

Package ‘mpath’

June 1, 2020

Title Regularized Linear Models

Version 0.3-26

Date 2020-06-01

Author Zhu Wang, with contributions from Achim Zeileis, Simon Jackman, Brian Ripley, Trevor Hastie, Rob Tibshirani, Balasubramanian Narasimhan, Gil Chu and Patrick Breheny

Maintainer Zhu Wang <wangz1@uthscsa.edu>

Description Algorithms optimize penalized models. Currently the models include penalized Poisson, negative binomial, zero-inflated Poisson, zero-inflated negative binomial regression models and robust models. The penalties include least absolute shrinkage and selection operator (LASSO), smoothly clipped absolute deviation (SCAD), minimax concave penalty (MCP), and each possibly combining with L_2 penalty. See Wang et al. (2014) <doi:10.1002/sim.6314>, Wang et al. (2015) <doi:10.1002/bimj.201400143>, Wang et al. (2016) <doi:10.1177/0962280214530608>, Wang (2019) <arXiv:1912.11119>.

Imports MASS, pscl, numDeriv, foreach, doParallel, bst

Depends methods

Suggests zic, R.rsp, knitr, gdata

VignetteBuilder R.rsp, knitr

License GPL-2

URL <https://github.com/zhuwang46/mpath>

BugReports <https://github.com/zhuwang46/mpath>

NeedsCompilation yes

Repository CRAN

Date/Publication 2020-06-01 20:30:06 UTC

R topics documented:

| | |
|-----------------------|---|
| be.zeroinfl | 2 |
| breadReg | 3 |
| conv2glmreg | 4 |
| conv2zipath | 5 |

| | |
|----------------------------|----|
| cv.glmreg | 5 |
| cv.glmregNB | 7 |
| cv.glmreg_fit | 9 |
| cv.nclreg | 11 |
| cv.nclreg_fit | 13 |
| cv.zipath | 14 |
| cv.zipath_fit | 17 |
| estfunReg | 19 |
| glmreg | 20 |
| glmregNB | 22 |
| glmreg_fit | 25 |
| hessianReg | 28 |
| meatReg | 29 |
| methods | 30 |
| ncl | 31 |
| nclreg | 33 |
| nclreg_fit | 35 |
| ncl_fit | 38 |
| plot.glmreg | 39 |
| predict.glmreg | 40 |
| predict.zipath | 41 |
| pval.zipath | 43 |
| rzi | 45 |
| sandwichReg | 46 |
| se | 47 |
| stan | 48 |
| summary.glmregNB | 49 |
| tuning.zipath | 50 |
| zipath | 51 |
| zipath_fit | 54 |

Index **59**

| | |
|-------------|--|
| be.zeroinfl | <i>conduct backward stepwise variable elimination for zero inflated count regression</i> |
|-------------|--|

Description

conduct backward stepwise variable elimination for zero inflated count regression from zeroinfl function

Usage

```
be.zeroinfl(object, data, dist=c("poisson", "negbin", "geometric"), alpha=0.05, trace=FALSE)
```

Arguments

| | |
|--------|--|
| object | an object from function zeroinfl |
| data | argument controlling formula processing via <code>model.frame</code> . |
| dist | one of the distributions in zeroinfl function |
| alpha | significance level of variable elimination |
| trace | logical value, if TRUE, print detailed calculation results |

Details

conduct backward stepwise variable elimination for zero inflated count regression from zeroinfl function

Value

an object of zeroinfl with all variables having p-values less than the significance level alpha

Author(s)

Zhu Wang <wangz1@uthscsa.edu>

References

Zhu Wang, Shuangge Ma, Ching-Yun Wang, Michael Zappitelli, Prasad Devarajan and Chirag R. Parikh (2014) *EM for Regularized Zero Inflated Regression Models with Applications to Postoperative Morbidity after Cardiac Surgery in Children*, *Statistics in Medicine*. 33(29):5192-208.

Zhu Wang, Shuangge Ma and Ching-Yun Wang (2015) *Variable selection for zero-inflated and overdispersed data with application to health care demand in Germany*, *Biometrical Journal*. 57(5):867-84.

| | |
|----------|---|
| breadReg | <i>Bread for Sandwiches in Regularized Estimators</i> |
|----------|---|

Description

Generic function for extracting an estimator for the bread of sandwiches.

Usage

```
breadReg(x, which, ...)
```

Arguments

| | |
|-------|------------------------------|
| x | a fitted model object. |
| which | which penalty parameter(s)? |
| ... | arguments passed to methods. |

Value

A matrix containing an estimator for the penalized second derivative of log-likelihood function. Typically, this should be an $k \times k$ matrix corresponding to k parameters. The rows and columns should be named as in `coef` or `terms`, respectively.

Author(s)

Zhu Wang <wangz1@uthscsa.edu>

References

Zhu Wang, Shuangge Ma and Ching-Yun Wang (2015) *Variable selection for zero-inflated and overdispersed data with application to health care demand in Germany*, *Biometrical Journal*. 57(5):867-84.

See Also

[meatReg](#), [sandwichReg](#)

Examples

```
data("bioChemists", package = "pscl")
fm_zinb <- zipath(art ~ . | ., data = bioChemists, family = "negbin", nlambda=10, maxit.em=1)
breadReg(fm_zinb, which=which.min(fm_zinb$bic))
```

conv2glmreg

convert glm object to class glmreg

Description

convert glm object to class glmreg, which then can be used for other purposes

Usage

```
conv2glmreg(object, family=c("poisson", "negbin"))
```

Arguments

| | |
|--------|------------------------------|
| object | an object of class glm |
| family | one of families in glm class |

Value

an object of class glmreg

Author(s)

Zhu Wang <wangz1@uthscsa.edu>

| | |
|-------------|--|
| conv2zipath | <i>convert zeroinfl object to class zipath</i> |
|-------------|--|

Description

convert zeroinfl object to class zipath, which then can be used to predict new data

Usage

```
conv2zipath(object, family=c("poisson", "negbin", "geometric"))
```

Arguments

| | |
|--------|-----------------------------------|
| object | an object of class zeroinfl |
| family | one of families in zeroinfl class |

Value

an object of class zipath

Author(s)

Zhu Wang <wangz1@uthscsa.edu>

| | |
|-----------|------------------------------------|
| cv.glmreg | <i>Cross-validation for glmreg</i> |
|-----------|------------------------------------|

Description

Does k-fold cross-validation for glmreg, produces a plot, and returns cross-validated log-likelihood values for lambda

Usage

```
## S3 method for class 'formula'
cv.glmreg(formula, data, weights, offset=NULL, contrasts=NULL, ...)
## S3 method for class 'matrix'
cv.glmreg(x, y, weights, offset=NULL, ...)
## Default S3 method:
cv.glmreg(x, ...)
## S3 method for class 'cv.glmreg'
plot(x, se=TRUE, ylab=NULL, main=NULL, width=0.02, col="darkgrey", ...)
## S3 method for class 'cv.glmreg'
predict(object, newx, ...)
## S3 method for class 'cv.glmreg'
coef(object, which=object$lambda.which, ...)
```

Arguments

| | |
|-----------|--|
| formula | symbolic description of the model, see details. |
| data | argument controlling formula processing via <code>model.frame</code> . |
| x | x matrix as in <code>glmreg</code> . It could be object of <code>cv.glmreg</code> . |
| y | response y as in <code>glmreg</code> . |
| weights | Observation weights; defaults to 1 per observation |
| offset | this can be used to specify an a priori known component to be included in the linear predictor during fitting. This should be <code>NULL</code> or a numeric vector of length equal to the number of cases. Currently only one offset term can be included in the formula. |
| contrasts | the contrasts corresponding to levels from the respective models |
| object | object of <code>cv.glmreg</code> |
| newx | Matrix of values at which predictions are to be made. Not used for <code>type="coefficients"</code> |
| which | Indices of the penalty parameter lambda at which estimates are extracted. By default, the one which generates the optimal cross-validation value. |
| se | logical value, if <code>TRUE</code> , standard error curve is also plotted |
| ylab | ylab on y-axis |
| main | title of plot |
| width | width of lines |
| col | color of standard error curve |
| ... | Other arguments that can be passed to <code>glmreg</code> . |

Details

The function runs `glmreg` `nfolds+1` times; the first to compute the lambda sequence, and then to compute the fit with each of the folds omitted. The error or the log-likelihood value is accumulated, and the average value and standard deviation over the folds is computed. Note that `cv.glmreg` can be used to search for values for alpha: it is required to call `cv.glmreg` with a fixed vector `foldid` for different values of alpha.

Value

an object of class `"cv.glmreg"` is returned, which is a list with the ingredients of the cross-validation fit.

| | |
|----------|---|
| fit | a fitted <code>glmreg</code> object for the full data. |
| residmat | matrix of log-likelihood values with row values for lambda and column values for kth cross-validation |
| bic | matrix of BIC values with row values for lambda and column values for kth cross-validation |
| cv | The mean cross-validated log-likelihood values - a vector of length <code>length(lambda)</code> . |
| cv.error | estimate of standard error of cv. |

| | |
|--------------|--|
| foldid | an optional vector of values between 1 and nfold identifying what fold each observation is in. |
| lambda | a vector of lambda values |
| lambda.which | index of lambda that gives maximum cv value. |
| lambda.optim | value of lambda that gives maximum cv value. |

Author(s)

Zhu Wang <wangz1@uthscsa.edu>

References

Zhu Wang, Shuangge Ma, Michael Zappitelli, Chirag Parikh, Ching-Yun Wang and Prasad Devarajan (2014) *Penalized Count Data Regression with Application to Hospital Stay after Pediatric Cardiac Surgery*, *Statistical Methods in Medical Research*. 2014 Apr 17. [Epub ahead of print]

See Also

[glmreg](#) and [plot](#), [predict](#), and [coef](#) methods for "cv.glmreg" object.

