimates are used. Could be changed to squared values by `ada-power = 2`.

`Q` Number of nodes to be used in Gauss-Hermite quadrature.

`lambda2` Tuning parameter for ridge penalty on all coefficients except sigma/slope parameters. Should be small, only used to stabilize results.

`cvalue` Internal parameter for the quadratic approximation of the L1 penalty. Should be sufficiently small.

| | |
|---|---|
| trace | Should the trace of the progress (current tuning parameter) be printed? |
| folds | Number of folds for cross-validation. Only relevant if cv = TRUE in GPCMlasso. |
| cores | Number of cores to be used parallel when fitting the model. |
| null_thresh | Threshold which is used to distinguih between values equal and unequal to zero. |
| gradtol | Parameter to tune optimization accuracy, for details see nlm. |
| steptol | Parameter to tune optimization accuracy, for details see nlm. |
| iterlim | Parameter to tune optimization accuracy, for details see nlm. |
| precision | Number of decimal places used to round coefficient estimates. |
| all.dummies | Should (in case of factors with more than 2 categories) the dummy variables for all categories be included in the design matrix? If all.dummies = TRUE, the dependence on the reference category is eliminated for multi-categorical covariates. |
| ctrl.gpcm | List of control arguments for initial gpcm fit, which is needed to get good starting values. Does not apply to RSM or GRSM. |

## Author(s)

Gunther Schauberger
<gunther.schauberger@tum.de>

## References

Schauberger, Gunther and Mair, Patrick (2019): A Regularization Approach for the Detection of Differential Item Functioning in Generalized Partial Credit Models, *Behavior Research Methods*, https://link.springer.com/article/10.3758/s13428-019-01224-2

## Examples

```
data(tenseness_small)

## formula for simple model without covariates
form.0 <- as.formula(paste("cbind(",paste(colnames(tenseness_small)[1:5],collapse=","),")~0"))

######
## fit simple RSM where loglikelihood and score function are evaluated parallel on 2 cores
rsm.0 <- GPCMlasso(form.0, tenseness_small, model = "RSM",
control= ctrl_GPCMlasso(cores=2))
rsm.0

## Not run:
## formula for model with covariates (and DIF detection)
form <- as.formula(paste("cbind(",paste(colnames(tenseness_small)[1:5],collapse=","),")~."))

######
## fit GPCM model with 10 different tuning parameters
gpcm <- GPCMlasso(form, tenseness_small, model = "GPCM",
                  control = ctrl_GPCMlasso(l.lambda = 10))
gpcm
```

```
plot(gpcm)
pred.gpcm <- predict(gpcm)
trait.gpcm <- trait.posterior(gpcm)

######
## fit RSM, detect differential step functioning (DSF)
rsm.DSF <- GPCMlasso(form, tenseness_small, model = "RSM", DSF = TRUE,
                     control = ctrl_GPCMlasso(l.lambda = 10))
rsm.DSF
plot(rsm.DSF)

## create binary data set
tenseness_small_binary <- tenseness_small
tenseness_small_binary[,1:5][tenseness_small[,1:5]>1] <- 2

######
## fit and cross-validate Rasch model
set.seed(1860)
rm.cv <- GPCMlasso(form, tenseness_small_binary, model = "RM", cv = TRUE,
                   control = ctrl_GPCMlasso(l.lambda = 10))
rm.cv
plot(rm.cv)

## End(Not run)
```

---

GPCMlasso *GPCMlasso*

---

## Description

Performs GPCMlasso, a method to identify differential item functioning (DIF) in Generalized Partial Credit Models. A joint parametric model is set up based on an IRT model chosen by the user. Several variables can be considered simultaneously. For each pair between variable and item, a parametric DIF effect is introduced which indicates DIF if the respective parameter is selected (estimated to be unequal zero). Parameter selection is done using a lasso-type penalization term.

