Package ‘ordPens’

August 17, 2021

Type Package

Title Selection, Fusion, Smoothing and Principal Components Analysis
for Ordinal Variables

Version 1.0.0

Description Selection, fusion, and/or smoothing of ordinally scaled independent variables using a group lasso, fused lasso or generalized ridge penalty, as well as non-linear principal components analysis for ordinal variables using a second-order difference/smoothing penalty.

Depends grplasso, mgcv, RLRsim, quadprog, glmpath

Suggests psy, knitr, rmarkdown

VignetteBuilder knitr

License GPL-2

LazyLoad yes

NeedsCompilation no

Repository CRAN

Author Jan Gertheiss [aut],
Aisouda Hoshiyar [aut, cre],
Fabian Scheipl [ctb]

Maintainer Aisouda Hoshiyar <aisouda.hoshiyar@hsu-hh.de>

Date/Publication 2021-08-17 16:10:08 UTC

R topics documented:

ordPens-package *Selection and/or Smoothing and Principal Components Analysis for Ordinal Variables*

Description

Selection, and/or smoothing/fusing of ordinally scaled independent variables using a group lasso or generalized ridge penalty. Nonlinear principal components analysis for ordinal variables using a second-order difference penalty.

Details

Package:	ordPens
Type:	Package
Version:	1.0.0
Date:	2021-08-17
Depends:	grplasso, mgcv, RLRsim, quadprog, glmpath
Suggests:	psy
License:	GPL-2
LazyLoad:	yes

Smoothing and selection of ordinal predictors is done by the function `ordSelect`; smoothing only, by `ordSmooth`; fusion and selection of ordinal predictors by `ordFusion`. For ANOVA with ordinal factors, use `ordAOV`. Nonlinear PCA, performance evaluation and selection of an optimal penalty parameter can be done using `ordPCA`.

Author(s)

Authors: Jan Gertheiss <`jan.gertheiss@hsu-hh.de`>, Aisouda Hoshiyar <`aisouda.hoshiyar@hsu-hh.de`>.

Contributors: Fabian Scheipl

Maintainer: Aisouda Hoshiyar <`aisouda.hoshiyar@hsu-hh.de`>

References

Gertheiss, J. (2014). *ANOVA for factors with ordered levels*, Journal of Agricultural, Biological and Environmental Statistics, 19, 258-277.

Gertheiss, J., S. Hogger, C. Oberhauser and G. Tutz (2011). *Selection of ordinally scaled independent variables with applications to international classification of functioning core sets.* Journal of the Royal Statistical Society C (Applied Statistics), 60, 377-395.

Gertheiss, J. and F. Oehrlein (2011). *Testing relevance and linearity of ordinal predictors*, Electronic Journal of Statistics, 5, 1935-1959.

Gertheiss, J., F. Scheipl, T. Lauer, and H. Ehrhardt (2021). *Statistical inference for ordinal predictors in generalized linear and additive models with application to bronchopulmonary dysplasia*. Preprint, available from https://arxiv.org/abs/2102.01946.

Gertheiss, J. and G. Tutz (2009). *Penalized regression with ordinal predictors*. International Statistical Review, 77, 345-365.

Gertheiss, J. and G. Tutz (2010). *Sparse modeling of categorial explanatory variables*. The Annals of Applied Statistics, 4, 2150-2180.

Hoshiyar, A. (2020). *Analyzing Likert-type data using penalized non-linear principal components analysis*, in: Proceedings of the 35th International Workshop on Statistical Modelling, Vol. I, 337-340.

Hoshiyar, A., H.A.L. Kiers, and J. Gertheiss (2021). *Penalized non-linear principal components analysis for ordinal variables with an application to international classification of functioning core sets*, Preprint.

Tutz, G. and J. Gertheiss (2014). *Rating scales as predictors – the old question of scale level and some answers*. Psychometrica, 79, 357-376.

Tutz, G. and J. Gertheiss (2016). *Regularized regression for categorical data*. Statistical Modelling, 16, 161-200.

See Also

ordSelect, ordSmooth, ordFusion, ordAOV, ordPCA

Examples

```
## Not run:
### smooth modeling of a simulated dataset
set.seed(123)

# generate (ordinal) predictors
x1 <- sample(1:8,100,replace=TRUE)
x2 <- sample(1:6,100,replace=TRUE)
x3 <- sample(1:7,100,replace=TRUE)

# the response
y <- -1 + log(x1) + sin(3*(x2-1)/pi) + rnorm(100)

# x matrix
x <- cbind(x1,x2,x3)

# lambda values
lambda <- c(1000,500,200,100,50,30,20,10,1)

# smooth modeling
o1 <- ordSmooth(x = x, y = y, lambda = lambda)

# results
round(o1$coef,digits=3)
plot(o1)
```

```
# If for a certain plot the x-axis should be annotated in a different way,
# this can (for example) be done as follows:
plot(o1, whx = 1, xlim = c(0,9), xaxt = "n")
axis(side = 1, at = c(1,8), labels = c("no agreement","total agreement"))


### nonlinear PCA on chronic widespread pain data
# load example data
data(ICFCoreSetCWP)

# adequate coding to get levels 1,..., max
H <- ICFCoreSetCWP[, 1:67] + matrix(c(rep(1, 50), rep(5, 16), 1),
                                    nrow(ICFCoreSetCWP), 67,
                                    byrow = TRUE)

# nonlinear PCA
ordPCA(H, p = 2, lambda = 0.5, maxit = 1000,
       Ks = c(rep(5, 50), rep(9, 16), 5),
       constr = c(rep(TRUE, 50), rep(FALSE, 16), TRUE))


# k-fold cross-validation
set.seed(1234)
lambda <- 10^seq(4,-4, by = -0.1)
cvResult1 <- ordPCA(H, p = 2, lambda = lambda, maxit = 100,
       Ks = c(rep(5, 50), rep(9, 16), 5),
       constr = c(rep(TRUE, 50), rep(FALSE, 16), TRUE),
       CV = TRUE, k = 5)

# optimal lambda
lambda[which.max(apply(cvResult1$VAFtest,2,mean))]

## End(Not run)
```

ICFCoreSetCWP *ICF core set for chronic widespread pain*

Description

The data set contains observed levels of ICF categories from the (comprehensive) ICF Core Set for chronic widespread pain (CWP) and a physical health component summary measure for n = 420 patients.

