Package 'serp'

September 2, 2021

Type Package

Title Smooth Effects on Response Penalty for 'CLM'

Version 0.2.1

Description A regularization method for the cumulative link models. The 'smooth-effect-on-response penalty' ('SERP') provides flexible modelling of the ordinal model by enabling the smooth transition from the general cumulative link model to a coarser form of the same model. In other words, as the tuning parameter goes from zero to infinity, the subject-specific effects associated with each variable in the model tend to a unique global effect. The parameter estimates of the general cumulative model are mostly unidentifiable or at least only identifiable within a range of the entire parameter space. Thus, by maximizing a penalized rather than the usual non-penalized log-likelihood, this and other numerical problems common with the general model are to a large extent eliminated. Fitting is via a modified Newton's method. Several standard model performance and descriptive methods are also available. For more details on the penalty implemented here, see, 'Ugba et al. (2021)' <doi:10.3390/stats4030037> and Tutz and Gertheiss (2016) <doi:10.1177/1471082X16642560>.

License GPL-2 | file LICENSE

URL https://github.com/ejikeugba/serp

BugReports https://github.com/ejikeugba/serp/issues

Depends R (>= 3.2.0)

Imports ordinal (>= 2016-12-12), stats

Suggests covr, testthat, VGAM (>= 1.1-4)

Encoding UTF-8

LazyData True

RoxygenNote 7.1.1

NeedsCompilation no

Author Ejike R. Ugba [aut, cre, cph] (<https://orcid.org/0000-0003-2572-0023>)

Maintainer Ejike R. Ugba <ejike.ugba@outlook.com>

Repository CRAN

Date/Publication 2021-09-01 22:10:02 UTC

R topics documented:

`AIC.serp` *AIC for a fitted serp object*

Description

Returns the akaike information criterion of a fitted object of class `serp`. For the penalized slope, the effective degrees of freedom (edf) is obtained from the trace of the generalized hat matrix which depends on the tuning parameter.

Usage

```
## S3 method for class 'serp'
AIC(object, ..., k = 2)
```

Arguments

`object`	An object of class `serp`.
`...`	additional arguments.
`k`	fixed value equal to 2.

Value

A single numeric value of the model AIC.

See Also

serp, BIC.serp, coef.serp, logLik.serp,

Examples

```
library(serp)
m <- serp(rating ~ temp + contact, slope = "parallel", link = "probit",
          data = wine)
AIC(m)
```

---

anova.serp *ANOVA method for a fitted serp object*

Description

Provides a likelihood ratio test for comparing two or more serp objects. This does not currently support model(s) with penalized slope.

Usage

```
## S3 method for class 'serp'
anova(object, ..., test = c("Chisq", "none"))
```

Arguments

object	An object of class serp.
...	additional arguments.
test	type of test to be conducted.

Value

An ANOVA table with the following components on display:

model the respective model aliases.

slope type of slope fitted, which may be any of, unparallel, parallel, or partial slope.

no.par the no of parameters in the model.

AIC the akaike information criterion.

logLik the realized log-likelihood.

Test the different pair(s) of test(s) conducted.

LR.stat the computed Likelihood ratio statistic.

df the degree of freedom.

Pr(chi) the p-value of test statitic.

See Also

serp, confint.serp, vcov.serp, errorMetrics

Examples

```
library(serp)
m1 <- serp(rating ~ temp + contact, slope = "parallel", link = "logit",
           data = wine)
m2 <- update(m1, ~ contact)
anova(m1, m2)
```

`BIC.serp` *BIC for a fitted serp object*

Description

Returns the bayesian information criterion of a fitted object of class `serp`. For the penalized slopes, the effective degrees of freedom (edf) is obtained from the trace of the generalized hat matrix which depends on the tuning parameter.

Usage

```
## S3 method for class 'serp'
BIC(object, ...)
```

Arguments

`object`	An object of class `serp`.
`...`	additional arguments.