## Usage

```
GPCMlasso(
  formula,
  data,
  DSF = FALSE,
  model = c("PCM", "RSM", "GPCM", "GRSM", "RM", "2PL"),
  control = ctrl_GPCMlasso(),
  cv = FALSE,
  main.effects = TRUE
)
```

## Arguments

| | |
|---|---|
| formula | Formula to indicate which items are considered and which covariates should be used to find DIF. Items are considered to be the response and are concatenated by cbind(). If the RHS of the formula is ~0, simply the model specified in model is calulated. |
| data | Data frame containing the ordinal item response data (as ordered factors) and all covariates. |
| DSF | Should Differential Step Functioning (DSF) be considered? If DSF = TRUE, one parameter per step between two response categories is introduced. For binary items, DSF and DIF conincide. |
| model | Specify the underlying basic model. Currently, you can choose between the partial credit model and the rating scale model and the respective generalized versions of both models called 'PCM', 'RSM', 'GPCM' and 'GRSM'. Generalized models allow for different discrimination parameters between items. |
| control | Control argument to specify further arguments for the algorithm and numerical optimization, specified by ctrl_GPCMlasso. |
| cv | Should cross-validation be performed? Cross-validation can be used as an alternative to BIC to select the optimal tuning parameter. |
| main.effects | Should also main effects of the variables be included in the model? Default is TRUE. Here, positive parameter estimates correspond to an increase of the respective trait if the variable increases. |

## Value

| | |
|---|---|
| coefficients | Matrix containing all parameters for the GPCMlasso model, one row per tuning parameter lambda. Due to the penalty the parameters are scaled and, therefore, are comparable with respect to their size. |
| logLik | Vector of log-likelihoods, one value per tuning parameter lambda. |
| call | The function call of GPCMlasso |
| cv_error | Vector of cv_errors, one per tuning parameter. Only relevant if cv = TRUE. |
| model | Basic IRT model chosen by user. |
| data | Data from call. |
| control | Control list. |
| DSF | DSF from call. |
| formula | Formula from call. |
| item.names | Item names. |
| Y | Matrix containing item responses. |
| design_list | List containing several helpful objects for internal use. |
| AIC | Vector of AIC values, one per tuning parameter. |
| BIC | Vector of BIC values, one per tuning parameter. |
| cAIC | Vector of corrected AIC values, one per tuning parameter. |
| df | Vector of degrees of freedom, one per tuning parameter. |

coef.rescal — Matrix containing all rescaled parameters for the GPCMlasso model, one row per tuning parameter lambda. In contrast to `coefficients`, all parameters are rescaled back to their original scales.

## Author(s)

Gunther Schauberger
<gunther.schauberger@tum.de>

## References

Schauberger, Gunther and Mair, Patrick (2019): A Regularization Approach for the Detection of Differential Item Functioning in Generalized Partial Credit Models, *Behavior Research Methods*, https://link.springer.com/article/10.3758/s13428-019-01224-2

## See Also

GPCMlasso-package, ctrl_GPCMlasso, print.GPCMlasso, plot.GPCMlasso, predict.GPCMlasso, trait.posterior

## Examples

```
data(tenseness_small)

## formula for simple model without covariates
form.0 <- as.formula(paste("cbind(",paste(colnames(tenseness_small)[1:5],collapse=","),")~0"))

######
## fit simple RSM where loglikelihood and score function are evaluated parallel on 2 cores
rsm.0 <- GPCMlasso(form.0, tenseness_small, model = "RSM",
control= ctrl_GPCMlasso(cores=2))
rsm.0

## Not run:
## formula for model with covariates (and DIF detection)
form <- as.formula(paste("cbind(",paste(colnames(tenseness_small)[1:5],collapse=","),")~."))

######
## fit GPCM model with 10 different tuning parameters
gpcm <- GPCMlasso(form, tenseness_small, model = "GPCM",
                  control = ctrl_GPCMlasso(l.lambda = 10))
gpcm
plot(gpcm)
pred.gpcm <- predict(gpcm)
trait.gpcm <- trait.posterior(gpcm)

######
## fit RSM, detect differential step functioning (DSF)
rsm.DSF <- GPCMlasso(form, tenseness_small, model = "RSM", DSF = TRUE,
                     control = ctrl_GPCMlasso(l.lambda = 10))
rsm.DSF
plot(rsm.DSF)
```

```
## create binary data set
tenseness_small_binary <- tenseness_small
tenseness_small_binary[,1:5][tenseness_small[,1:5]>1] <- 2

######
## fit and cross-validate Rasch model
set.seed(1860)
rm.cv <- GPCMlasso(form, tenseness_small_binary, model = "RM", cv = TRUE,
                  control = ctrl_GPCMlasso(l.lambda = 10))
rm.cv
plot(rm.cv)

## End(Not run)
```

---

`plot.GPCMlasso` *Plot function for GPCMlasso*

---

## Description

Plot function for a `GPCMlasso` object. Plots show coefficient paths of DIF (or DSF) parameters along (a transformation of) the tuning parameter lambda. One plot per item is created, every single parameter corresponding to this item is depicted by a single path. The optimal model is highlighted with a red dashed line.