Usage

```
data(ICFCoreSetCWP)
```

Format

The data frame has 420 rows and 68 columns. The first 67 columns contain observed levels of ICF categories from the (comprehensive) ICF Core Set for chronic widespread pain (CWP). In the last column, the physical health component summary measure is given. Each row corresponds to one patient with CWP. ICF categories have discrete ordinal values between 0 and 4 (columns 1 - 50 and 67), or between -4 and 4 (columns 51 - 66). See the given references for details.

Details

The original data set contained some missing values, which have been imputed using R package `Amelia`.

The data were collected within the study *Validation of ICF Core Sets for chronic conditions*, which was a collaboration effort between the ICF Research Branch of the collaborating centers for the Family of International Classifications in German, the Classification, Terminology and standards Team from the World Health Organization and the International Society for Physical and Rehabilitation Medicine.

Special thanks go to the following participating study centers: Ankara University, Turkey; Azienda Ospedaliera di Sciacca, Italy; Donauspital, Vienna, Austria; Drei-Burgen-Klinik, Bad Muenster, Germany; Edertal Klinik, Bad Wildungen, Germany; Fachklinik Bad Bentheim, Germany; Hospital das Clinicas, School of Medicine, University of Sao Paulo, Brazil; Hospital San Juan Bautista, Catamarca, Argentina; Istituto Scientifico di Montescano, Italy; Istituto Scientifico di Veruno, Italy; Kaiser-Franz-Josef-Spital, Vienna, Austria; Klinik am Regenbogen, Nittenau, Germany; Klinik Bavaria Kreischa, Germany; Klinik Hoher Meissner, Bad Sooden-Allendorf, Germany; Klinikum Berchtesgadener Land, Schoenau, Germany; Kuwait Physical Medicine and Rehabilitation Society, Safat, Kuwait; National Institute for Medical Rehabilitation, Budapest, Hungary; Neuro-Orthopaedisches Krankenhaus und Zentrum fuer Rehabilitative Medizin Soltau, Germany; Praxis fuer Physikalische Medizin und Rehabilitation, Goettingen, Germany; Rehabilitationsklinik Seehof der Bundesversicherungsanstalt fuer Angestellte, Teltow, Germany; Rehaklinik Rheinfelden, Switzerland; Spanish Society of Rheumatology, Madrid, Spain; University Hospital Zurich, Switzerland; University of Santo Tomas, Quelonchy, Philippines.

Most special thanks go to all the patients participating in the study.

If you use the data, please cite the following two references.

References

Cieza, A., G. Stucki, M. Weigl, L. Kullmann, T. Stoll, L. Kamen, N. Kostanjsek, and N. Walsh (2004). *ICF Core Sets for chronic widespread pain.* Journal of Rehabilitation Medicine, Suppl. 44, 63-68.

Gertheiss, J., S. Hogger, C. Oberhauser and G. Tutz (2011). *Selection of ordinally scaled independent variables with applications to international classification of functioning core sets.* Journal of the Royal Statistical Society C (Applied Statistics), 60, 377-395.

Examples

```
# load the data
data(ICFCoreSetCWP)
```

```
# available variables
names(ICFCoreSetCWP)

# adequate coding of x matrix (using levels 1,2,...)
p <- ncol(ICFCoreSetCWP) - 1
n <- nrow(ICFCoreSetCWP)
add <- c(rep(1,50),rep(5,16),1)
add <- matrix(add,n,p,byrow=TRUE)
x <- ICFCoreSetCWP[,1:p] + add

# make sure that also a coefficient is fitted for levels
# that are not observed in the data
addrow <- c(rep(5,50),rep(9,16),5)
x <- rbind(x,addrow)
y <- c(ICFCoreSetCWP$phcs,NA)

# some lambda values
lambda <- c(600,500,400,300,200,100)

# smoothing and selection
modelICF <- ordSelect(x = x, y = y, lambda = lambda)

# results
plot(modelICF)

# plot a selected ICF category (e.g. e1101 'drugs')
# with adequate class labels
plot(modelICF, whx = 51, xaxt = ”n”)
axis(side = 1, at = 1:9, labels = -4:4)
```

`ordAOV` *ANOVA for factors with ordered levels*

Description

This function performs analysis of variance when the factor(s) of interest has/have ordinal scale level. For testing, values from the null distribution are simulated.

Usage

```
ordAOV(x, y, type = c(”RLRT”, ”LRT”), nsim = 10000,
null.sample = NULL, ...)
```

Arguments

`x` a vector or matrix of integers 1,2,... giving the observed levels of the ordinal factor(s). If `x` is a matrix, it is assumed that each column corresponds to one ordinal factor.

`y` the vector of response values.

type the type of test to carry out: likelihood ratio ("LRT") or restricted likelihood ratio ("RLRT").

nsim number of values to simulate from the null distribution.

null.sample a vector, or a list of vectors (in case of multi-factorial ANOVA) containing values already simulated from the null distribution (overrides nsim)

... additional arguments to LRTSim and RLRTSim, respectively.

Details

The method assumes that ordinal factor levels (contained in vector/columns of matrix x) take values 1,2,...,max, where max denotes the highest level of the respective factor observed in the data. Every level between 1 and max has to be observed at least once.

The method uses a mixed effects formulation of the usual one- or multi-factorial ANOVA model (with main effects only) while penalizing (squared) differences of adjacent means. Testing for equal means across factor levels is done by (restricted) likelihood ratio testing for a zero variance component in a linear mixed model. For simulating values from the finite sample null distribution of the (restricted) likelihood ratio statistic, the algorithms implemented in Package RLRsim are used. See LRTSim and RLRTSim for further information.