Value

A single numeric value of the model.

See Also

`serp`, `AIC.serp`, `coef.serp`, `logLik.serp`,

Examples

```
library(serp)
m <- serp(rating ~ temp + contact, slope = "parallel", link = "loglog",
          data = wine)
BIC(m)
```

coef.serp *Coefficients for a fitted serp object*

Description

Returns the coefficients of a fitted object of class serp.

Usage

```
## S3 method for class 'serp'
coef(object, ...)
```

Arguments

object	An object of class serp.
...	additional arguments.

Value

A vector of model coefficients.

See Also

serp, AIC.serp, BIC.serp, logLik.serp

Examples

```
library(serp)
m <- serp(rating ~ temp + contact, slope = "parallel", link = "loglog",
          data = wine)
coef(m)
```

confint.serp *Confidence interval for a fitted serp object*

Description

Provides the confidence interval of estimates for an object of class serp.

Usage

```
## S3 method for class 'serp'
confint(object, ..., parm, level = 0.95)
```

Arguments

object	An object of class serp.
...	additional arguments.
parm	unused argument.
level	significance level.

Value

A matrix of the the confidence intervals of fitted model.

See Also

serp, anova.serp, vcov.serp, errorMetrics

Examples

```
library(serp)
m <- serp(rating ~ temp + contact, slope = "parallel", link = "logit",
          data = wine)
confint(m)
```

---

errorMetrics *Performance metrics for categorical models*

Description

Calculates performance metrics of fitted categorical models, including binary and multi-categorical models. Metrics calculated include the brier score, log loss and misclassification error.

Usage

```
errorMetrics(
             actual,
             predicted,
             model= c("multiclass", "binary"),
             type= c("brier", "logloss", "misclass"),
             eps=.Machine$double.eps)
```

Arguments

actual	vector of actual values observed
predicted	predicted probability matrix of a categorical model or a vector of fitted values for binary models.
model	specifies whether multi-categorical or binary model
type	specifies type of error metrics
eps	a near-zero value introduced only if the fitted probabilities go beyond a specified threshold. It helps to minimize the chances of running into numerical problems.

Value

A numeric value of computed performance metric determining how good a categorical model is compare to competing models.

brier the brier score of fitted model.

logloss the logloss of fitted model.

misclass the misclassification error of fitted model.

See Also

serp, anova.serp, confint.serp, vcov.serp

Examples

```
f1 <- serp(rating ~ temp + contact, tuneMethod = "user",
slope = "penalize", lambda = 0.3, reverse = TRUE, link = "logit",
data = wine)
errorMetrics(f1, type = "brier")
errorMetrics(f1, type = "logloss")
errorMetrics(f1, type = "misclass")

## For non-serp object of class, 'actual'  and 'predicted' values
## must be provided
set.seed(1)
y <- as.factor(rbinom(50,1,0.5))
xx <- runif(50)
f2 <- glm(y ~ xx, family = binomial(link="logit"))
p2 <- f2$fitted.values

errorMetrics(actual=y, predicted=p2, model= "binary", type = "brier")
errorMetrics(actual=y, predicted=p2, model= "binary", type = "logloss")
errorMetrics(actual=y, predicted=p2, model= "binary", type = "misclass")
```

---

logLik.serp *Log-likelihood for a fitted serp object*

Description

Returns the Log-likelihood for a fitted object of class serp.

Usage

```
## S3 method for class 'serp'
logLik(object, ...)
```

Arguments

object	An object of class serp.
...	additional arguments.

Value

A single numeric value of model log-likelihood

See Also

serp, AIC.serp, BIC.serp, coef.serp

Examples

```
library(serp)
m <- serp(rating ~ temp + contact, slope = "parallel", link = "loglog",
          data = wine)
logLik(m)
```

---

predict.serp *Prediction from fitted serp model*

Description

This function takes a fitted serp object produced by serp() and produces predicted values. Type of predictions returned include response, link and class. Prediction is also possible with new set of values having the same column names as in the original values used for the model fit.