Examples

```
data("bioChemists", package = "pscl")
fm_pois <- cv.glmreg(art ~ ., data = bioChemists, family = "poisson")
title("Poisson Family", line=2.5)
predict(fm_pois, newx=bioChemists[,-1])[1:4]
coef(fm_pois)
```

cv.glmregNB

Cross-validation for glmregNB

Description

Does k-fold cross-validation for glmregNB, produces a plot, and returns cross-validated log-likelihood values for lambda

Usage

```
cv.glmregNB(formula, data, weights, offset=NULL, lambda=NULL, nfolds=10,
foldid, plot.it=TRUE, se=TRUE, n.cores=2, trace=FALSE, parallel=FALSE, ...)
```

Arguments

| | |
|----------|--|
| formula | symbolic description of the model |
| data | arguments controlling formula processing via <code>model.frame</code> . |
| weights | Observation weights; defaults to 1 per observation |
| offset | this can be used to specify an a priori known component to be included in the linear predictor during fitting. This should be <code>NULL</code> or a numeric vector of length equal to the number of cases. Currently only one offset term can be included in the formula. |
| lambda | Optional user-supplied lambda sequence; default is <code>NULL</code> , and <code>glmregNB</code> chooses its own sequence |
| nfolds | number of folds - default is 10. Although <code>nfolds</code> can be as large as the sample size (leave-one-out CV), it is not recommended for large datasets. Smallest value allowable is <code>nfolds=3</code> |
| foldid | an optional vector of values between 1 and <code>nfold</code> identifying what fold each observation is in. If supplied, <code>nfold</code> can be missing. |
| plot.it | a logical value, to plot the estimated log-likelihood values if <code>TRUE</code> . |
| se | a logical value, to plot with standard errors. |
| n.cores | The number of CPU cores to use. The cross-validation loop will attempt to send different CV folds off to different cores. |
| trace | a logical value, print progress of cross-validation or not |
| parallel | a logical value, parallel computing or not |
| ... | Other arguments that can be passed to <code>glmregNB</code> . |

Details

The function runs `glmregNB` `nfolds+1` times; the first to get the lambda sequence, and then the remainder to compute the fit with each of the folds omitted. The error is accumulated, and the average error and standard deviation over the folds is computed. Note that `cv.glmregNB` does NOT search for values for alpha. A specific value should be supplied, else `alpha=1` is assumed by default. If users would like to cross-validate alpha as well, they should call `cv.glmregNB` with a pre-computed vector `foldid`, and then use this same fold vector in separate calls to `cv.glmregNB` with different values of alpha.

Value

| | |
|----------|--|
| | an object of class " <code>cv.glmregNB</code> " is returned, which is a list with the ingredients of the cross-validation fit. |
| fit | a fitted <code>glmregNB</code> object for the full data. |
| residmat | matrix of log-likelihood values with row values for lambda and column values for kth cross-validation |
| cv | The mean cross-validated log-likelihood values - a vector of length <code>length(lambda)</code> . |
| cv.error | The standard error of cross-validated log-likelihood values - a vector of length <code>length(lambda)</code> . |

| | |
|--------------|---|
| lambda | a vector of lambda values |
| foldid | indicators of data used in each cross-validation, for reproductive purposes |
| lambda.which | index of lambda that gives maximum cv value. |
| lambda.optim | value of lambda that gives maximum cv value. |

Author(s)

Zhu Wang <wangz1@uthscsa.edu>

References

Zhu Wang, Shuangge Ma, Michael Zappitelli, Chirag Parikh, Ching-Yun Wang and Prasad Devarajan (2014) *Penalized Count Data Regression with Application to Hospital Stay after Pediatric Cardiac Surgery*, *Statistical Methods in Medical Research*. 2014 Apr 17. [Epub ahead of print]

See Also

[glmregNB](#) and [plot](#), [predict](#), and [coef](#) methods for "cv.glmregNB" object.

Examples

```
## Not run:
data("bioChemists", package = "pscl")
fm_nb <- cv.glmregNB(art ~ ., data = bioChemists)
plot(fm_nb)

## End(Not run)
```

cv.glmreg_fit

Internal function of cross-validation for glmreg

Description

Internal function to conduct k-fold cross-validation for glmreg, produces a plot, and returns cross-validated log-likelihood values for lambda

Usage

```
cv.glmreg_fit(x, y, weights, offset, lambda=NULL, balance=TRUE,
family=c("gaussian", "binomial", "poisson", "negbin"), nfolds=10, foldid,
plot.it=TRUE, se=TRUE, n.cores=2, trace=FALSE, parallel=FALSE, ...)
```

Arguments

| | |
|----------|---|
| x | x matrix as in glmreg. |
| y | response y as in glmreg. |
| weights | Observation weights; defaults to 1 per observation |
| offset | this can be used to specify an a priori known component to be included in the linear predictor during fitting. This should be NULL or a numeric vector of length equal to the number of cases. Currently only one offset term can be included in the formula. |
| lambda | Optional user-supplied lambda sequence; default is NULL, and glmreg chooses its own sequence |
| balance | for family="binomial" only |
| family | response variable distribution |
| nfolds | number of folds ≥ 3 , default is 10 |
| foldid | an optional vector of values between 1 and nfold identifying what fold each observation is in. If supplied, nfold can be missing and will be ignored. |
| plot.it | a logical value, to plot the estimated log-likelihood values if TRUE. |
| se | a logical value, to plot with standard errors. |
| n.cores | The number of CPU cores to use. The cross-validation loop will attempt to send different CV folds off to different cores. |
| trace | a logical value, print progress of cross validation or not |
| parallel | a logical value, parallel computing or not |
| ... | Other arguments that can be passed to glmreg. |

Details

The function runs glmreg nfolds+1 times; the first to compute the lambda sequence, and then to compute the fit with each of the folds omitted. The error or the log-likelihood value is accumulated, and the average value and standard deviation over the folds is computed. Note that cv.glmreg can be used to search for values for alpha: it is required to call cv.glmreg with a fixed vector foldid for different values of alpha.

Value

an object of class "cv.glmreg" is returned, which is a list with the ingredients of the cross-validation fit.

| | |
|----------|---|
| fit | a fitted glmreg object for the full data. |
| residmat | matrix of log-likelihood values with row values for lambda and column values for kth cross-validation |
| cv | The mean cross-validated log-likelihood values - a vector of length length(lambda). |
| cv.error | estimate of standard error of cv. |
| foldid | an optional vector of values between 1 and nfold identifying what fold each observation is in. |

lambda a vector of lambda values
 lambda.which index of lambda that gives maximum cv value.
 lambda.optim value of lambda that gives maximum cv value.

Author(s)

Zhu Wang <wangz1@uthscsa.edu>

References

Zhu Wang, Shuangge Ma, Michael Zappitelli, Chirag Parikh, Ching-Yun Wang and Prasad Devarajan (2014) *Penalized Count Data Regression with Application to Hospital Stay after Pediatric Cardiac Surgery*, *Statistical Methods in Medical Research*. 2014 Apr 17. [Epub ahead of print]

See Also

[glmreg](#) and [plot](#), [predict](#), and [coef](#) methods for "cv.glmreg" object.

cv.nclreg *Cross-validation for nclreg*

Description

Does k-fold cross-validation for nclreg, produces a plot, and returns cross-validated log-likelihood values for lambda

Usage

```
## S3 method for class 'formula'
cv.nclreg(formula, data, weights, offset=NULL, ...)
## S3 method for class 'matrix'
cv.nclreg(x, y, weights, offset=NULL, ...)
## Default S3 method:
cv.nclreg(x, ...)
## S3 method for class 'cv.nclreg'
plot(x,se=TRUE,ylab=NULL, main=NULL, width=0.02, col="darkgrey", ...)
## S3 method for class 'cv.nclreg'
coef(object,which=object$lambda.which, ...)
```

Arguments

formula symbolic description of the model, see details.
 data argument controlling formula processing via [model.frame](#).
 x x matrix as in nclreg. It could be object of cv.nclreg.
 y response y as in nclreg.
 weights Observation weights; defaults to 1 per observation

| | |
|--------|--|
| offset | Not implemented yet |
| object | object of <code>cv.nclreg</code> |
| which | Indices of the penalty parameter <code>lambda</code> at which estimates are extracted. By default, the one which generates the optimal cross-validation value. |
| se | logical value, if TRUE, standard error curve is also plotted |
| ylab | ylab on y-axis |
| main | title of plot |
| width | width of lines |
| col | color of standard error curve |
| ... | Other arguments that can be passed to <code>nclreg</code> . |

Details

The function runs `nclreg` `nfolds+1` times; the first to compute the `lambda` sequence, and then to compute the fit with each of the folds omitted. The error or the loss value is accumulated, and the average value and standard deviation over the folds is computed. Note that `cv.nclreg` can be used to search for values for `alpha`: it is required to call `cv.nclreg` with a fixed vector `foldid` for different values of `alpha`.