If x is a vector (or one-column matrix), one-factorial ANOVA is applied, and it is simulated from the exact finite sample null distribution as derived by Crainiceanu & Ruppert (2004). If x is a matrix, multi-factorial ANOVA (with main effects only) is done, and the approximation of the finite sample null distribution proposed by Greven et al. (2008) is used. Simulation studies by Gertheiss (2014) suggest that for ANOVA with ordinal factors RLRT should rather be used than LRT.

Value

In case of one-factorial ANOVA, a list of class htest containing the following components (see also exactLRT and exactRLRT):

statistic the observed (restricted) likelihood ratio.

p p-value for the observed test statistic.

method a character string indicating what type of test was performed and how many values were simulated to determine the critical value.

sample the samples from the null distribution returned by LRTSim and RLRTSim,respectively.

In case of multi-factorial ANOVA, a list (of lists) with the jth component giving the results above when testing the main effect of factor j.

Author(s)

Jan Gertheiss

References

Crainiceanu, C. and D. Ruppert (2004). *Likelihood ratio tests in linear mixed models with one variance component*, Journal of the Royal Statistical Society B, 66, 165-185.

Gertheiss, J. (2014). *ANOVA for factors with ordered levels*, Journal of Agricultural, Biological and Environmental Statistics, 19, 258-277.

Gertheiss, J. and F. Oehrlein (2011). *Testing relevance and linearity of ordinal predictors*, Electronic Journal of Statistics, 5, 1935-1959.

Greven, S., C. Crainiceanu, H. Kuechenhoff, and A. Peters (2008). *Restricted likelihood ratio testing for zero variance components in linear mixed models*, Journal of Computational and Graphical Statistics, 17, 870-891.

Scheipl, F., S. Greven, and H. Kuechenhoff (2008). *Size and power of tests for a zero random effect variance or polynomial regression in additive and linear mixed models*, Computational Statistics & Data Analysis, 52, 3283-3299.

See Also

`LRTSim`, `RLRTSim`

Examples

```
# load some data
data(ICFCoreSetCWP)

# the pysical health component summary
y <- ICFCoreSetCWP$phcs

# consider the first ordinal factor
x <- ICFCoreSetCWP[,1]

# adequate coding
x <- as.integer(x - min(x) + 1)

# ANOVA
ordAOV(x, y, type = "RLRT", nsim=1000000)
```

`ordFusion` *Fusion and selection of dummy coefficients of ordinal predictors*

Description

Fits dummy coefficients of ordinally scaled independent variables with a fused lasso penalty on differences of adjacent dummy coefficients using the `glmpath` algorithm.

Usage

```
ordFusion(x, y, u = NULL, z = NULL, offset = rep(0,length(y)), lambda,
  model = c("linear", "logit", "poisson"), scalex = TRUE,
  nonpenx = NULL, frac.arclength = NULL, ...)
```

Arguments

`x`	the matrix of ordinal predictors, with each column corresponding to one predictor and containing numeric values from {1,2,...}; for each covariate, category 1 is taken as reference category with zero dummy coefficient.
`y`	the response vector.
`u`	a matrix (or `data.frame`) of additional categorical (nominal) predictors, with each column corresponding to one (additional) predictor and containing numeric values from {1,2,...}; corresponding dummy coefficients will not be penalized, and for each covariate category 1 is taken as reference category.
`z`	a matrix (or `data.frame`) of additional metric predictors, with each column corresponding to one (additional) predictor; corresponding coefficients will not be penalized.
`offset`	vector of offset values.
`lambda`	vector of penalty parameters, i.e., lambda values.
`model`	the model which is to be fitted. Possible choices are "linear" (default), "logit" or "poisson". See details below.
`scalex`	logical. Should (split-coded) design matrix corresponding to `x` be scaled to have unit variance over columns before fitting? See details below.
`nonpenx`	vectors of indices indicating columns of `x` whose regression coefficients are not penalized.
`frac.arclength`	just in case the corresponding `glmpath` argument is to be modified; default is `1` for `model == "linear"`, and `0.1` otherwise.
`...`	additional arguments to `glmpath`.

Details

The method assumes that categorical covariates (contained in `x` and `u`) take values 1,2,...,max, where max denotes the (columnwise) highest level observed in the data. If any level between 1 and max is not observed for an ordinal predictor, a corresponding (dummy) coefficient is fitted anyway (by linear interpolation, due to some additional but small quadratic penalty, see `glmpath` for details). If any level > max is not observed but possible in principle, and a corresponding coefficient is to be fitted, the easiest way is to add a corresponding row to `x` (and `u`,`z`) with corresponding `y` value being `NA`.

If a linear regression model is fitted, response vector `y` may contain any numeric values; if a logit model is fitted, `y` has to be 0/1 coded; if a poisson model is fitted, `y` has to contain count data.

If `scalex` is `TRUE`, (split-coded) design matrix constructed from `x` is scaled to have unit variance over columns (see `standardize` argument of `glmpath`).

Value

An `ordPen` object, which is a list containing:

`fitted`	the matrix of fitted response values of the training data. Columns correspond to different `lambda` values.

`coefficients` the matrix of fitted coefficients with respect to dummy-coded (ordinal or nominal) categorical input variables (including the reference category) as well as metric predictors. Columns correspond to different lambda values.

`model` the type of the fitted model: "linear", "logit", or "poisson".

`lambda` the used lambda values.

`xlevels` a vector giving the number of levels of the ordinal predictors.

`ulevels` a vector giving the number of levels of the nominal predictors (if any).

`zcovars` the number of metric covariates (if any).

Author(s)

Jan Gertheiss

References

Gertheiss, J. and G. Tutz (2010). *Sparse modeling of categorial explanatory variables*. The Annals of Applied Statistics, 4, 2150-2180.

Park, M.Y. and T. Hastie (2007). *L1 regularization path algorithm for generalized linear models*. Journal of the Royal Statistical Society B, 69, 659-677.

Tutz, G. and J. Gertheiss (2014). *Rating scales as predictors – the old question of scale level and some answers*. Psychometrica, 79, 357-376.