## Usage

```
## S3 method for class 'GPCMlasso'
plot(x, select = c("BIC", "AIC", "cAIC", "cv"),
type = c("DIF", "Variable"),
log.lambda = TRUE, items_per_page = 1, items = "all",
columns = NULL, ask_new = TRUE, lambda.lines = TRUE,
equal_range = TRUE, add.to.lambda = 0, ...)
```

## Arguments

| | |
|---|---|
| `x` | `GPCMlasso` object |
| `select` | Specifies which criterion to use for the optimal model, we recommend the default value "BIC". If cross-validation was performed, automatically the optimal model according to cross-validation is used. The chosen optimal model is highlighted with a red dashed line. |
| `type` | Which plot type should be used. `DIF` means that one plot per item is shown, indicating whether DIF was found for this item. `Variable` means that one plot per variable is shown, which is useful for the option `fuse.per.variable`. |
| `log.lambda` | A logical value indicating whether lambda or a log-transformation of lambda should be used as x-axis in the plots. |

| | |
|---|---|
| items_per_page | By default, each plot/item is put on a separate page. For example, items_per_page=4 would put four plots/items on one page. |
| items | By default, all items are plotted. If items=c(1,3), only the first and the third item are plotted. |
| columns | Specifies the number of columns to use when several plots are on one page. Only relevant if items_per_page>1. |
| ask_new | If TRUE, the user is asked to confirm before the next item is plotted. |
| lambda.lines | A logical value indicating whether a thin gray line plotted for each value from the vector of tuning parameters from object |
| equal_range | A logical value indicating whether for each plot equal limits on the y-axis shall be used. |
| add.to.lambda | Constant c to be added to lambda as in log(lambda + c) for plotting coefficient paths. |
| ... | Further plot arguments. |

## Author(s)

Gunther Schauberger
<gunther.schauberger@tum.de>

## References

Schauberger, Gunther and Mair, Patrick (2019): A Regularization Approach for the Detection of Differential Item Functioning in Generalized Partial Credit Models, *Behavior Research Methods*, https://link.springer.com/article/10.3758/s13428-019-01224-2

## See Also

GPCMlasso

## Examples

```
data(tenseness_small)

## formula for simple model without covariates
form.0 <- as.formula(paste("cbind(",paste(colnames(tenseness_small)[1:5],collapse=","),")~0"))

######
## fit simple RSM where loglikelihood and score function are evaluated parallel on 2 cores
rsm.0 <- GPCMlasso(form.0, tenseness_small, model = "RSM",
control= ctrl_GPCMlasso(cores=2))
rsm.0

## Not run:
## formula for model with covariates (and DIF detection)
form <- as.formula(paste("cbind(",paste(colnames(tenseness_small)[1:5],collapse=","),")~."))

######
## fit GPCM model with 10 different tuning parameters
```

```
gpcm <- GPCMlasso(form, tenseness_small, model = "GPCM",
                  control = ctrl_GPCMlasso(l.lambda = 10))
gpcm
plot(gpcm)
pred.gpcm <- predict(gpcm)
trait.gpcm <- trait.posterior(gpcm)

######
## fit RSM, detect differential step functioning (DSF)
rsm.DSF <- GPCMlasso(form, tenseness_small, model = "RSM", DSF = TRUE,
                     control = ctrl_GPCMlasso(l.lambda = 10))
rsm.DSF
plot(rsm.DSF)

## create binary data set
tenseness_small_binary <- tenseness_small
tenseness_small_binary[,1:5][tenseness_small[,1:5]>1] <- 2

######
## fit and cross-validate Rasch model
set.seed(1860)
rm.cv <- GPCMlasso(form, tenseness_small_binary, model = "RM", cv = TRUE,
                   control = ctrl_GPCMlasso(l.lambda = 10))
rm.cv
plot(rm.cv)

## End(Not run)
```

---

`predict.GPCMlasso` *Predict function for GPCMlasso*

---

## Description

Predict function for a `GPCMlasso` object. Predictions can be linear predictors or probabilities separately for each person and each item.