Value

an object of class "`cv.nclreg`" is returned, which is a list with the ingredients of the cross-validation fit.

| | |
|---------------------------|---|
| <code>fit</code> | a fitted <code>nclreg</code> object for the full data. |
| <code>residmat</code> | matrix of log-likelihood values with row values for <code>lambda</code> and column values for <code>kth</code> cross-validation |
| <code>bic</code> | matrix of BIC values with row values for <code>lambda</code> and column values for <code>kth</code> cross-validation |
| <code>cv</code> | The mean cross-validated log-likelihood values - a vector of length <code>length(lambda)</code> . |
| <code>cv.error</code> | estimate of standard error of <code>cv</code> . |
| <code>foldid</code> | an optional vector of values between 1 and <code>nfold</code> identifying what fold each observation is in. |
| <code>lambda</code> | a vector of <code>lambda</code> values |
| <code>lambda.which</code> | index of <code>lambda</code> that gives minimum <code>cv</code> value. |
| <code>lambda.optim</code> | value of <code>lambda</code> that gives minimum <code>cv</code> value. |

Author(s)

Zhu Wang <wangz1@uthscsa.edu>

References

Zhu Wang (2019) *MM for Penalized Estimation*, <https://arxiv.org/abs/1912.11119>

See Also

[nclreg](#) and [plot](#), [predict](#), and [coef](#) methods for "cv.nclreg" object.

 cv.nclreg_fit

Internal function of cross-validation for nclreg

Description

Internal function to conduct k-fold cross-validation for nclreg, produces a plot, and returns cross-validated log-likelihood values for lambda

Usage

```
cv.nclreg_fit(x, y, weights, lambda=NULL, balance=TRUE,
             rfamily=c("clossR", "closs", "gloss", "qloss"), s=1.5, nfolds=10, foldid,
             type = c("loss", "error"), plot.it=TRUE, se=TRUE, n.cores=2, trace=FALSE,
             parallel=FALSE, ...)
```

Arguments

| | |
|----------|---|
| x | x matrix as in nclreg. |
| y | response y as in nclreg. |
| weights | Observation weights; defaults to 1 per observation |
| lambda | Optional user-supplied lambda sequence; default is NULL, and nclreg chooses its own sequence |
| balance | for rfamily="closs", "gloss", "qloss" only |
| rfamily | response variable distribution and nonconvex loss function |
| s | nonconvex loss tuning parameter for robust regression and classification. |
| nfolds | number of folds ≥ 3 , default is 10 |
| foldid | an optional vector of values between 1 and nfold identifying what fold each observation is in. If supplied, nfold can be missing and will be ignored. |
| type | cross-validation criteria. For type="loss", loss function values and type="error" is misclassification error. |
| plot.it | a logical value, to plot the estimated log-likelihood values if TRUE. |
| se | a logical value, to plot with standard errors. |
| n.cores | The number of CPU cores to use. The cross-validation loop will attempt to send different CV folds off to different cores. |
| trace | a logical value, print progress of cross validation or not |
| parallel | a logical value, parallel computing or not |
| ... | Other arguments that can be passed to nclreg. |

Details

The function runs `nclreg` `nfolds+1` times; the first to compute the `lambda` sequence, and then to compute the fit with each of the folds omitted. The error or the log-likelihood value is accumulated, and the average value and standard deviation over the folds is computed. Note that `cv.nclreg` can be used to search for values for `alpha`: it is required to call `cv.nclreg` with a fixed vector `foldid` for different values of `alpha`.

Value

an object of class `"cv.nclreg"` is returned, which is a list with the ingredients of the cross-validation fit.

| | |
|---------------------------|---|
| <code>fit</code> | a fitted <code>nclreg</code> object for the full data. |
| <code>residmat</code> | matrix of log-likelihood values with row values for <code>lambda</code> and column values for <code>kth</code> cross-validation |
| <code>cv</code> | The mean cross-validated log-likelihood values - a vector of length <code>length(lambda)</code> . |
| <code>cv.error</code> | estimate of standard error of <code>cv</code> . |
| <code>foldid</code> | an optional vector of values between 1 and <code>nfold</code> identifying what fold each observation is in. |
| <code>lambda</code> | a vector of <code>lambda</code> values |
| <code>lambda.which</code> | index of <code>lambda</code> that gives minimum <code>cv</code> value. |
| <code>lambda.optim</code> | value of <code>lambda</code> that gives minimum <code>cv</code> value. |

Author(s)

Zhu Wang <wangz1@uthscsa.edu>

References

Zhu Wang (2019) *MM for Penalized Estimation*, <https://arxiv.org/abs/1912.11119>

See Also

`nclreg` and `plot`, `predict`, and `coef` methods for `"cv.nclreg"` object.

cv.zipath

Cross-validation for zipath

Description

Does `k`-fold cross-validation for `zipath`, produces a plot, and returns cross-validated log-likelihood values for `lambda`

Usage

```
## S3 method for class 'formula'
cv.zipath(formula, data, weights, offset=NULL, contrasts=NULL, ...)
## S3 method for class 'matrix'
cv.zipath(X, Z, Y, weights, offsetx=NULL, offsetz=NULL, ...)
## Default S3 method:
cv.zipath(X, ...)
## S3 method for class 'cv.zipath'
predict(object, newdata, ...)
## S3 method for class 'cv.zipath'
coef(object, which=object$lambda.which, model = c("full", "count", "zero"), ...)
```

Arguments

| | |
|------------------|---|
| formula | symbolic description of the model with an optional numeric vector <code>offset</code> with an a priori known component to be included in the linear predictor of the count model or zero model. Offset must be a variable in <code>data</code> if used, while this is optional in <code>zipath</code> . See an example below. |
| data | arguments controlling formula processing via model.frame . |
| weights | Observation weights; defaults to 1 per observation |
| offset | optional numeric vector with an a priori known component to be included in the linear predictor of the count model or zero model. See below for an example. |
| X | predictor matrix of the count model |
| Z | predictor matrix of the zero model |
| Y | response variable |
| offsetx, offsetz | optional numeric vector with an a priori known component to be included in the linear predictor of the count model (<code>offsetx</code>) or zero model (<code>offsetz</code>). |
| contrasts | a list with elements "count" and "zero" containing the contrasts corresponding to levels from the respective models |
| object | object of class <code>cv.zipath</code> . |
| newdata | optionally, a data frame in which to look for variables with which to predict. If omitted, the original observations are used. |
| which | Indices of the pair of penalty parameters <code>lambda.count</code> and <code>lambda.zero</code> at which estimates are extracted. By default, the one which generates the optimal cross-validation value. |
| model | character specifying for which component of the model the estimated coefficients should be extracted. |
| ... | Other arguments that can be passed to <code>zipath</code> . |

Details

The function runs `zipath` `nfolds+1` times; the first to compute the `(lambda.count, lambda.zero)` sequence, and then to compute the fit with each of the folds omitted. The log-likelihood value is

accumulated, and the average value and standard deviation over the folds is computed. Note that `cv.zipath` can be used to search for values for `count.alpha` or `zero.alpha`: it is required to call `cv.zipath` with a fixed vector `foldid` for different values of `count.alpha` or `zero.alpha`.

The method for `coef` by default return a single vector of coefficients, i.e., all coefficients are concatenated. By setting the `model` argument, the estimates for the corresponding model components can be extracted.

Value

an object of class "cv.zipath" is returned, which is a list with the components of the cross-validation fit.

| | |
|---------------------------|--|
| <code>fit</code> | a fitted zipath object for the full data. |
| <code>residmat</code> | matrix for cross-validated log-likelihood at each $(\text{count.lambda}, \text{zero.lambda})$ sequence |
| <code>bic</code> | matrix of BIC values with row values for <code>lambda</code> and column values for <code>kth</code> cross-validation |
| <code>cv</code> | The mean cross-validated log-likelihood - a vector of length <code>length(count.lambda)</code> . |
| <code>cv.error</code> | estimate of standard error of <code>cv</code> . |
| <code>foldid</code> | an optional vector of values between 1 and <code>nfold</code> identifying what fold each observation is in. |
| <code>lambda.which</code> | index of $(\text{count.lambda}, \text{zero.lambda})$ that gives maximum <code>cv</code> . |
| <code>lambda.optim</code> | value of $(\text{count.lambda}, \text{zero.lambda})$ that gives maximum <code>cv</code> . |

Author(s)

Zhu Wang <wangz1@uthscsa.edu>

References

Zhu Wang, Shuangge Ma, Michael Zappitelli, Chirag Parikh, Ching-Yun Wang and Prasad Devarajan (2014) *Penalized Count Data Regression with Application to Hospital Stay after Pediatric Cardiac Surgery*, *Statistical Methods in Medical Research*. 2014 Apr 17. [Epub ahead of print]

Zhu Wang, Shuangge Ma, Ching-Yun Wang, Michael Zappitelli, Prasad Devarajan and Chirag R. Parikh (2014) *EM for Regularized Zero Inflated Regression Models with Applications to Postoperative Morbidity after Cardiac Surgery in Children*, *Statistics in Medicine*. 33(29):5192-208.

Zhu Wang, Shuangge Ma and Ching-Yun Wang (2015) *Variable selection for zero-inflated and overdispersed data with application to health care demand in Germany*, *Biometrical Journal*. 57(5):867-84.

See Also

`zipath` and `plot`, `predict`, and `coef` methods for "cv.zipath" object.

Examples

```

## Not run:
data("bioChemists", package = "pscl")
fm_zip <- zipath(art ~ . | ., data = bioChemists, family = "poisson", nlambda=10)
fm_cvzip <- cv.zipath(art ~ . | ., data = bioChemists, family = "poisson", nlambda=10)
### prediction from the best model
pred <- predict(fm_zip, newdata=bioChemists, which=fm_cvzip$lambda.which)
coef(fm_zip, which=fm_cvzip$lambda.which)
fm_znb <- zipath(art ~ . | ., data = bioChemists, family = "negbin", nlambda=10)
fm_cvznb <- cv.zipath(art ~ . | ., data = bioChemists, family = "negbin", nlambda=10)
pred <- predict(fm_znb, which=fm_cvznb$lambda.which)
coef(fm_znb, which=fm_cvznb$lambda.which)
fm_zinb2 <- zipath(art ~ . +offset(log(phd))| ., data = bioChemists,
  family = "poisson", nlambda=10)
fm_cvzinb2 <- cv.zipath(art ~ . +offset(log(phd))| ., data = bioChemists,
  family = "poisson", nlambda=10)
pred <- predict(fm_zinb2, which=fm_cvzinb2$lambda.which)
coef(fm_zinb2, which=fm_cvzinb2$lambda.which)

## End(Not run)

```

cv.zipath_fit

Cross-validation for zipath

Description

Internal function k-fold cross-validation for zipath, produces a plot, and returns cross-validated log-likelihood values for lambda

Usage

```

cv.zipath_fit(X, Z, Y, weights, offsetx, offsetz, nlambda=100, lambda.count=NULL,
  lambda.zero=NULL, nfolds=10, foldid, plot.it=TRUE, se=TRUE, n.cores=2,
  trace=FALSE, parallel=FALSE, ...)