Usage

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

Arguments

object	An object of class serp.
type	could be any of these: response, link or terms.
newdata	fresh dataset with all relevant variables.
...	additional arguments.

Value

A vector of predicted classes with type equal to 'class' or a dataframe of predicted values for type equal to 'response' and 'link'.

See Also

anova.serp, summary.serp, confint.serp, vcov.serp, errorMetrics

Examples

```
library(serp)
m <- serp(rating ~ temp + contact, slope = "penalize",
          reverse = TRUE, link = "logit", tuneMethod = "user",
          lambda = 1, data = wine)

head(predict(m, type = "link"))
head(predict(m, type = "response"))
predict(m, type = "class")

n.wine <- wine[1:20,]
predict(m, newdata = n.wine, type = "class")
```

---

print.serp *Print method for a fitted serp object*

Description

Prints out a vector of coefficients of the fitted model with some additional goodness-of-fit measures.

Usage

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

Arguments

x	An object of class serp.
...	additional arguments.

Value

No return value

See Also

serp, print.summary.serp

print.summary.serp *Print method for an object of class summary.serp*

Description

Prints out the information supplied via `summary.serp` method.

Usage

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

Arguments

`x`	An object of class `summary.serp`.
`...`	additional arguments.

Value

No return value

See Also

`serp`, `print.serp`

serp *Smooth Effects on Response Penalty for CLM*

Description

Fits cumulative link models (CLMs) with the smooth-effect-on-response penalty (SERP) via a modified Newton-Raphson algorithm. SERP enables the regularization of the parameter space between the general and the restricted cumulative models, with a resultant shrinkage of all subject-specific effects to global effects. The Akaike information critrion (`aic`), K-fold cross validation (`cv`), among other tuning aproaches, provide the means of arriving at an optimal tuning parameter in a in a situation where a user-supplied tuning value is not available. The `slope` argument allows for the selection of a penalized, unparallel, parallel, or partial slope.

Usage

```
serp(
     formula,
     link = c("logit", "probit","loglog", "cloglog", "cauchit"),
     slope = c("penalize", "parallel", "unparallel", "partial"),
     tuneMethod = c("aic", "cv", "finite", "user"),
     reverse = FALSE,
     lambdaGrid = NULL,
     cvMetric = c("brier", "logloss", "misclass"),
     gridType = c("discrete", "fine"),
     globalEff = NULL,
     data,
     subset,
     weights = NULL,
     weight.type = c("analytic", "frequency"),
     na.action = NULL,
     lambda = NULL,
     contrasts = NULL,
     control = list(),
     ...)
```

Arguments

formula	regression formula of the form: response ~ predictors. The response should be a factor (ordered).
link	sets the link function for the cumulative link model including: logit, probit, complementary log-log, cloglog, cauchit.
slope	selects the form of coefficients used in the model, with penalize denoting the penalized coefficients, unparallel, parallel and partial denoting the unpenalized non-parallel, parallel and semi-parallel coefficients respectively.
tuneMethod	sets the method of choosing an optimal shrinkage parameter, including: aic, cv, finite and user. i.e., the lambda value along parameter shrinkage path at which the fit's AIC or the k-fold cross-validated test error is minimal. The finite tuning is used to obtain the model along parameter shrinkage for which the log-Likelihood exist (is finite). The 'user' tuning supports a user-supplied lambda value.
reverse	false by default, when true the sign of the linear predictor is reversed.
lambdaGrid	optional user-supplied lambda grid for the aic, and cv tuning methods, when the discrete gridType is chosen. Negative range of values are not allowed. A short lambda grid could increase computation time assuming large number of predictors and cases in the model.
cvMetric	sets the performance metric for the cv tuning, with the brier score used by default.
gridType	chooses if a discrete or a continuous lambda grid should be used to select the optimal tuning parameter. The former is used by default and could be adjusted as desired in serp.control. The latter is on the range (0, maxPen). A user-supplied grid is also possible, which automatically overrides the internal grid.