## Usage

```
## S3 method for class 'GPCMlasso'
predict(
  object,
  coefs = NULL,
  newdata = NULL,
  type = c("link", "response"),
  ...
)
```

### Arguments

| | |
|---|---|
| `object` | `GPCMlasso` object |
| `coefs` | Optional vector of coefficients, can be filled with a specific row from `object$coefficients`. If not specified, `coefs` are specifed to be the BIC-optimal coefficients or, if cross-validation was performed, the optimal coefficients according to cross-validation. |
| `newdata` | List possibly containing slots Y, X, Z1 and Z2 to use new data for prediction. |
| `type` | Type "link" gives vectors of linear predictors for separate categories (of length $k_i-1$) and type "response" gives the respective probabilities (of length $k_i$). |
| `...` | Further predict arguments. |

### Details

Results are lists of vectors with length equal to the number of response categories $k_i$ in case of probabilities (`type="response"`) or $k_i-1$ in case of linear predictors (`type="link"`).

### Author(s)

Gunther Schauberger
<gunther@stat.uni-muenchen.de>

### See Also

`GPCMlasso`

### Examples

```
data(tenseness_small)

## formula for simple model without covariates
form.0 <- as.formula(paste("cbind(",paste(colnames(tenseness_small)[1:5],collapse=","),")~0"))

######
## fit simple RSM where loglikelihood and score function are evaluated parallel on 2 cores
rsm.0 <- GPCMlasso(form.0, tenseness_small, model = "RSM",
control= ctrl_GPCMlasso(cores=2))
rsm.0

## Not run:
## formula for model with covariates (and DIF detection)
form <- as.formula(paste("cbind(",paste(colnames(tenseness_small)[1:5],collapse=","),")~."))

######
## fit GPCM model with 10 different tuning parameters
gpcm <- GPCMlasso(form, tenseness_small, model = "GPCM",
                  control = ctrl_GPCMlasso(l.lambda = 10))
gpcm
plot(gpcm)
pred.gpcm <- predict(gpcm)
```

```
trait.gpcm <- trait.posterior(gpcm)

######
## fit RSM, detect differential step functioning (DSF)
rsm.DSF <- GPCMlasso(form, tenseness_small, model = "RSM", DSF = TRUE,
                     control = ctrl_GPCMlasso(l.lambda = 10))
rsm.DSF
plot(rsm.DSF)

## create binary data set
tenseness_small_binary <- tenseness_small
tenseness_small_binary[,1:5][tenseness_small[,1:5]>1] <- 2

######
## fit and cross-validate Rasch model
set.seed(1860)
rm.cv <- GPCMlasso(form, tenseness_small_binary, model = "RM", cv = TRUE,
                   control = ctrl_GPCMlasso(l.lambda = 10))
rm.cv
plot(rm.cv)

## End(Not run)
```

---

`print.GPCMlasso` *Print function for GPCMlasso*

---

## Description

Print function for a `GPCMlasso` object. Prints parameters estimates for all model components for the optimal model chosen by a specific criterion (by default BIC).

## Usage

```
## S3 method for class 'GPCMlasso'
print(x, select = c("BIC", "AIC", "cAIC", "cv"), ...)
```

## Arguments

| | |
|---|---|
| `x` | `GPCMlasso` object |
| `select` | Specifies which criterion to use for the optimal model, we recommend the default value "BIC". If cross-validation was performed, automatically the optimal model according to cross-validation is used. Only the parameter estimates from the chosen optimal model are printed. |
| `...` | Further print arguments. |

## Author(s)

Gunther Schauberger
<gunther.schauberger@tum.de>

## References

Schauberger, Gunther and Mair, Patrick (2019): A Regularization Approach for the Detection of Differential Item Functioning in Generalized Partial Credit Models, *Behavior Research Methods*, https://link.springer.com/article/10.3758/s13428-019-01224-2