Tutz, G. and J. Gertheiss (2016). *Regularized regression for categorical data*. Statistical Modelling, 16, 161-200.

See Also

`plot.ordPen`, `predict.ordPen`, `ICFCoreSetCWP`

Examples

```
# fusion and selection of ordinal covariates on a simulated dataset
set.seed(123)

# generate (ordinal) predictors
x1 <- sample(1:8,100,replace=TRUE)
x2 <- sample(1:6,100,replace=TRUE)
x3 <- sample(1:7,100,replace=TRUE)

# the response
y <- -1 + log(x1) + sin(3*(x2-1)/pi) + rnorm(100)

# x matrix
x <- cbind(x1,x2,x3)

# lambda values
lambda <- c(80,70,60,50,40,30,20,10,5,1)

# fusion and selection
```

```
ofu <- ordFusion(x = x, y = y, lambda = lambda)

# results
round(ofu$coef,digits=3)
plot(ofu)

# If for a certain plot the x-axis should be annotated in a different way,
# this can (for example) be done as follows:
plot(ofu, whx = 1, xlim = c(0,9), xaxt = "n")
axis(side = 1, at = c(1,8), labels = c("no agreement","total agreement"))
```

ordGene *Testing for differentially expressed genes*

Description

This function can be used to test for genes that are differentially expressed between levels of an ordinal factor, such as dose levels or ordinal phenotypes.

Usage

```
ordGene(xpr, lvs, type = c("RLRT", "LRT"), nsim = 1e6,
null.sample=NULL, ...)
```

Arguments

`xpr`	a matrix or data frame of gene expression data with Probe IDs as row names.
`lvs`	a numeric vector containing the factor levels (e.g., dose levels) corresponding to the columns of `xpr`.
`type`	the type of test to carry out: likelihood ratio ("LRT") or restricted likelihood ratio ("RLRT").
`nsim`	number of values to simulate from the null distribution.
`null.sample`	a vector containing values already simulated from the null distribution (overrides `nsim`)
`...`	additional arguments to `LRTSim` and `RLRTSim`, respectively.

Details

For each gene in the dataset, `ordAOV` is applied to test for differences between levels given in `lvs`. See `ordAOV` for further information on the testing procedure. Simulation studies by Gertheiss (2014) suggest that a restricted likelihood test (RLRT) should rather be used than a likelihood ratio test (LRT).

In addition to (R)LRT, results of usual one-way ANOVA (not taking the factor's ordinal scale level into account) and a t-test assuming a linear trend across factor levels are reported. Note that the t-test does not assume linearity in the doses (such as 0, 0.5, 2.0, 5.0, ...), if given, but in the levels, i.e., 1, 2, 3, etc.

Value

A matrix containing the raw p-values for each gene (rows) when using (R)LRT, ANOVA or a t-test (columns).

Author(s)

Jan Gertheiss

References

Crainiceanu, C. and D. Ruppert (2004). *Likelihood ratio tests in linear mixed models with one variance component*, Journal of the Royal Statistical Society B, 66, 165-185.

Gertheiss, J. (2014). *ANOVA for factors with ordered levels*, Journal of Agricultural, Biological and Environmental Statistics, 19, 258-277.

Gertheiss, J. and F. Oehrlein (2011). *Testing relevance and linearity of ordinal predictors*, Electronic Journal of Statistics, 5, 1935-1959.

Sweeney, E., C. Crainiceanu, and J. Gertheiss (2015). *Testing differentially expressed genes in dose-response studies and with ordinal phenotypes*, Statistical Applications in Genetics and Molecular Biology, 15, 213-235.

See Also

ordAOV

Examples

```
## Not run:
# generate toy gene expression data
set.seed(321)
ni <- 5
n <- sum(5*ni)
xpr <- matrix(NA, ncol = n, nrow = 100)
mu_lin <- 3:7
mu_sq2 <- (-2:2)^2 * 0.5 + 3
a <- seq(0.75, 1.25, length.out = 10)

for(i in 1:10){
  xpr[i,] <- a[i] * rep(mu_lin, each = ni) + rnorm(n)
  xpr[i+10,] <- a[i] * rep(mu_sq2, each = ni) + rnorm(n)
}
for(i in 21:100) xpr[i,] <- 3 + rnorm(n)

dose <- rep(c(0,0.01,0.05,0.2,1.5), each = ni)

# continuous representation
oldpar <- par(mfrow = c(2,2))
plot(dose, xpr[4,], col = as.factor(dose), lwd = 2, ylab = "expression", main = "gene 4")
lines(sort(unique(dose)), mu_lin * a[4], lty = 1, col = 1)
plot(dose, xpr[14,], col = as.factor(dose), lwd = 2, ylab = "expression", main = "gene 14")
lines(sort(unique(dose)), mu_sq2 * a[4], lty = 1, col = 1)
```

```
# dose on ordinal scale
plot(1:length(sort(unique(dose))), ylim = range(xpr[4,]), pch = "", ylab = "expression",
     xlab = "levels", xaxt="n")
axis(1, at = 1:length(sort(unique(dose))) )
points(as.factor(dose), xpr[4,], col=as.factor(dose), lwd = 2)
lines(1:length(sort(unique(dose))), mu_lin * a[4], lty = 1)
plot(1:length(sort(unique(dose))), ylim = range(xpr[14,]), pch = "", ylab = "expression",
     xlab = "levels", xaxt="n")
axis(1, at = 1:length(sort(unique(dose))) )
points(as.factor(dose), xpr[14,], col=as.factor(dose), lwd = 2)
lines(1:length(sort(unique(dose))), mu_sq2 * a[4], lty = 1)
par(oldpar)

# calculate p-values
library(ordPens)
pvals <- ordGene(xpr = xpr, lvs = dose, nsim = 1e6)

# compare distribution of (small) p-values
plot(ecdf(pvals[,1]), xlim = c(0,0.05), ylim = c(0, 0.25),
     main = "", xlab = "p-value", ylab = "F(p-value)")
plot(ecdf(pvals[,2]), xlim = c(0, 0.05), add = TRUE, col = 2)
plot(ecdf(pvals[,3]), xlim = c(0, 0.05), add = TRUE, col = 3)
legend('topleft', colnames(pvals), col = 1:3, lwd = 2, lty = 1)

## End(Not run)
```

ordPCA *Penalized nonlinear PCA for ordinal variables*

Description

This function performs nonlinear principal components analysis when the variables of interest have ordinal level scale using a second-order difference penalty.