```

Arguments

| | |
|---------|---|
| X | predictor matrix of the count model |
| Z | predictor matrix of the zero model |
| Y | response variable |
| weights | optional numeric vector of weights. |
| offsetx | optional numeric vector with an a priori known component to be included in the linear predictor of the count model. |
| offsetz | optional numeric vector with an a priori known component to be included in the linear predictor of the zero model. |
| nlambda | number of lambda value, default value is 10. |

| | |
|--------------|---|
| lambda.count | Optional user-supplied lambda.count sequence; default is NULL |
| lambda.zero | Optional user-supplied lambda.zero sequence; default is NULL |
| nfolds | number of folds ≥ 3 , default is 10 |
| foldid | an optional vector of values between 1 and nfold identifying what fold each observation is in. If supplied, nfold can be missing and will be ignored. |
| plot.it | a logical value, to plot the estimated log-likelihood values if TRUE. |
| se | a logical value, to plot with standard errors. |
| n.cores | The number of CPU cores to use. The cross-validation loop will attempt to send different CV folds off to different cores. |
| trace | a logical value, print progress of cross-validation or not |
| parallel | a logical value, parallel computing or not |
| ... | Other arguments that can be passed to zipath. |

Details

The function runs `zipath` `nfolds+1` times; the first to compute the `(lambda.count, lambda.zero)` sequence, and then to compute the fit with each of the folds omitted. The log-likelihood value is accumulated, and the average value and standard deviation over the folds is computed. Note that `cv.zipath` can be used to search for values for `count.alpha` or `zero.alpha`: it is required to call `cv.zipath` with a fixed vector `foldid` for different values of `count.alpha` or `zero.alpha`.

The method for `coef` by default return a single vector of coefficients, i.e., all coefficients are concatenated. By setting the `model` argument, the estimates for the corresponding model components can be extracted.

Value

an object of class "`cv.zipath`" is returned, which is a list with the components of the cross-validation fit.

| | |
|--------------|---|
| fit | a fitted zipath object for the full data. |
| residmat | matrix for cross-validated log-likelihood at each <code>(count.lambda, zero.lambda)</code> sequence |
| bic | matrix of BIC values with row values for lambda and column values for kth cross-validation |
| cv | The mean cross-validated log-likelihood - a vector of length <code>length(count.lambda)</code> . |
| cv.error | estimate of standard error of cv. |
| foldid | an optional vector of values between 1 and nfold identifying what fold each observation is in. |
| lambda.which | index of <code>(count.lambda, zero.lambda)</code> that gives maximum cv. |
| lambda.optim | value of <code>(count.lambda, zero.lambda)</code> that gives maximum cv. |

Author(s)

Zhu Wang <wangz1@uthscsa.edu>

References

Zhu Wang, Shuangge Ma, Michael Zappitelli, Chirag Parikh, Ching-Yun Wang and Prasad Devarajan (2014) *Penalized Count Data Regression with Application to Hospital Stay after Pediatric Cardiac Surgery*, *Statistical Methods in Medical Research*. 2014 Apr 17. [Epub ahead of print]

Zhu Wang, Shuangge Ma, Ching-Yun Wang, Michael Zappitelli, Prasad Devarajan and Chirag R. Parikh (2014) *EM for Regularized Zero Inflated Regression Models with Applications to Postoperative Morbidity after Cardiac Surgery in Children*, *Statistics in Medicine*. 33(29):5192-208.

Zhu Wang, Shuangge Ma and Ching-Yun Wang (2015) *Variable selection for zero-inflated and overdispersed data with application to health care demand in Germany*, *Biometrical Journal*. 57(5):867-84.

See Also

[zipath](#) and [plot](#), [predict](#), and [coef](#) methods for "cv.zipath" object.

estfunReg

Extract Empirical First Derivative of Log-likelihood Function

Description

Generic function for extracting the empirical first derivative of log-likelihood function of a fitted regularized model.

Usage

```
estfunReg(x, ...)
```

Arguments

`x` a fitted model object.
`...` arguments passed to methods.

Value

A matrix containing the empirical first derivative of log-likelihood functions. Typically, this should be an $n \times k$ matrix corresponding to n observations and k parameters. The columns should be named as in [coef](#) or [terms](#), respectively.

Author(s)

Zhu Wang <wangz1@uthscsa.edu>

References

Zhu Wang, Shuangge Ma and Ching-Yun Wang (2015) *Variable selection for zero-inflated and overdispersed data with application to health care demand in Germany*, *Biometrical Journal*. 57(5):867-84.

See Also[zipath](#)**Examples**

```
data("bioChemists", package = "pscl")
fm_zinb <- zipath(art ~ . | ., data = bioChemists, family = "negbin", nlambda=10, maxit.em=1)
res <- estfunReg(fm_zinb, which=which.min(fm_zinb$bic))
```

glmreg

*fit a GLM with lasso (or elastic net), snet or mnet regularization***Description**

Fit a generalized linear model via penalized maximum likelihood. The regularization path is computed for the lasso (or elastic net penalty), scad (or snet) and mcp (or mnet penalty), at a grid of values for the regularization parameter lambda. Fits linear, logistic, Poisson and negative binomial (fixed scale parameter) regression models.

Usage

```
## S3 method for class 'formula'
glmreg(formula, data, weights, offset=NULL, contrasts=NULL,
x.keep=FALSE, y.keep=TRUE, ...)
## S3 method for class 'matrix'
glmreg(x, y, weights, offset=NULL, ...)
## Default S3 method:
glmreg(x, ...)
```

Arguments

| | |
|----------------|---|
| formula | symbolic description of the model, see details. |
| data | argument controlling formula processing via model.frame . |
| weights | optional numeric vector of weights. If standardize=TRUE, weights are renormalized to weights/sum(weights). If standardize=FALSE, weights are kept as original input |
| offset | this can be used to specify an a priori known component to be included in the linear predictor during fitting. This should be NULL or a numeric vector of length equal to the number of cases. Currently only one offset term can be included in the formula. |
| x | input matrix, of dimension nobs x nvars; each row is an observation vector |
| y | response variable. Quantitative for family="gaussian". Non-negative counts for family="poisson" or family="negbin". For family="binomial" should be either a factor with two levels or a vector of proportions. |
| x.keep, y.keep | logical values: keep response variables or keep response variable? |
| contrasts | the contrasts corresponding to levels from the respective models |
| ... | Other arguments passing to glmreg_fit |

Details

The sequence of models implied by lambda is fit by coordinate descent. For family="gaussian" this is the lasso, mcp or scad sequence if alpha=1, else it is the enet, mnet or snet sequence. For the other families, this is a lasso (mcp, scad) or elastic net (mnet, snet) regularization path for fitting the generalized linear regression paths, by maximizing the appropriate penalized log-likelihood. Note that the objective function for "gaussian" is

$$1/2 * weights * RSS + \lambda * penalty,$$

if standardize=FALSE and

$$1/2 * \frac{weights}{\sum(weights)} * RSS + \lambda * penalty,$$

if standardize=TRUE. For the other models it is

$$- \sum(weights * loglik) + \lambda * penalty$$

if standardize=FALSE and

$$- \frac{weights}{\sum(weights)} * loglik + \lambda * penalty$$

if standardize=TRUE.

Value

An object with S3 class "glmreg" for the various types of models.

| | |
|---------------|--|
| call | the call that produced this object |
| b0 | Intercept sequence of length length(lambda) |
| beta | A nvars x length(lambda) matrix of coefficients. |
| lambda | The actual sequence of lambda values used |
| offset | the offset vector used. |
| dev | The computed deviance (for "gaussian", this is the R-square). The deviance calculations incorporate weights if present in the model. The deviance is defined to be 2*(loglike_sat - loglike), where loglike_sat is the log-likelihood for the saturated model (a model with a free parameter per observation). |
| nulldev | Null deviance (per observation). This is defined to be 2*(loglike_sat - loglike(NULL)); The NULL model refers to the intercept model. |
| nobs | number of observations |
| pll | penalized log-likelihood values for standardized coefficients in the IRLS iterations. For family="gaussian", not implemented yet. |
| pllres | penalized log-likelihood value for the estimated model on the original scale of coefficients |
| fitted.values | the fitted mean values, obtained by transforming the linear predictors by the inverse of the link function. |

Author(s)

Zhu Wang <wangz1@uthscsa.edu>

References

Breheeny, P. and Huang, J. (2011) *Coordinate descent algorithms for nonconvex penalized regression, with applications to biological feature selection*. *Ann. Appl. Statist.*, **5**: 232-253.

Zhu Wang, Shuangge Ma, Michael Zappitelli, Chirag Parikh, Ching-Yun Wang and Prasad Devarajan (2014) *Penalized Count Data Regression with Application to Hospital Stay after Pediatric Cardiac Surgery*, *Statistical Methods in Medical Research*. 2014 Apr 17. [Epub ahead of print]

See Also

[print](#), [predict](#), [coef](#) and [plot](#) methods, and the [cv.glmreg](#) function.

Examples

```
#binomial
x=matrix(rnorm(100*20),100,20)
g2=sample(0:1,100,replace=TRUE)
fit2=glmreg(x,g2,family="binomial")
#poisson and negative binomial
data("bioChemists", package = "pscl")
fm_pois <- glmreg(art ~ ., data = bioChemists, family = "poisson")
coef(fm_pois)
fm_nb1 <- glmreg(art ~ ., data = bioChemists, family = "negbin", theta=1)
coef(fm_nb1)
#offset
x <- matrix(rnorm(100*20),100,20)
y <- rpois(100, lambda=1)
exposure <- rep(0.5, length(y))
fit2 <- glmreg(x,y, lambda=NULL, nlambda=10, lambda.min.ratio=1e-4,
              offset=log(exposure), family="poisson")
predict(fit2, newx=x, newoffset=log(exposure))
## Not run:
fm_nb2 <- glmregNB(art ~ ., data = bioChemists)
coef(fm_nb2)

## End(Not run)
```

glmregNB

fit a negative binomial model with lasso (or elastic net), snet and mnet regularization

Description

Fit a negative binomial linear model via penalized maximum likelihood. The regularization path is computed for the lasso (or elastic net penalty), snet and mnet penalty, at a grid of values for the regularization parameter lambda.