## See Also

GPCMlasso

## Examples

```
data(tenseness_small)

## formula for simple model without covariates
form.0 <- as.formula(paste("cbind(",paste(colnames(tenseness_small)[1:5],collapse=","),")~0"))

######
## fit simple RSM where loglikelihood and score function are evaluated parallel on 2 cores
rsm.0 <- GPCMlasso(form.0, tenseness_small, model = "RSM",
control= ctrl_GPCMlasso(cores=2))
rsm.0

## Not run:
## formula for model with covariates (and DIF detection)
form <- as.formula(paste("cbind(",paste(colnames(tenseness_small)[1:5],collapse=","),")~."))

######
## fit GPCM model with 10 different tuning parameters
gpcm <- GPCMlasso(form, tenseness_small, model = "GPCM",
                  control = ctrl_GPCMlasso(l.lambda = 10))
gpcm
plot(gpcm)
pred.gpcm <- predict(gpcm)
trait.gpcm <- trait.posterior(gpcm)

######
## fit RSM, detect differential step functioning (DSF)
rsm.DSF <- GPCMlasso(form, tenseness_small, model = "RSM", DSF = TRUE,
                     control = ctrl_GPCMlasso(l.lambda = 10))
rsm.DSF
plot(rsm.DSF)

## create binary data set
tenseness_small_binary <- tenseness_small
tenseness_small_binary[,1:5][tenseness_small[,1:5]>1] <- 2

######
## fit and cross-validate Rasch model
set.seed(1860)
rm.cv <- GPCMlasso(form, tenseness_small_binary, model = "RM", cv = TRUE,
                   control = ctrl_GPCMlasso(l.lambda = 10))
rm.cv
```

```
plot(rm.cv)

## End(Not run)
```

---

`tenseness` *Tenseness data from the Freiburg Complaint Checklist*

---

## Description

Data from the Freiburg Complaint Checklist. The data contain all 8 items corresponding to the scale *Tenseness* for 2042 participants of the standardization sample of the Freiburg Complaint Checklist.

## Format

A data frame containing data from the Freiburg Complaint Checklist with 1847 observations. All items refer to the scale *Tenseness* and are measured on a 5-point Likert scale where low numbers correspond to low frequencies or low intensitites of the respective complaint and vice versa.

**Clammy_hands** Do you have clammy hands?

**Sweat_attacks** Do you have sudden attacks of sweating?

**Clumsiness** Do you notice that you behave clumsy?

**Wavering_hands** Are your hands wavering frequently, e.g. when lightning a cigarette or when holding a cup?

**Restless_hands** Do you notice that your hands are restless?

**Restless_feet** Do you notice that your feet are restless?

**Twitching_eyes** Do you notice unvoluntary twitching of your eyes?

**Twitching_mouth** Do you notice unvoluntary twitching of your mouth?

**Gender** Gender of the person

**Household** Does the person live alone in a household or together with somebody?

**Income** Income, categorized to levels from 1 (low income) to 11(high income). For simplicity, due to the high number of categories income can be treated as a metric variable.

**WestEast** Is the person from East Germany (former GDR)?

**Abitur** Does the person have Abitur (A-levels)?

**Age** Age of the person

## Source

ZPID (2013). PsychData of the Leibniz Institute for Psychology Information ZPID. Trier: Center for Research Data in Psychology.

Fahrenberg, J. (2010). Freiburg Complaint Checklist [Freiburger Beschwerdenliste (FBL)]. Goettingen, Hogrefe.

## Examples

```
data(tenseness)
```

tenseness_small *Subset of tenseness data from the Freiburg Complaint Checklist*

## Description

Data from the Freiburg Complaint Checklist. The data contain 5 items (out of 8) corresponding to the scale *Tenseness* for a subset of 200 participants of the standardization sample of the Freiburg Complaint Checklist.

## Format

A data frame containing data from the Freiburg Complaint Checklist a subset of 200 observations. The complete data set with 1847 observations can be found in tenseness. All items refer to the scale *Tenseness* and are measured on a 5-point Likert scale where low numbers correspond to low frequencies or low intensitites of the respective complaint and vice versa.

**Clammy_hands** Do you have clammy hands?

**Sweat_attacks** Do you have sudden attacks of sweating?

**Clumsiness** Do you notice that you behave clumsy?

**Wavering_hands** Are your hands wavering frequently, e.g. when lightning a cigarette or when holding a cup?

**Restless_hands** Do you notice that your hands are restless?

**Gender** Gender of the person

**Age** Age of the person

## Source

ZPID (2013). PsychData of the Leibniz Institute for Psychology Information ZPID. Trier: Center for Research Data in Psychology.

Fahrenberg, J. (2010). Freiburg Complaint Checklist [Freiburger Beschwerdenliste (FBL)]. Goettingen, Hogrefe.