Usage

```
ordPCA(H, p, lambda = c(1), maxit = 100, crit = 1e-7, qstart = NULL,
       Ks = apply(H,2,max), constr = rep(FALSE, ncol(H)), trace = FALSE,
       CV = FALSE, k = 5, CVfit = FALSE)
```

Arguments

H	a matrix or data frame of of integers 1,2,... giving the observed levels of the ordinal variables; provides the data for the principal components analysis.
p	the number of principal components to be extracted.
lambda	a numeric value or a vector (in decreasing order) defining the amount of shrinkage; defaults to 1.

`maxit` the maximum number of iterations; defaults to 100.

`crit` convergence tolerance; defaults to 1e-7.

`qstart` optional list of quantifications for the initial linear PCA.

`Ks` a vector containing the highest level of each variable.

`constr` a logical vector specifying whether monotonicity constraints should be applied to the variables.

`trace` logical; if `TRUE`, tracing information on the progress of the optimization is produced in terms of VAF in each iteration.

`CV` a logical value indicating whether k-fold cross-validation should be performed in order to evaluate the performance and/or select an optimal smoothing parameter.

`k` the number of folds to be specified; only if `CV` is set to `TRUE`.

`CVfit` logical; to be specified only if `CV = TRUE`. If `CVfit = TRUE` and `lambda` is a vector of length > 5, additional yes/no dialog appears; if `FALSE`, only VAF values are provided (recommended); else, also lists of matrices of PCA results are produced and stored.

Details

In order to respect the ordinal scale of the data, principal components analysis is not applied to data matrix `H` itself, but to newly constructed variables by assigning numerical values – the quantifications – to the categories via penalized, optimal scaling/scoring. The calculation is done by alternately cycling through data scoring and PCA until convergence.

The penalty parameter controls the amount of shrinkage: For `lambda = 0`, purely nonlinear PCA via standard, optimal scaling is obtained. As `lambda` becomes very large, the quantifications are shrunken towars linearity, i.e., usual PCA is applied to levels 1,2,... ignoring the ordinal scale level of the variables.

Note that optimization starts with the first component of `lambda`. Thus, if `lambda` is not in decreasing order, the vector will be sorted internally and so will be corresponding results.

In case of cross-validation, for each `lambda` the proportion of variance accounted for (VAF) is given for both the training and test data (see below).

Value

A List with components:

`qs` a list of quantifications, if `lambda` is specified as a single value. Otherwise, a list of matrices, each column corresponding to a certain `lambda` value.

`Q` data matrix after scaling, if `lambda` is scalar. Otherwise, a list of matrices with each list entry corresponding to a certain `lambda` value.

`X` matrix of factor values resulting from `prcomp`, if `lambda` is scalar. Otherwise, list of matrices.

`A` loadings matrix as a result from `prcomp`, if `lambda` is scalar. Otherwise, list of matrices.

`iter` number of iterations used.

`pca` object of class "`prcomp`" returned by `prcomp`.

`trace` vector of VAF values in each iteration, if `lambda` is specified as a single value. Otherwise, a list of vectors, each entry corresponding to a certain `lambda` value.

`VAFtrain` matrix with columns corresponding to `lambda` and rows corresponding to the folds `k`. Contains corresponding proportions of variance accounted for (VAF) on the training data within cross-validation. VAF here is defined in terms of the proportion of variance explained by the first `p` PCs.

`VAFtest` VAF matrix for the test data within cross-validation.

If cross-validation is desired, the pca results are stored in a list called `fit` with each list entry corresponding to a certain fold. Within such a list entry, all sub entries can be accessed as described above. However, VAF values are stored in `VAFtrain` or `VAFtest` and can be accessed directly.

Author(s)

Aisouda Hoshiyar, Jan Gertheiss

References

Hoshiyar, A. (2020). *Analyzing Likert-type data using penalized non-linear principal components analysis*, in: Proceedings of the 35th International Workshop on Statistical Modelling, Vol. I, 337-340.

Hoshiyar, A., H.A.L. Kiers, and J. Gertheiss (2021). *Penalized non-linear principal components analysis for ordinal variables with an application to international classification of functioning core sets*, Preprint.

Linting, M., J.J. Meulmann, A.J. von der Kooji, and P.J.F. Groenen (2007). *Nonlinear principal components analysis: Introduction and application*, Psychological Methods, 12, 336-358.