Usage

```
glmregNB(formula, data, weights, offset=NULL, nlambda = 100, lambda=NULL,
lambda.min.ratio = ifelse(nobs<nvars,0.05,0.001), alpha=1, gamma=3, rescale=TRUE,
standardize = TRUE, penalty.factor = rep(1, nvars), thresh = 0.001, maxit.theta = 10,
maxit=1000, eps=.Machine$double.eps, trace=FALSE, start = NULL,
etastart = NULL, mustart = NULL, theta.fixed=FALSE, theta0=NULL,
init.theta=ifelse(!theta.fixed, theta0[1],NULL), link=log,
penalty=c("enet","mnet","snet"), method="glmreg_fit", model=TRUE,
x.keep=FALSE, y.keep=TRUE, contrasts=NULL, convex=FALSE, ...)
```

Arguments

| | |
|------------------|--|
| formula | formula used to describe a model. |
| data | argument controlling formula processing via <code>model.frame</code> . |
| weights | an optional vector of ‘prior weights’ to be used in the fitting process. Should be NULL or a numeric vector. Default is a vector of 1s with equal weight for each observation. |
| offset | optional numeric vector with an a priori known component to be included in the linear predictor of the model. |
| nlambda | The number of lambda values - default is 100. |
| lambda | A user supplied lambda sequence |
| lambda.min.ratio | Smallest value for lambda, as a fraction of lambda.max, the (data derived) entry value (i.e. the smallest value for which all coefficients are zero). The default depends on the sample size nobs relative to the number of variables nvars. If nobs > nvars, the default is 0.001, close to zero. If nobs < nvars, the default is 0.05. |
| alpha | The L2 penalty mixing parameter, with $0 \leq \alpha \leq 1$. alpha=1 is lasso (mcp, scad) penalty; and alpha=0 the ridge penalty. |
| gamma | The tuning parameter of the snet or mnet penalty. |
| rescale | logical value, if TRUE, adaptive rescaling of the penalty parameter for penalty="mnet" or penalty="snet" with family other than "gaussian". See reference |
| standardize | Logical flag for x variable standardization, prior to fitting the model sequence. The coefficients are always returned on the original scale. Default is standardize=TRUE. If variables are in the same units already, you might not wish to standardize. |
| penalty.factor | This is a number that multiplies lambda to allow differential shrinkage of coefficients. Can be 0 for some variables, which implies no shrinkage, and that variable is always included in the model. Default is same shrinkage for all variables. |
| thresh | Convergence threshold for coordinate descent. Defaults value is 1e-6. |
| maxit.theta | Maximum number of iterations for estimating theta scaling parameter |
| maxit | Maximum number of coordinate descent iterations for each lambda value; default is 1000. |
| eps | If a number is less than eps in magnitude, then this number is considered as 0 |

| | |
|-------------------------------|---|
| trace | If TRUE, fitting progress is reported |
| start, etastart, mustart, ... | arguments for the link{glmreg} function |
| init.theta | initial scaling parameter theta |
| theta.fixed | Estimate scale parameter theta? Default is FALSE. Note, the algorithm may become slow. In this case, one may use glmreg function with family="negbin", and a fixed theta. |
| theta0 | initial scale parameter vector theta, with length nlambda if theta.fixed=TRUE. Default is NULL |
| convex | Calculate index for which objective function ceases to be locally convex? Default is FALSE and only useful if penalty="mnet" or "snet". |
| link | link function, default is log |
| penalty | Type of regularization |
| method | estimation method |
| model, x.keep, y.keep | logicals. If TRUE the corresponding components of the fit (model frame, response, model matrix) are returned. |
| contrasts | the contrasts corresponding to levels from the respective models |

Details

The sequence of models implied by lambda is fit by coordinate descent. This is a lasso (mcp, scad) or elastic net (mnet, snet) regularization path for fitting the negative binomial linear regression paths, by maximizing the penalized log-likelihood. Note that the objective function is

$$- \sum (weights * loglik) + \lambda * penalty$$

if standardize=FALSE and

$$- \frac{weights}{\sum(weights)} * loglik + \lambda * penalty$$

if standardize=TRUE.

Value

An object with S3 class "glmreg", "glmregNB" for the various types of models.

| | |
|---------|---|
| call | the call that produced the model fit |
| b0 | Intercept sequence of length length(lambda) |
| beta | A nvars x length(lambda) matrix of coefficients. |
| lambda | The actual sequence of lambda values used |
| dev | The computed deviance. The deviance calculations incorporate weights if present in the model. The deviance is defined to be 2*(loglike_sat - loglike), where loglike_sat is the log-likelihood for the saturated model (a model with a free parameter per observation). |
| nulldev | Null deviance (per observation). This is defined to be 2*(loglike_sat - loglike(NULL)); The NULL model refers to the intercept model. |
| nobs | number of observations |

Author(s)

Zhu Wang <wangz1@uthscsa.edu>

References

Zhu Wang, Shuangge Ma, Michael Zappitelli, Chirag Parikh, Ching-Yun Wang and Prasad Devarajan (2014) *Penalized Count Data Regression with Application to Hospital Stay after Pediatric Cardiac Surgery*, *Statistical Methods in Medical Research*. 2014 Apr 17. [Epub ahead of print]

See Also

[print](#), [predict](#), [coef](#) and [plot](#) methods, and the [cv.glmregNB](#) function.

Examples

```
## Not run:
data("bioChemists", package = "pscl")
fm_nb <- glmregNB(art ~ ., data = bioChemists)
coef(fm_nb)
### ridge regression
fm <- glmregNB(art ~ ., alpha=0, data = bioChemists, lambda=seq(0.001, 1, by=0.01))
fm <- cv.glmregNB(art ~ ., alpha=0, data = bioChemists, lambda=seq(0.001, 1, by=0.01))

## End(Not run)
```

glmreg_fit

Internal function to fit a GLM with lasso (or elastic net), snet and mnet regularization

Description

Fit a generalized linear model via penalized maximum likelihood. The regularization path is computed for the lasso (or elastic net penalty), snet and mnet penalty, at a grid of values for the regularization parameter lambda. Fits linear, logistic, Poisson and negative binomial (fixed scale parameter) regression models.

Usage

```
glmreg_fit(x, y, weights, start=NULL, etastart=NULL, mustart=NULL, offset = rep(0, nobs),
nlambda=100, lambda=NULL, lambda.min.ratio=ifelse(nobs<nvars, .05, .001), alpha=1,
gamma=3, rescale=TRUE, standardize=TRUE, intercept=TRUE, penalty.factor = rep(1, nvars),
thresh=1e-6, eps.bino=1e-5, maxit=1000, eps=.Machine$double.eps, theta,
family=c("gaussian", "binomial", "poisson", "negbin"), penalty=c("enet", "mnet", "snet"),
convex=FALSE, x.keep=FALSE, y.keep=TRUE, trace=FALSE)
```

Arguments

| | |
|-------------------------------|---|
| <code>x</code> | input matrix, of dimension <code>nobs</code> x <code>nvars</code> ; each row is an observation vector. |
| <code>y</code> | response variable. Quantitative for <code>family="gaussian"</code> . Non-negative counts for <code>family="poisson"</code> or <code>family="negbin"</code> . For <code>family="binomial"</code> should be either a factor with two levels or a vector of proportions. |
| <code>weights</code> | observation weights. Can be total counts if responses are proportion matrices. Default is 1 for each observation |
| <code>start</code> | starting values for the parameters in the linear predictor. |
| <code>etastart</code> | starting values for the linear predictor. |
| <code>mustart</code> | starting values for the vector of means. |
| <code>offset</code> | this can be used to specify an a priori known component to be included in the linear predictor during fitting. This should be <code>NULL</code> or a numeric vector of length equal to the number of cases. Currently only one offset term can be included in the formula. |
| <code>nlambda</code> | The number of lambda values - default is 100. The sequence may be truncated before <code>nlambda</code> is reached if a close to saturated model is fitted. See also <code>satu</code> . |
| <code>lambda</code> | by default, the algorithm provides a sequence of regularization values, or a user supplied lambda sequence |
| <code>lambda.min.ratio</code> | Smallest value for lambda, as a fraction of <code>lambda.max</code> , the (data derived) entry value (i.e. the smallest value for which all coefficients are zero except the intercept). Note, there is no closed formula for <code>lambda.max</code> in general. If <code>rescale=TRUE</code> , <code>lambda.max</code> is the same for <code>penalty="mnet"</code> or <code>"snet"</code> . Otherwise, some modifications are required. For instance, for small gamma value, half of the square root (if <code>lambda.max</code> is too small) of the computed <code>lambda.max</code> can be used when <code>penalty="mnet"</code> or <code>"snet"</code> . The default of <code>lambda.min.ratio</code> depends on the sample size <code>nobs</code> relative to the number of variables <code>nvars</code> . If <code>nobs > nvars</code> , the default is <code>0.001</code> , close to zero. If <code>nobs < nvars</code> , the default is <code>0.05</code> . |
| <code>alpha</code> | The L_2 penalty mixing parameter, with $0 \leq \alpha \leq 1$. <code>alpha=1</code> is lasso (mcp, scad) penalty; and <code>alpha=0</code> the ridge penalty. However, if <code>alpha=0</code> , one must provide lambda values. |
| <code>gamma</code> | The tuning parameter of the <code>snet</code> or <code>mnet</code> penalty. |
| <code>rescale</code> | logical value, if <code>TRUE</code> , adaptive rescaling of the penalty parameter for <code>penalty="mnet"</code> or <code>penalty="snet"</code> with family other than "gaussian". See reference |
| <code>standardize</code> | logical value for x variable standardization, prior to fitting the model sequence. The coefficients are always returned on the original scale. Default is <code>standardize=TRUE</code> . |
| <code>intercept</code> | logical value: if <code>TRUE</code> (default), <code>intercept(s)</code> are fitted; otherwise, <code>intercept(s)</code> are set to zero |
| <code>penalty.factor</code> | This is a number that multiplies lambda to allow differential shrinkage of coefficients. Can be 0 for some variables, which implies no shrinkage, and that variable is always included in the model. Default is same shrinkage for all variables. |