`globalEff`	specifies variable(s) to be assigned global effects during penalization or when `slope` is set to `partial`. Variables are specified as a formula with an empty left hand side, for instance, globalEff = ~predictors.
`data`	optional dataframe explaining the variables used in the formula.
`subset`	specifies which subset of the rows of the data should be used for fit. All observations are used by default.
`weights`	optional case weights in fitting. Negative weights are not allowed. Defaults to 1.
`weight.type`	chooses between analytic and frequency weights with the former used by default. The latter should be used when weights are mere case counts used to compress the data set.
`na.action`	a function to filter missing data.
`lambda`	a user-supplied single numeric value for the tuning parameter when using the `user` tuning method. Negative values are not allowed.
`contrasts`	a list of contrasts to be used for some or all of the factors appearing as variables in the model formula.
`control`	A list of fit control parameters to replace default values returned by `serp.control`. Values not set assume default values.
`...`	additional arguments.

Details

The `serp` function fits the cumulative link model (CLM) with smooth-effect-on-response penalty (SERP). The cumulative model developed by McCullagh (1980) is probably most frequently used ordinal model. When motivated by an underlying latent variable, a simple form of the model is expressed as follows:

$$P(Y \leq r|x) = F(\delta_{0r} + x^T\delta)$$

where x is a vector of covariates, δ a vector of regression parameters and F a continuous distribution function. This model assumes that the effect of x does not depend on the category. However, with this assumption relaxed, one obtains the following general cumulative model:

$$P(Y \leq r|x) = F(\delta_{0r} + x^T\delta_r),$$

where r=1,...,k-1. This model, however, has the stochastic ordering property, which implies that $P(Y \leq r-1|x) < P(Y \leq r|x)$ holds for all x and all categories r. Such assumption is often problematic, resulting in unstable likelihoods with ill-conditioned parameter space during the iterative procedure.

SERP offers a means of arriving at stable estimates of the general model. It provides a form of regularization that is based on minimizing the penalized log-likelihood:

$$l_p(\delta) = l(\delta) - J_\lambda(\delta)$$

where $l(\delta)$, is the log-likelihood of the general cumulative model and $J_\lambda(\delta) = \lambda J(\delta)$ the penalty function weighted by the turning parameter λ. Assuming an ordered categorical outcome $Y \in \{1, \ldots, k\}$, and considering that the corresponding parameters $\delta_{1j}, \ldots \delta_{k-1,j}$ vary smoothly over the categories, the following penalty (Tutz and Gertheiss, 2016),

$$J_\lambda(\delta) = \sum_{j=1}^{p} \sum_{r=1}^{k-2} (\delta_{r+1,j} - \delta_{rj})^2$$

enables the smoothing of response categories such that all category-specific effects associated with the response turn towards a common global effect. SERP could also be applied to a semi-parallel model with only the category-specific part of the model penalized. See, Ugba et al. (2021) for a discussion and an application of SERP in an empirical study.

Value

An object of class `serp` with the components listed below, depending on the type of slope modeled. Other summary methods include: `summary`, `coef`, `predict`, `vcov`, `anova`, `errorMetrics`, etc.

aic the akaike information criterion, with effective degrees of freedom obtained from the trace of the generalized hat matrix depending on the tuning parameter.

bic the bayesian information criterion, with effective degrees of freedom obtained from the trace of the generalized hat matrix depending on the tuning parameter.

call the matched call.

coef a vector of coefficients of the fitted model.

converged a character vector of fit convergence status.

contrasts (where relevant) the contrasts used in the model.

control list of control parameters from `serp.control`.

cvMetric the performance metric used for cv tuning.

deviance the residual deviance.

edf the (effective) number of degrees of freedom used by the model

fitted.values the fitted probabilities.

globalEff variable(s) in model treated as global effect(s)

gradient a column vector of gradients for the coefficients at the model convergence.