See Also

`prcomp`

Examples

```
## Not run:
## load ICF data
data(ICFCoreSetCWP)

# adequate coding to get levels 1,..., max
H <- ICFCoreSetCWP[, 1:67] + matrix(c(rep(1, 50), rep(5, 16), 1),
                                    nrow(ICFCoreSetCWP), 67,
                                    byrow = TRUE)
xnames <- colnames(H)

# nonlinear PCA
icf_pca1 <- ordPCA(H, p = 2, lambda = c(5, 0.5, 0.0001), maxit = 1000,
                   Ks = c(rep(5, 50), rep(9, 16), 5),
                   constr = c(rep(TRUE, 50), rep(FALSE, 16), TRUE))

# estimated quantifications
```

```
icf_pca1$qs[[55]]

plot(1:9, icf_pca1$qs[[55]][,1], type="b",
xlab="category", ylab="quantification", col=1, main=xnames[55],
ylim=range(c(icf_pca1$qs[[55]][,1],icf_pca1$qs[[55]][,2],icf_pca1$qs[[55]][,3])))
lines(icf_pca1$qs[[55]][,2], type = "b", col = 2, lty = 2, pch = 2, lwd=2)
lines(icf_pca1$qs[[55]][,3], type = "b", col = 3, lty = 3, pch = 3, lwd=2)

# compare VAF
icf_pca2 <- ordPCA(H, p = 2, lambda = c(5, 0.5, 0.0001), maxit = 1000,
                   Ks = c(rep(5, 50), rep(9, 16), 5),
                   constr = c(rep(TRUE, 50), rep(FALSE, 16), TRUE),
                   CV = TRUE, k = 5)
icf_pca2$VAFtest

## load ehd data
require(psy)
data(ehd)

# recoding to get levels 1,..., max
H <- ehd + 1

# nonlinear PCA
ehd1 <- ordPCA(H, p = 5, lambda = 0.5, maxit = 100,
               constr = rep(TRUE,ncol(H)),
               CV = FALSE)

# resulting PCA on the scaled variables
summary(ehd1$pca)

# plot quantifications
oldpar <- par(mfrow = c(4,5))
for(j in 1:length(ehd1$qs))
  plot(1:5, ehd1$qs[[j]], type = "b", xlab = "level", ylab = "quantification",
  main = colnames(H)[j])
par(oldpar)

# include cross-validation
lambda <- 10^seq(4,-4, by = -0.1)
set.seed(456)
cvResult <- ordPCA(H, p = 5, lambda = lambda, maxit = 100,
                    constr = rep(TRUE,ncol(H)),
                    CV = TRUE, k = 5, CVfit = FALSE)
# optimal lambda
lambda[which.max(apply(cvResult$VAFtest,2,mean))]

## End(Not run)
```

ordSelect *Selection and smoothing of dummy coefficients of ordinal predictors*

Description

Fits dummy coefficients of ordinally scaled independent variables with a group lasso penalty on differences of adjacent dummy coefficients.

Usage

```
ordSelect(x, y, u = NULL, z = NULL, offset = rep(0,length(y)), lambda,
  model = c(”linear”, ”logit”, ”poisson”), penscale = sqrt, scalex = TRUE,
  nonpenx = NULL, eps = 1e-3, ...)
```

Arguments

`x` the matrix of ordinal predictors, with each column corresponding to one predictor and containing numeric values from {1,2,...}; for each covariate, category 1 is taken as reference category with zero dummy coefficient.

`y` the response vector.

`u` a matrix (or `data.frame`) of additional categorical (nominal) predictors, with each column corresponding to one (additional) predictor and containing numeric values from {1,2,...}; corresponding dummy coefficients will not be penalized, and for each covariate category 1 is taken as reference category.

`z` a matrix (or `data.frame`) of additional metric predictors, with each column corresponding to one (additional) predictor; corresponding coefficients will not be penalized.

`offset` vector of offset values.

`lambda` vector of penalty parameters (in decreasing order). Optimization starts with the first component. See details below.

`model` the model which is to be fitted. Possible choices are "linear" (default), "logit" or "poisson". See details below.

`penscale` rescaling function to adjust the value of the penalty parameter to the degrees of freedom of the parameter group. See the references below.

`scalex` logical. Should (split-coded) design matrix corresponding to `x` be scaled to have unit variance over columns before fitting? See details below.

`nonpenx` vectors of indices indicating columns of `x` whose regression coefficients are not penalized.

`eps` a (small) constant to be added to the columnwise standard deviations when scaling the design matrix, to control the effect of very small stds. See details below.

`...` additional arguments.

Details

The method assumes that categorical covariates (contained in `x` and `u`) take values 1,2,...,max, where max denotes the (columnwise) highest level observed in the data. If any level between 1 and max is not observed for an ordinal predictor, a corresponding (dummy) coefficient is fitted anyway. If any level > max is not observed but possible in principle, and a corresponding coefficient is to be fitted, the easiest way is to add a corresponding row to `x` (and `u`,`z`) with corresponding `y` value being `NA`.

If a linear regression model is fitted, response vector `y` may contain any numeric values; if a logit model is fitted, `y` has to be 0/1 coded; if a poisson model is fitted, `y` has to contain count data.

If `scalex` is `TRUE`, (split-coded) design matrix constructed from `x` is scaled to have unit variance over columns. If a certain `x`-category, however, is observed only a few times, variances may become very small and scaling has enormous effects on the result and may cause numerical problems. Hence a small constant `eps` can be added to each standard deviation when used for scaling.

Value

An `ordPen` object, which is a list containing:

`fitted` the matrix of fitted response values of the training data. Columns correspond to different `lambda` values.

`coefficients` the matrix of fitted coefficients with respect to dummy-coded (ordinal or nominal) categorical input variables (including the reference category) as well as metric predictors. Columns correspond to different lambda values.

`model` the type of the fitted model: "linear", "logit", or "poisson".

`lambda` the used lambda values.

`fraction` the used fraction values (`NULL` in case of `ordSelect`).

`xlevels` a vector giving the number of levels of the ordinal predictors.

`ulevels` a vector giving the number of levels of the nominal predictors (if any).

`zcovars` the number of metric covariates (if any).

Author(s)

Jan Gertheiss

References

Gertheiss, J., S. Hogger, C. Oberhauser and G. Tutz (2011). *Selection of ordinally scaled independent variables with applications to international classification of functioning core sets*. Journal of the Royal Statistical Society C (Applied Statistics), 60, 377-395.

Meier, L., S. van de Geer and P. Buehlmann (2008). *The group lasso for logistic regression*. Journal of the Royal Statistical Society B, 70, 53-71.

Tutz, G. and J. Gertheiss (2014). *Rating scales as predictors – the old question of scale level and some answers*. Psychometrika, 79, 357-376.

Tutz, G. and J. Gertheiss (2016). *Regularized regression for categorical data*. Statistical Modelling, 16, 161-200.

Yuan, M. and Y. Lin (2006). *Model selection and estimation in regression with grouped variables*. Journal of the Royal Statistical Society B, 68, 49-67.