| | |
|----------------|--|
| thresh | Convergence threshold for coordinate descent. Defaults value is 1e-6. |
| eps.bino | a lower bound of probabilities to be truncated, for computing weights and related values when family="binomial". It is also used when family="negbin". |
| maxit | Maximum number of coordinate descent iterations for each lambda value; default is 1000. |
| eps | If a coefficient is less than eps in magnitude, then it is reported to be 0 |
| convex | Calculate index for which objective function ceases to be locally convex? Default is FALSE and only useful if penalty="mnet" or "snet". |
| theta | an overdispersion scaling parameter for family="negbin" |
| family | Response type (see above) |
| penalty | Type of regularization |
| x.keep, y.keep | For glmreg: logical values indicating whether the response vector and model matrix used in the fitting process should be returned as components of the returned value. For glmreg_fit: x is a design matrix of dimension n * p, and x is a vector of observations of length n. |
| trace | If TRUE, fitting progress is reported |

Details

The sequence of models implied by lambda is fit by coordinate descent. For family="gaussian" this is the lasso, mcp or scad sequence if alpha=1, else it is the enet, mnet or snet sequence. For the other families, this is a lasso (mcp, scad) or elastic net (mnet, snet) regularization path for fitting the generalized linear regression paths, by maximizing the appropriate penalized log-likelihood. Note that the objective function for "gaussian" is

$$1/2 * weights * RSS + \lambda * penalty,$$

if standardize=FALSE and

$$1/2 * \frac{weights}{\sum(weights)} * RSS + \lambda * penalty,$$

if standardize=TRUE. For the other models it is

$$- \sum(weights * loglik) + \lambda * penalty$$

if standardize=FALSE and

$$- \frac{weights}{\sum(weights)} * loglik + \lambda * penalty$$

if standardize=TRUE.

Value

An object with S3 class "glmreg" for the various types of models.

| | |
|------|---|
| call | the call that produced the model fit |
| b0 | Intercept sequence of length length(lambda) |

| | |
|---------|--|
| beta | A $nvars \times length(\lambda)$ matrix of coefficients. |
| lambda | The actual sequence of λ values used |
| satu | satu=1 if a saturated model (deviance/null deviance < 0.05) is fit. Otherwise satu=0. The number of $n\lambda$ sequence may be truncated before $n\lambda$ is reached if satu=1. |
| dev | The computed deviance (for "gaussian", this is the R-square). The deviance calculations incorporate weights if present in the model. The deviance is defined to be $2*(\loglike_sat - \loglike)$, where \loglike_sat is the log-likelihood for the saturated model (a model with a free parameter per observation). |
| nulldev | Null deviance (per observation). This is defined to be $2*(\loglike_sat - \loglike(Null))$; The NULL model refers to the intercept model. |
| nobs | number of observations |

Author(s)

Zhu Wang <wangz1@uthscsa.edu>

References

Breheny, P. and Huang, J. (2011) *Coordinate descent algorithms for nonconvex penalized regression, with applications to biological feature selection*. *Ann. Appl. Stat.*, **5**: 232-253.

Zhu Wang, Shuangge Ma, Michael Zappitelli, Chirag Parikh, Ching-Yun Wang and Prasad Devarajan (2014) *Penalized Count Data Regression with Application to Hospital Stay after Pediatric Cardiac Surgery*, *Statistical Methods in Medical Research*. 2014 Apr 17. [Epub ahead of print]

See Also

[glmreg](#)

hessianReg

Hessian Matrix of Regularized Estimators

Description

Constructing Hessian matrix for regularized regression parameters.

Usage

```
hessianReg(x, which, ...)
```

Arguments

| | |
|-------|---|
| x | a fitted model object. |
| which | which penalty parameter(s)? |
| ... | arguments passed to the meatReg function. |

Details

hessianReg is a function to compute the Hessian matrix estimate of non-zero regularized estimators. Implemented only for zipath object with family="negbin" in the current version.

Value

A matrix containing the Hessian matrix estimate for the non-zero parameters.

Author(s)

Zhu Wang <wangz1@uthscsa.edu>

References

Zhu Wang, Shuangge Ma and Ching-Yun Wang (2015) *Variable selection for zero-inflated and overdispersed data with application to health care demand in Germany*, *Biometrical Journal*. 57(5):867-84.

See Also

[breadReg](#), [meatReg](#)

Examples

```
data("bioChemists", package = "pscl")
fm_zinb <- zipath(art ~ . | ., data = bioChemists, family = "negbin", nlambda=10, maxit.em=1)
hessianReg(fm_zinb, which=which.min(fm_zinb$bic))
```

meatReg

Meat Matrix Estimator

Description

Estimating the variance of the first derivative of log-likelihood function

Usage

```
meatReg(x, which, ...)
```

Arguments

x a fitted model object. Currently only implemented for zipath object with family="negbin"
 which which penalty parameter(s)?
 ... arguments passed to the [estfunReg](#) function.

Details

See reference below

Value

A

 $k \times k$

covariance matrix of first derivative of log-likelihood function

Author(s)

Zhu Wang <wangz1@uthscsa.edu>

References

Zhu Wang, Shuangge Ma and Ching-Yun Wang (2015) *Variable selection for zero-inflated and overdispersed data with application to health care demand in Germany*, *Biometrical Journal*. 57(5):867-84.

See Also

[sandwichReg](#), [breadReg](#), [estfunReg](#)

Examples

```
data("bioChemists", package = "pscl")
fm_zinb <- zipath(art ~ . | ., data = bioChemists, family = "negbin", nlambda=10, maxit.em=1)
meatReg(fm_zinb, which=which.min(fm_zinb$bic))
```

 methods

Methods for mpath Objects

Description

Methods for models fitted by coordinate descent algorithms.

Usage

```
## S3 method for class 'glmreg'
AIC(object, ..., k)
## S3 method for class 'zipath'
AIC(object, ..., k)
## S3 method for class 'glmreg'
BIC(object, ...)
## S3 method for class 'zipath'
BIC(object, ...)
```

Arguments

| | |
|--------|--|
| object | objects of class glmreg or zipath. |
| ... | additional arguments passed to callies. |
| k | numeric, the <i>penalty</i> per parameter to be used; the default k = 2 is the classical AIC. k has been hard coded in the function and there is no impact to the value of AIC if k is changed |

Author(s)

Zhu Wang <wangz1@uthscsa.edu>

References

Zhu Wang, Shuangge Ma, Michael Zappitelli, Chirag Parikh, Ching-Yun Wang and Prasad Devarajan (2014) *Penalized Count Data Regression with Application to Hospital Stay after Pediatric Cardiac Surgery*, *Statistical Methods in Medical Research*. 2014 Apr 17. [Epub ahead of print]

Zhu Wang, Shuangge Ma, Ching-Yun Wang, Michael Zappitelli, Prasad Devarajan and Chirag R. Parikh (2014) *EM for Regularized Zero Inflated Regression Models with Applications to Postoperative Morbidity after Cardiac Surgery in Children*, *Statistics in Medicine*. 33(29):5192-208.

Zhu Wang, Shuangge Ma and Ching-Yun Wang (2015) *Variable selection for zero-inflated and overdispersed data with application to health care demand in Germany*, *Biometrical Journal*. 57(5):867-84.

ncl *fit a nonconvex loss based robust linear model*

Description

Fit a linear model via penalized nonconvex loss function.

Usage

```
## S3 method for class 'formula'
ncl(formula, data, weights, offset=NULL, contrasts=NULL,
x.keep=FALSE, y.keep=TRUE, ...)
## S3 method for class 'matrix'
ncl(x, y, weights, offset=NULL, ...)
## Default S3 method:
ncl(x, ...)
```

Arguments

| | |
|----------------|---|
| formula | symbolic description of the model, see details. |
| data | argument controlling formula processing via <code>model.frame</code> . |
| weights | optional numeric vector of weights. If <code>standardize=TRUE</code> , weights are renormalized to <code>weights/sum(weights)</code> . If <code>standardize=FALSE</code> , weights are kept as original input |
| x | input matrix, of dimension <code>nobs x nvars</code> ; each row is an observation vector |
| y | response variable. Quantitative for <code>rfamily="clossR"</code> and <code>-1/1</code> for classification. |
| offset | Not implemented yet |
| contrasts | the contrasts corresponding to <code>levels</code> from the respective models |
| x.keep, y.keep | For <code>glmreg</code> : logical values indicating whether the response vector and model matrix used in the fitting process should be returned as components of the returned value. For <code>ncl_fit</code> : x is a design matrix of dimension <code>n * p</code> , and x is a vector of observations of length <code>n</code> . |
| ... | Other arguments passing to <code>ncl_fit</code> |

Details

The robust linear model is fit by majorization-minimization along with linear regression. Note that the objective function is

$$1/2 * weights * loss$$

Value

An object with S3 class "ncl" for the various types of models.

| | |
|---------------|--|
| call | the call that produced this object |
| fitted.values | predicted values |
| h | pseudo response values in the MM algorithm |

Author(s)

Zhu Wang <wangz1@uthscsa.edu>

References

Zhu Wang (2019) *MM for Penalized Estimation*, <https://arxiv.org/abs/1912.11119>

See Also

[print](#), [predict](#), [coef](#).