Hessian the hessian matrix for the coefficients at the model convergence.

iter number of interactions before convergence or non-convergence.

lambda a user-supplied single numeric value for the `user` tuning tuning method.

lambdaGrid a numeric vector of lambda values used to determine the optimum tuning parameter.

logLik the realized log-likelihood at the model convergence.

link character vector indicating the link function of the fit.

message character vector stating the type of convergence obtained

misc a list to hold miscellaneous fit information.

model model.frame having variables from formula.

na.action (where relevant) information on the treatment of NAs.

nobs the number of observations.

nrFold the number of k-fold cross validation for the cv tuning method. Default to k = 5.

rdf the residual degrees of freedom

reverse a logical vector indicating the the direction of the cumulative probabilities. Default to P(Y<=r).

slope a character vector indicating the type of slope parameters fitted. Default to `penalize`.

Terms the terms structure describing the model.

testError numeric value of the cross-validated test error at which the optimal tuning parameter emerged.

tuneMethod a character vector specifying the method for choosing an optimal shrinkage parameter.

value numeric value of AIC or logLik obtained at the optimal tuning parameter when using `aic` or `finite` tuning methods respectively.

ylev the number of the response levels.

References

McCullagh, P. (1980). Regression Models for Ordinal Data. *Journal of the Royal Statistical Society. Series B (Methodological)*, 42, pp. 109-142.

Tutz, G. and Gertheiss, J. (2016). Regularized Regression for Categorical Data (With Discussion and Rejoinder). *Statistical Modelling*, 16, pp. 161-260. https://doi.org/10.1177/1471082X16642560

Ugba, E. R., Mörlein, D. and Gertheiss, J. (2021). Smoothing in Ordinal Regression: An Application to Sensory Data. *Stats*, 4, 616–633. https://doi.org/10.3390/stats4030037

See Also

`anova.serp`, `summary.serp`, `predict.serp`, `confint.serp`, `vcov.serp`, `errorMetrics`

Examples

```
require(serp)
## The unpenalized non-proportional odds model returns unbounded estimates, hence,
## not fully identifiable.
f1 <- serp(rating ~ temp + contact, slope = "unparallel",
           reverse = TRUE, link = "logit", data = wine)
coef(f1)

## The penalized non-proportional odds model with a user-supplied lambda gives
## a fully identified model with bounded estimates. A suitable tuning criterion
## could as well be used to select lambda (e.g., aic, cv)
f2 <- serp(rating ~ temp + contact, slope = "penalize",
           link = "logit", reverse = TRUE, tuneMethod = "user",
           lambda = 1e1, data = wine)
coef(f2)

## A penalized partial proportional odds model with one variable set to
```

```
## global effect is also possible.
f3 <- serp(rating ~ temp + contact, slope = "penalize",
           reverse = TRUE, link = "logit", tuneMethod = "user",
           lambda = 2e1, globalEff = ~ temp, data = wine)
coef(f3)


## The unpenalized proportional odds model with constrained estimates. Using a
## very strong lambda in f2 will result in estimates equal to estimates in f4.
f4 <-  serp(rating ~ temp + contact, slope = "parallel",
            reverse = FALSE, link = "logit", data = wine)
summary(f4)
```

`serp.control` *Control parameters for a fitted serp object*

Description

Default control parameters for 'serp' fit. User-supplied control parameters could be specified in the main function.