## See Also

GPCMlasso, ctrl_GPCMlasso, trait.posterior

## Examples

```
data(tenseness_small)

## formula for simple model without covariates
form.0 <- as.formula(paste("cbind(",paste(colnames(tenseness_small)[1:5],collapse=","),")~0"))

######
## fit simple RSM where loglikelihood and score function are evaluated parallel on 2 cores
rsm.0 <- GPCMlasso(form.0, tenseness_small, model = "RSM",
```

```
control= ctrl_GPCMlasso(cores=2))
rsm.0

## Not run:
## formula for model with covariates (and DIF detection)
form <- as.formula(paste("cbind(",paste(colnames(tenseness_small)[1:5],collapse=","),")~."))

######
## fit GPCM model with 10 different tuning parameters
gpcm <- GPCMlasso(form, tenseness_small, model = "GPCM",
                  control = ctrl_GPCMlasso(l.lambda = 10))
gpcm
plot(gpcm)
pred.gpcm <- predict(gpcm)
trait.gpcm <- trait.posterior(gpcm)

######
## fit RSM, detect differential step functioning (DSF)
rsm.DSF <- GPCMlasso(form, tenseness_small, model = "RSM", DSF = TRUE,
                     control = ctrl_GPCMlasso(l.lambda = 10))
rsm.DSF
plot(rsm.DSF)

## create binary data set
tenseness_small_binary <- tenseness_small
tenseness_small_binary[,1:5][tenseness_small[,1:5]>1] <- 2

######
## fit and cross-validate Rasch model
set.seed(1860)
rm.cv <- GPCMlasso(form, tenseness_small_binary, model = "RM", cv = TRUE,
                   control = ctrl_GPCMlasso(l.lambda = 10))
rm.cv
plot(rm.cv)

## End(Not run)
```

---

`trait.posterior` *Calculate Posterior Estimates for Trait Parameters*

---

## Description

Calculates posterior estimates for trait/person parameters using the assumption of Gaussian distributed parameters.

## Usage

```
trait.posterior(model, coefs = NULL, cores = 25, tol = 1e-04)
```

### Arguments

| | |
|---|---|
| model | Object of class GPCMlasso. |
| coefs | Vector of coefficients to be used for prediction. If coefs = NULL, the parameters from the BIC-optimal model will be used. If cross-validation was performed, automatically the parameters from the optimal model according to cross-validation are used. |
| cores | Number of cores to be used in parallelized computation. |
| tol | The maximum tolerance for numerical integration, for more details see pcubature. |

### Value

Vector containing all estimates of trait/person parameters.

### Author(s)

Gunther Schauberger
<gunther.schauberger@tum.de>

### References

Schauberger, Gunther and Mair, Patrick (2019): A Regularization Approach for the Detection of Differential Item Functioning in Generalized Partial Credit Models, *Behavior Research Methods*, https://link.springer.com/article/10.3758/s13428-019-01224-2

### See Also

GPCMlasso GPCMlasso-package

### Examples

```
data(tenseness_small)

## formula for simple model without covariates
form.0 <- as.formula(paste("cbind(",paste(colnames(tenseness_small)[1:5],collapse=","),")~0"))

######
## fit simple RSM where loglikelihood and score function are evaluated parallel on 2 cores
rsm.0 <- GPCMlasso(form.0, tenseness_small, model = "RSM",
control= ctrl_GPCMlasso(cores=2))
rsm.0

## Not run:
## formula for model with covariates (and DIF detection)
form <- as.formula(paste("cbind(",paste(colnames(tenseness_small)[1:5],collapse=","),")~."))

######
## fit GPCM model with 10 different tuning parameters
gpcm <- GPCMlasso(form, tenseness_small, model = "GPCM",
                  control = ctrl_GPCMlasso(l.lambda = 10))
gpcm
```

```
plot(gpcm)
pred.gpcm <- predict(gpcm)
trait.gpcm <- trait.posterior(gpcm)

######
## fit RSM, detect differential step functioning (DSF)
rsm.DSF <- GPCMlasso(form, tenseness_small, model = "RSM", DSF = TRUE,
                     control = ctrl_GPCMlasso(l.lambda = 10))
rsm.DSF
plot(rsm.DSF)

## create binary data set
tenseness_small_binary <- tenseness_small
tenseness_small_binary[,1:5][tenseness_small[,1:5]>1] <- 2

######
## fit and cross-validate Rasch model
set.seed(1860)
rm.cv <- GPCMlasso(form, tenseness_small_binary, model = "RM", cv = TRUE,
                   control = ctrl_GPCMlasso(l.lambda = 10))
rm.cv
plot(rm.cv)

## End(Not run)
```

# Index