See Also

`plot.ordPen`, `predict.ordPen`, `ICFCoreSetCWP`

Examples

```
# smoothing and selection of ordinal covariates on a simulated dataset
set.seed(123)

# generate (ordinal) predictors
x1 <- sample(1:8,100,replace=TRUE)
x2 <- sample(1:6,100,replace=TRUE)
x3 <- sample(1:7,100,replace=TRUE)

# the response
y <- -1 + log(x1) + sin(3*(x2-1)/pi) + rnorm(100)

# x matrix
x <- cbind(x1,x2,x3)

# lambda values
lambda <- c(1000,500,200,100,50,30,20,10,1)

# smoothing and selection
osl <- ordSelect(x = x, y = y, lambda = lambda)

# results
round(osl$coef,digits=3)
plot(osl)

# If for a certain plot the x-axis should be annotated in a different way,
# this can (for example) be done as follows:
plot(osl, whx = 1, xlim = c(0,9), xaxt = "n")
axis(side = 1, at = c(1,8), labels = c("no agreement","total agreement"))
```

`ordSmooth` *Smoothing dummy coefficients of ordinal predictors*

Description

Fits dummy coefficients of ordinally scaled independent variables with the sum of squared differences of adjacent dummy coefficients being penalized.

Usage

```
ordSmooth(x, y, u = NULL, z = NULL, offset = rep(0,length(y)), lambda,
  model = c("linear", "logit", "poisson"), penscale = identity, scalex = TRUE,
  nonpenx = NULL, eps = 1e-3, delta = 1e-6, maxit = 25, ...)
```

Arguments

`x` the matrix (or `data.frame`) of ordinal predictors, with each column corresponding to one predictor and containing numeric values from {1,2,...}; for each covariate, category 1 is taken as reference category with zero dummy coefficient.

`y` the response vector.

`u` a matrix (or `data.frame`) of additional categorical (nominal) predictors, with each column corresponding to one (additional) predictor and containing numeric values {1,2,...}; corresponding dummy coefficients will not be penalized, and for each covariate category 1 is taken as reference category.

`z` a matrix (or `data.frame`) of additional metric predictors, with each column corresponding to one (additional) predictor; corresponding coefficients will not be penalized.

`offset` vector of offset values.

`lambda` vector of penalty parameters (in decreasing order). Optimization starts with the first component. See details below.

`model` the model which is to be fitted. Possible choices are "linear" (default), "logit" or "poisson". See details below.

`penscale` rescaling function to adjust the value of the penalty parameter to the degrees of freedom of the parameter group.

`scalex` logical. Should (split-coded) design matrix corresponding to `x` be scaled to have unit variance over columns before fitting? See details below.

`nonpenx` vector of indices indicating columns of `x` whose regression coefficients are not penalized.

`eps` a (small) constant to be added to the columnwise standard deviations when scaling the design matrix, to control the effect of very small stds. See details below.

`delta` a small positive convergence tolerance which is used as stopping criterion for the penalized Fisher scoring when a logit or poisson model is fitted. See details below.

`maxit` integer given the maximal number of (penalized) Fisher scoring iterations.

`...` additional arguments.

Details

The method assumes that categorical covariates (contained in `x` and `u`) take values 1,2,...,max, where max denotes the (columnwise) highest level observed in the data. If any level between 1 and max is not observed for an ordinal predictor, a corresponding (dummy) coefficient is fitted anyway. If any level > max is not observed but possible, and a corresponding coefficient is to be fitted, the easiest way is to add a corresponding row to `x` (and `u`,`z`) with corresponding `y` value being `NA`.

If a linear regression model is fitted, response vector `y` may contain any numeric values; if a logit model is fitted, `y` has to be 0/1 coded; if a poisson model is fitted, `y` has to contain count data.

If `scalex` is `TRUE`, (split-coded) design matrix constructed from `x` is scaled to have unit variance over columns. If a certain `x`-category, however, is observed only a few times, variances may become very small and scaling has enormous effects on the result and may cause numerical problems. Hence a small constant `eps` can be added to each standard deviation when used for scaling.

A logit or poisson model is fitted by penalized Fisher scoring. For stopping the iterations the criterion `sqrt(sum((b.new-b.old)^2)/sum(b.old^2))` < `delta` is used.

Please note, `ordSmooth` is intended for use with high-dimensional ordinal predictors; more precisely, if the number of ordinal predictors is large. Package `ordPens`, however, also includes auxiliary functions such that `gam` from `mgcv` can be used for fitting generalized linear and additive models with first- and second-order ordinal smoothing penalty as well as built-in smoothing parameter selection. In addition, `mgcv` tools for further statistical inference can be used. Note, however, significance of smooth (ordinal) terms is only reliable in case of the second-order penalty. Also note, if using `gam`, dummy coefficients/fitted functions are centered over the data observed. For details, please see Gertheiss et al. (2021) and examples below.

Value

An `ordPen` object, which is a list containing:

`fitted` the matrix of fitted response values of the training data. Columns correspond to different `lambda` values.

`coefficients` the matrix of fitted coefficients with respect to dummy-coded (ordinal or nominal) categorical input variables (including the reference category) as well as metric predictors. Columns correspond to different lambda values.

`model` the type of the fitted model: "linear", "logit", or "poisson".

`lambda` the used lambda values.

`fraction` the used fraction values (`NULL` in case of `ordSmooth`).

`xlevels` a vector giving the number of levels of the ordinal predictors.

`ulevels` a vector giving the number of levels of the nominal predictors (if any).

`zcovars` the number of metric covariates (if any).

Author(s)

Jan Gertheiss

References

Gertheiss, J., F. Scheipl, T. Lauer, and H. Ehrhardt (2021). *Statistical inference for ordinal predictors in generalized linear and additive models with application to bronchopulmonary dysplasia*. Preprint, available from https://arxiv.org/abs/2102.01946.

Gertheiss, J. and G. Tutz (2009). *Penalized regression with ordinal predictors*. International Statistical Review, 77, 345-365.

Tutz, G. and J. Gertheiss (2014). *Rating scales as predictors – the old question of scale level and some answers*. Psychometrica, 79, 357-376.

Tutz, G. and J. Gertheiss (2016). *Regularized regression for categorical data*. Statistical Modelling, 16, 161-200.