Usage

```
serp.control(
             maxits = 5e01,
             eps = 1e-07,
             maxpen = 1e07,
             trace = 0L,
             maxAdjIter = 5e0,
             max.half.iter = 1e01,
             relTol = 1e-03,
             nrFold = 5e0,
             cv.seed = 1e01,
             grid.length = 5e01,
             minP = .Machine$double.eps,
             ...)
```

Arguments

`maxits`	the maximum number of Newton's iterations. Default to 100.
`eps`	threshold value during optimization at which the iteration routine terminates. In other words, when the reported change in the log-likelihood goes below this threshold, convergence is achieved.
`maxpen`	the upper end point of the interval from zero to be searched for a tuning parameter.
`trace`	prints the Newton's fitting process at each iteration step.If 0 (default) no information is printed, if 1, 2 or 3 different shades of information are printed.

maxAdjIter	the maximum allowable number of Newton step adjustment to forestall an early optimization failure. Defaults to 5.
max.half.iter	the maximum number of iteration step-halfings. Defaults to 10.
relTol	relative convergence tolerance, defaults to 1e-03. checks relative changes in the parameter estimates between Newton iterations.
nrFold	the number of k-fold cross validation for the CV tuning method. Default to k = 5.
cv.seed	single numeric value to change the random seed in CV tuning.
grid.length	the length of the discrete lambda grid for the penalty method.
minP	A near zero minimum value the fitted probabilities are allowed to get during iteration to prevent numerical instability .
...	additional arguments.

Value

a list of control parameters.

See Also

serp

Examples

```
library(serp)
serp(rating ~ contact, slope = "parallel", link = "logit",
     control = list(maxits = 2e01, eps=1e-05, trace = 2),
     data = wine)
```

---

summary.serp *Summary method for a fitted serp object.*

Description

This function summarizes the result of a fitted serp object in a dataframe.

Usage

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

Arguments

object	An object of class serp.
...	Not used. Additional summary arguments.

Value

an object of class `summary.serp`. A list (depending on the type of `slope` used) of all model components defined in the `serp`, function with additional components listed below.

coefficients the matrix of coefficients, standard errors, z-values and p-values.

null.deviance the deviance for the intercept only model.

null.logLik the log-likelihood for the intercept only model.

penalty list of penalization information obtained with `slope` set to "penalize".

expcoefs the exponentiated coefficients.

See Also

`anova.serp`, `predict.serp`, `confint.serp`, `vcov.serp`, `errorMetrics`

Examples

```
library(serp)
m <- serp(rating ~ temp + contact, slope = "penalize",
          reverse = TRUE, link = "logit", tuneMethod = "user",
          lambda = 0, data = wine)
summary(m)
```

`vcov.serp` *Variance covariance matrix for a fitted serp object*

Description

Provides the Variance covariance matrix of an object of class `serp`.

Usage

```
## S3 method for class 'serp'
vcov(object, ...)
```

Arguments

`object`	An object of class `serp`.
`...`	additional arguments.

Value

A variance covariance matrix of a fitted model.

See Also

`serp`

`serp`, `anova.serp`, `confint.serp`, `errorMetrics`

Examples

```
library(serp)
m <- serp(rating ~ temp + contact, slope = "parallel", link = "logit",
          data = wine)
vcov(m)
```

`wine` *Bitterness of wine dataset*

Description

The `wine` dataset adopted from Randall(1989), represents the outcome of a factorial experiment on factors determining the bitterness of wine. Two treatment factors (temperature and contact) with two levels each are provided, with the rating of wine taken on a continuous scale in the interval from 0 (none) to 100 (intense). These were subsequently grouped into five ordered categories ranging from 1 = 'least bitter' to 5 = 'most bitter'. Altogether, nine different judges assessed wine from two bottles and out of the four treatment conditions, making a total of 72 observations.

Usage

```
wine
```

Format

A data frame with 72 rows and 6 variables:

`response` scorings of wine bitterness on a 0—100 continuous scale.

`rating` ordered factor with 5 levels; a grouped version of `response`.

`contact` factor with two levels (`"no"` and `"yes"`).

`temp` temperature: factor with two levels.

`judge` factor with nine levels.

`bottle` factor with eight levels.

Source

Taken from Randall (1989).

References

Randall, J (1989). The analysis of sensory data by generalized linear model. *Biometrical journal 7*, pp. 781–793.

Examples

```
## Not run:
str(wine)
head(wine)

## End(Not run)
```

Index