Examples

```
#binomial
x=matrix(rnorm(100*20),100,20)
g2=sample(c(-1,1),100,replace=TRUE)
fit=ncl(x,g2,s=1,rfamily="closs")
```

| | |
|--------|--|
| nclreg | <i>fit a nonconvex loss based robust linear model with lasso (or elastic net), snet or mnet regularization</i> |
|--------|--|

Description

Fit a linear model via penalized nonconvex loss function. The regularization path is computed for the lasso (or elastic net penalty), scad (or snet) and mcp (or mnet penalty), at a grid of values for the regularization parameter lambda.

Usage

```
## S3 method for class 'formula'
nclreg(formula, data, weights, offset=NULL, contrasts=NULL, ...)
## S3 method for class 'matrix'
nclreg(x, y, weights, offset=NULL, ...)
## Default S3 method:
nclreg(x, ...)
```

Arguments

| | |
|-----------|---|
| formula | symbolic description of the model, see details. |
| data | argument controlling formula processing via model.frame . |
| weights | optional numeric vector of weights. If standardize=TRUE, weights are renormalized to weights/sum(weights). If standardize=FALSE, weights are kept as original input |
| x | input matrix, of dimension nobs x nvars; each row is an observation vector |
| y | response variable. Quantitative for rfamily="clossR" and -1/1 for classification. |
| offset | Not implemented yet |
| contrasts | the contrasts corresponding to levels from the respective models |
| ... | Other arguments passing to nclreg_fit |

Details

The sequence of robust models implied by `lambda` is fit by majorization-minimization along with coordinate descent. Note that the objective function is

$$1/2 * weights * loss + \lambda * penalty,$$

if `standardize=FALSE` and

$$1/2 * \frac{weights}{\sum(weights)} * loss + \lambda * penalty,$$

if `standardize=TRUE`.

Value

An object with S3 class "nclreg" for the various types of models.

| | |
|----------------------------|---|
| <code>call</code> | the call that produced this object |
| <code>b0</code> | Intercept sequence of length <code>length(lambda)</code> |
| <code>beta</code> | A <code>nvars x length(lambda)</code> matrix of coefficients. |
| <code>lambda</code> | The actual sequence of <code>lambda</code> values used |
| <code>nobs</code> | number of observations |
| <code>risk</code> | if <code>type.path="nonactive"</code> , a matrix with number of rows <code>iter</code> and number of columns <code>nlambda</code> , loss values along the regularization path. If <code>type.path="fast"</code> , a vector of length <code>nlambda</code> , loss values along the regularization path |
| <code>p11</code> | if <code>type.path="nonactive"</code> , a matrix with number of rows <code>iter</code> and number of columns <code>nlambda</code> , penalized loss values along the regularization path. If <code>type.path="fast"</code> , a vector of length <code>nlambda</code> , penalized loss values along the regularization path |
| <code>fitted.values</code> | predicted values depending on <code>standardize</code> , internal use only |

Author(s)

Zhu Wang <wangz1@uthscsa.edu>

References

Zhu Wang (2019) *MM for Penalized Estimation*, <https://arxiv.org/abs/1912.11119>

See Also

[print](#), [predict](#), [coef](#) and [plot](#) methods, and the [cv.nclreg](#) function.

Examples

```
#binomial
x=matrix(rnorm(100*20),100,20)
g2=sample(c(-1,1),100,replace=TRUE)
### different solution paths via a combination of type.path, decreasing and type.init
fit1=nclreg(x,g2,s=1,rfamily="closs",type.path="active",decreasing=TRUE,type.init="bst")
fit2=nclreg(x,g2,s=1,rfamily="closs",type.path="active",decreasing=FALSE,type.init="bst")
fit3=nclreg(x,g2,s=1,rfamily="closs",type.path="nonactive",decreasing=TRUE,type.init="bst")
fit4=nclreg(x,g2,s=1,rfamily="closs",type.path="nonactive",decreasing=FALSE,type.init="bst")
fit5=nclreg(x,g2,s=1,rfamily="closs",type.path="active",decreasing=TRUE,type.init="ncl")
fit6=nclreg(x,g2,s=1,rfamily="closs",type.path="active",decreasing=FALSE,type.init="ncl")
fit7=nclreg(x,g2,s=1,rfamily="closs",type.path="nonactive",decreasing=TRUE,type.init="ncl")
fit8=nclreg(x,g2,s=1,rfamily="closs",type.path="nonactive",decreasing=FALSE,type.init="ncl")
```

| | |
|------------|--|
| nclreg_fit | <i>Internal function to fit a nonconvex loss based robust linear model with lasso (or elastic net), snet and mnet regularization</i> |
|------------|--|

Description

Fit a linear model via penalized nonconvex loss function. The regularization path is computed for the lasso (or elastic net penalty), scad (or snet) and mcp (or mnet penalty), at a grid of values for the regularization parameter lambda.

Usage

```
nclreg_fit(x, y, weights, offset=NULL, cost=0.5,
  rfamily=c("clossR", "closs", "gloss", "qloss"),
  s=NULL, fk=NULL, iter=10, reltol=1e-5,
  penalty=c("enet", "mnet", "snet"), nlambda=100, lambda=NULL,
  type.path=c("active", "nonactive", "onestep"), decreasing=FALSE,
  lambda.min.ratio=ifelse(nobs<nvars,.05, .001), alpha=1, gamma=3,
  standardize=TRUE, intercept=TRUE, penalty.factor = NULL, maxit=1000,
  type.init=c("bst", "ncl", "heu"), mstop.init=10, nu.init=0.1,
  eps=.Machine$double.eps, epscycle=10, thresh=1e-6, trace=FALSE)
```

Arguments

| | |
|---------|---|
| x | input matrix, of dimension nobs x nvars; each row is an observation vector. |
| y | response variable. Quantitative for rfamily="clossR" and -1/1 for classifications. |
| weights | observation weights. Can be total counts if responses are proportion matrices. Default is 1 for each observation |
| offset | this can be used to specify an a priori known component to be included in the linear predictor during fitting. This should be NULL or a numeric vector of length equal to the number of cases. Currently only one offset term can be included in the formula. |

| | |
|------------------|--|
| cost | price to pay for false positive, $0 < \text{cost} < 1$; price of false negative is $1 - \text{cost}$. |
| rfamily | Response type and relevant loss functions (see above) |
| s | nonconvex loss tuning parameter for robust regression and classification. |
| fk | predicted values at an iteration in the MM algorithm |
| nlambda | The number of lambda values - default is 100. The sequence may be truncated before nlambda is reached if a close to saturated model is fitted. See also satu. |
| lambda | by default, the algorithm provides a sequence of regularization values, or a user supplied lambda sequence |
| type.path | solution path. If <code>type.path="active"</code> , then cycle through only the active set in the next increasing lambda sequence. If <code>type.path="nonactive"</code> , no active set for each element of the lambda sequence and cycle through all the predictor variables. If <code>type.path="onestep"</code> , update for one element of lambda depending on <code>decreasing=FALSE</code> (last element of lambda) or <code>decreasing=TRUE</code> (then first element of lambda) in each MM iteration, and iterate until convergency of prediction. Then fit a solution path based on the sequence of lambda. |
| lambda.min.ratio | Smallest value for lambda, as a fraction of lambda.max, the (data derived) entry value (i.e. the smallest value for which all coefficients are zero except the intercept). Note, there is no closed formula for lambda.max. The default of lambda.min.ratio depends on the sample size nobs relative to the number of variables nvars. If $\text{nobs} > \text{nvars}$, the default is 0.001 , close to zero. If $\text{nobs} < \text{nvars}$, the default is 0.05 . |
| alpha | The L_2 penalty mixing parameter, with $0 \leq \alpha \leq 1$. $\alpha=1$ is lasso (mcp, scad) penalty; and $\alpha=0$ the ridge penalty. However, if $\alpha=0$, one must provide lambda values. |
| gamma | The tuning parameter of the snet or mnet penalty. |
| standardize | logical value for x variable standardization, prior to fitting the model sequence. The coefficients are always returned on the original scale. Default is <code>standardize=TRUE</code> . |
| intercept | logical value: if TRUE (default), intercept(s) are fitted; otherwise, intercept(s) are set to zero |
| penalty.factor | This is a number that multiplies lambda to allow differential shrinkage of coefficients. Can be 0 for some variables, which implies no shrinkage, and that variable is always included in the model. Default is same shrinkage for all variables. |
| type.init | a method to determine the initial values. If <code>type.init="ncl"</code> , an intercept-only model as initial parameter and run nclreg regularization path forward from lambda_max to lambda_min. If <code>type.init="heu"</code> , heuristic initial parameters and run nclreg path backward or forward depending on decreasing, between lambda_min and lambda_max. If <code>type.init="bst"</code> , run a boosting model with bst in package bst, depending on <code>mstop.init</code> , <code>nu.init</code> and run nclreg backward or forward depending on decreasing. |
| mstop.init | an integer giving the number of boosting iterations when <code>type.init="bst"</code> |
| nu.init | a small number (between 0 and 1) defining the step size or shrinkage parameter when <code>type.init="bst"</code> . |

| | |
|------------|--|
| decreasing | only used if lambda=NULL. direction=FALSE for decreasing sequence of lambda, used to determine regularization path direction either from lambda_max to a potentially modified lambda_min or vice versa if type.init="bst", "heu". Since this is a nonconvex optimization, it is possible to generate different estimates for the same lambda depending on decreasing since the choice of decreasing picks different starting values. |
| iter | number of iteration in the MM algorithm |
| maxit | Within each MM algorithm iteration, maximum number of coordinate descent iterations for each lambda value; default is 1000. |
| reltol | convergency criteria |
| eps | If a coefficient is less than eps in magnitude, then it is reported to be 0 |
| epscycle | If nlambda > 1 and the relative loss values from two consecutive lambda values change > epscycle, then re-estimate parameters in an effort to avoid trap of local optimization. |
| thresh | Convergence threshold for coordinate descent. Defaults value is 1e-6. |
| penalty | Type of regularization |
| trace | If TRUE, fitting progress is reported |

Details

The sequence of robust models implied by lambda is fit by majorization-minimization along with coordinate descent. Note that the objective function is

$$1/2 * weights * loss + \lambda * penalty,$$

if standardize=FALSE and

$$1/2 * \frac{weights}{\sum(weights)} * loss + \lambda * penalty,$$

if standardize=TRUE.