See Also

`plot.ordPen`, `predict.ordPen`

Examples

```
# smooth modeling of a simulated dataset
set.seed(123)

# generate (ordinal) predictors
x1 <- sample(1:8,100,replace=TRUE)
x2 <- sample(1:6,100,replace=TRUE)
x3 <- sample(1:7,100,replace=TRUE)

# the response
y <- -1 + log(x1) + sin(3*(x2-1)/pi) + rnorm(100)

# x matrix
x <- cbind(x1,x2,x3)

# lambda values
lambda <- c(1000,500,200,100,50,30,20,10,1)

# smooth modeling
osm1 <- ordSmooth(x = x, y = y, lambda = lambda)

# results
round(osm1$coef,digits=3)
plot(osm1)

# If for a certain plot the x-axis should be annotated in a different way,
# this can (for example) be done as follows:
plot(osm1, whx = 1, xlim = c(0,9), xaxt = "n")
axis(side = 1, at = c(1,8), labels = c("no agreement","total agreement"))

# add a nominal covariate to control for
u1 <- sample(1:8,100,replace=TRUE)
u <- cbind(u1)
osm2 <- ordSmooth(x = x, y = y, u = u, lambda = lambda)
round(osm2$coef,digits=3)


## Use gam() from mgcv for model fitting:
# ordinal predictors need to be ordered factors
x1 <- as.ordered(x1)
x2 <- as.ordered(x2)
x3 <- as.ordered(x3)

# model fitting with first-order penalty and smoothing parameter selection by REML
gom1 <- gam(y ~ s(x1, bs = "ordinal", m = 1) + s(x2, bs = "ordinal", m = 1) +
s(x3, bs = "ordinal", m = 1) + factor(u1), method = "REML")

# plot with confidence intervals
plot(gom1)

# use second-order penalty instead
gom2 <- gam(y ~ s(x1, bs = "ordinal", m = 2) + s(x2, bs = "ordinal", m = 2) +
s(x3, bs = "ordinal", m = 2) + factor(u1), method = "REML")

# summary including significance of smooth terms
# please note, the latter is only reliable for m = 2
summary(gom2)

# plotting
plot(gom2)
```

plot.ordPen *Plot method for ordPen objects*

Description

Takes a fitted ordPen object and plots estimated dummy coefficients of ordinal predictors for different lambda values.

Usage

```
## S3 method for class 'ordPen'
plot(x, whl = NULL, whx = NULL,
  type = NULL, xlab = NULL, ylab = NULL, main = NULL,
  xlim = NULL, ylim = NULL, col = NULL, ...)
```

Arguments

x — an ordPen object.

whl — a vector of indices of lambda values corresponding to object$lambda for which plotting is done; if NULL, all values from object$lambda are considered.

whx — a vector of indices indicating the ordinal predictors whose dummy coefficients are plotted; e.g., set whx=2, if you just want the plot for the second smooth term.

type — 1-character string giving the type of plot desired, see plot.default.

xlab — a label for the x axis; if supplied then this will be used as the x label for all plots.

ylab — a label for the y axis; if supplied then this will be used as the y label for all plots.

main — a main title for the plot(s); if supplied then this will be used as the title for all plots.

xlim — the x limits; if supplied then this pair of numbers are used as the x limits for each plot.

ylim — the y limits; if supplied then this pair of numbers are used as the y limits for each plot.

col — the plotting color; can be a vector of the same length as whl specifying different colors for different lambda values. Default is shades of gray: the higher lambda the darker.

... — additional graphical parameters (see plot.default, or par).

Value

The function simply generates plots.

Author(s)

Jan Gertheiss

See Also

ordFusion, ordSelect, ordSmooth

Examples

```
# see for example
help(ordSelect)
```

---

predict.ordPen *Predict method for ordPen objects*

Description

Obtains predictions from an ordPen object.

Usage

```
## S3 method for class 'ordPen'
predict(object, newx, newu = NULL, newz = NULL,
  offset = rep(0,nrow(as.matrix(newx))),
  type = c("link", "response"), ...)
```

Arguments

object	an ordPen object.
newx	the matrix (or data.frame) of new observations of the considered ordinal predictors, with each column corresponding to one predictor and containing numeric values from {1,2,...}.
newu	a matrix (or data.frame) of new observations of the additional categorical (nominal) predictors, with each column corresponding to one (additional) predictor and containing numeric values {1,2,...}.
newz	a matrix (or data.frame) of new observations of the additional metric predictors, with each column corresponding to one (additional) predictor.
offset	potential offset values.
type	the type of prediction; type = "link" is on the scale of linear predictors, whereas type = "response" is on the scale of the response variable, i.e., type = "response" applies the inverse link function to the linear predictors.
...	additional arguments (not supported at this time).

Value

A matrix of predictions whose columns correspond to the different values of the penalty parameter `lambda` of the `ordPen` object.

Author(s)

Jan Gertheiss

See Also

`ordSelect`, `ordSmooth`, `ordFusion`

Examples

```
# the training data
set.seed(123)

# generate (ordinal) predictors
x1 <- sample(1:8,100,replace=TRUE)
x2 <- sample(1:6,100,replace=TRUE)
x3 <- sample(1:7,100,replace=TRUE)

# the response
y <- -1 + log(x1) + sin(3*(x2-1)/pi) + rnorm(100)

# x matrix
x <- cbind(x1,x2,x3)

# lambda values
lambda <- c(1000,500,200,100,50,30,20,10,1)

# selecting and/or smoothing/fusing
o1 <- ordSmooth(x = x, y = y, lambda = lambda)
o2 <- ordSelect(x = x, y = y, lambda = lambda)
o3 <- ordFusion(x = x, y = y, lambda = lambda)

# new data
x1 <- sample(1:8,10,replace=TRUE)
x2 <- sample(1:6,10,replace=TRUE)
x3 <- sample(1:7,10,replace=TRUE)
newx <- cbind(x1,x2,x3)

# prediction
round(predict(o1, newx), digits=3)
round(predict(o2, newx), digits=3)
round(predict(o3, newx), digits=3)
```

Index