Value

An object with S3 class "nclreg" for the various types of models.

| | |
|------------|--|
| call | the call that produced the model fit |
| b0 | Intercept sequence of length length(lambda) |
| beta | A nvars x length(lambda) matrix of coefficients. |
| lambda | The actual sequence of lambda values used |
| decreasing | if lambda is an increasing sequence or not, used to determine regularization path direction either from lambda_max to a potentially modified lambda_min or vice versa if type.init="bst", "heu". |

Author(s)

Zhu Wang <wangz1@uthscsa.edu>

References

Zhu Wang (2019) *MM for Penalized Estimation*, <https://arxiv.org/abs/1912.11119>

See Also

[nclreg](#)

ncl_fit

Internal function to fit a nonconvex loss based robust linear model

Description

Fit a linear model via penalized nonconvex loss function.

Usage

```
ncl_fit(x,y, weights, offset=NULL, cost=0.5,
        rfamily=c("clossR", "closs", "gloss", "qloss"),
        s=NULL, fk=NULL, iter=10, reltol=1e-5, trace=FALSE)
```

Arguments

| | |
|---------|---|
| x | input matrix, of dimension nobs x nvars; each row is an observation vector. |
| y | response variable. Quantitative for rfamily="clossR" and -1/1 for classifications. |
| weights | observation weights. Can be total counts if responses are proportion matrices. Default is 1 for each observation |
| offset | this can be used to specify an a priori known component to be included in the linear predictor during fitting. This should be NULL or a numeric vector of length equal to the number of cases. Currently only one offset term can be included in the formula. |
| cost | price to pay for false positive, $0 < \text{cost} < 1$; price of false negative is $1 - \text{cost}$. |
| rfamily | Response type and relevant loss functions (see above) |
| s | nonconvex loss tuning parameter for robust regression and classification. |
| fk | predicted values at an iteration in the MM algorithm |
| iter | number of iteration in the MM algorithm |
| reltol | convergency criteria |
| trace | If TRUE, fitting progress is reported |

Details

The robust linear model is fit by majorization-minimization along with least squares. Note that the objective function is

$$1/2 * weights * loss$$

Value

An object with S3 class "ncl" for the various types of models.

| | |
|---------------|--|
| call | the call that produced the model fit |
| fitted.values | predicted values |
| h | pseudo response values in the MM algorithm |

Author(s)

Zhu Wang <wangz1@uthscsa.edu>

References

Zhu Wang (2019) *MM for Penalized Estimation*, <https://arxiv.org/abs/1912.11119>

See Also

[ncl](#)

| | |
|-------------|---|
| plot.glmreg | <i>plot coefficients from a "glmreg" object</i> |
|-------------|---|

Description

Produces a coefficient profile plot of the coefficient paths for a fitted "glmreg" object.

Usage

```
## S3 method for class 'glmreg'
plot(x, xvar = c("norm", "lambda", "dev"), label = FALSE, shade=TRUE, ...)
```

Arguments

| | |
|-------|--|
| x | fitted "glmreg" model |
| xvar | What is on the X-axis. "norm" plots against the L1-norm of the coefficients, "lambda" against the log-lambda sequence, and "dev" against the percent deviance explained. |
| label | If TRUE, label the curves with variable sequence numbers. |
| shade | Should nonconvex region be shaded? Default is TRUE. Code developed for all weights=1 only |
| ... | Other graphical parameters to plot |

Details

A coefficient profile plot is produced.

Author(s)

Zhu Wang wangz1@uthscsa.edu

See Also

glmreg, and print, predict and coef methods.

Examples

```
x=matrix(rnorm(100*20),100,20)
y=rnorm(100)
fit1=glmreg(x,y)
plot(fit1)
plot(fit1,xvar="lambda",label=TRUE)
```

| | |
|----------------|---|
| predict.glmreg | <i>Model predictions based on a fitted "glmreg" object.</i> |
|----------------|---|

Description

This function returns predictions from a fitted "glmreg" object.

Usage

```
## S3 method for class 'glmreg'
predict(object,newx,which=1:length(object$lambda),
type=c("link","response","class","coefficients","nonzero"), newoffset = NULL,
na.action=na.pass, ...)
## S3 method for class 'glmreg'
coef(object,which=1:length(object$lambda),...)
```

Arguments

| | |
|-----------|---|
| object | Fitted "glmreg" model object. |
| newx | Matrix of values at which predictions are to be made. Not used for type="coefficients" |
| which | Indices of the penalty parameter lambda at which predictions are required. By default, all indices are returned. |
| type | Type of prediction: "link" returns the linear predictors; "response" gives the fitted values; "class" returns the binomial outcome with the highest probability; "coefficients" returns the coefficients. |
| newoffset | an offset term used in prediction |
| na.action | action for missing data value |
| ... | arguments for predict |

Value

The returned object depends on type.

Author(s)

Zhu Wang <wangz1@uthscsa.edu>

References

Zhu Wang, Shuangge Ma, Michael Zappitelli, Chirag Parikh, Ching-Yun Wang and Prasad Devarajan (2014) *Penalized Count Data Regression with Application to Hospital Stay after Pediatric Cardiac Surgery*, *Statistical Methods in Medical Research*. 2014 Apr 17. [Epub ahead of print]

See Also

[glmreg](#)

Examples

```
## Dobson (1990) Page 93: Randomized Controlled Trial :
counts <- c(18,17,15,20,10,20,25,13,12)
outcome <- gl(3,1,9)
treatment <- gl(3,3)
print(d.AD <- data.frame(treatment, outcome, counts))
fit <- glmreg(counts ~ outcome + treatment, data=d.AD, family="poisson")
predict(fit, newx=d.AD[,1:2])
summary(fit)
coef(fit)
```

predict.zipath

Methods for zipath Objects

Description

Methods for extracting information from fitted penalized zero-inflated regression model objects of class "zipath".

Usage

```
## S3 method for class 'zipath'
predict(object, newdata, which = 1:object$nlambda,
  type = c("response", "prob", "count", "zero", "nonzero"), na.action = na.pass,
  at = NULL, ...)
## S3 method for class 'zipath'
residuals(object, type = c("pearson", "response"), ...)

## S3 method for class 'zipath'
coef(object, which=1:object$nlambda, model = c("full", "count", "zero"), ...)

## S3 method for class 'zipath'
terms(x, model = c("count", "zero"), ...)
## S3 method for class 'zipath'
model.matrix(object, model = c("count", "zero"), ...)
```

Arguments

| | |
|-----------|--|
| object, x | an object of class "zipath" as returned by <code>zipath</code> . |
| newdata | optionally, a data frame in which to look for variables with which to predict. If omitted, the original observations are used. |
| which | Indices of the penalty parameters lambda at which predictions are required. By default, all indices are returned. |
| type | character specifying the type of predictions or residuals, respectively. For details see below. |
| na.action | function determining what should be done with missing values in newdata. The default is to predict NA. |
| at | optionally, if type = "prob", a numeric vector at which the probabilities are evaluated. By default $0:\max(y)$ is used where y is the original observed response. |
| model | character specifying for which component of the model the terms or model matrix should be extracted. |
| ... | currently not used. |

Details

Re-uses the design of function `zeroinfl` in package `pscl` (see reference). A set of standard extractor functions for fitted model objects is available for objects of class "zipath", including methods to the generic functions `print` and `summary` which print the estimated coefficients along with some further information. As usual, the `summary` method returns an object of class "summary.zipath" containing the relevant summary statistics which can subsequently be printed using the associated `print` method.

The methods for `coef` by default return a single vector of coefficients and their associated covariance matrix, respectively, i.e., all coefficients are concatenated. By setting the `model` argument, the estimates for the corresponding model components can be extracted.

Both the `fitted` and `predict` methods can compute fitted responses. The latter additionally provides the predicted density (i.e., probabilities for the observed counts), the predicted mean from the count component (without zero inflation) and the predicted probability for the zero component. The `residuals` method can compute raw residuals (observed - fitted) and Pearson residuals (raw residuals scaled by square root of variance function).

Author(s)

Zhu Wang <wangz1@uthscsa.edu>

References

- Zhu Wang, Shuangge Ma, Michael Zappitelli, Chirag Parikh, Ching-Yun Wang and Prasad Devarajan (2014) *Penalized Count Data Regression with Application to Hospital Stay after Pediatric Cardiac Surgery*, *Statistical Methods in Medical Research*. 2014 Apr 17. [Epub ahead of print]
- Zhu Wang, Shuangge Ma, Ching-Yun Wang, Michael Zappitelli, Prasad Devarajan and Chirag R. Parikh (2014) *EM for Regularized Zero Inflated Regression Models with Applications to Postoperative Morbidity after Cardiac Surgery in Children*, *Statistics in Medicine*. 33(29):5192-208.

Zhu Wang, Shuangge Ma and Ching-Yun Wang (2015) *Variable selection for zero-inflated and overdispersed data with application to health care demand in Germany*, *Biometrical Journal*. 57(5):867-84.

See Also

[zipath](#)

Examples

```
## Not run:
data("bioChemists", package = "pscl")
fm_zip <- zipath(art ~ . | ., data = bioChemists, nlambda=10)
plot(residuals(fm_zip) ~ fitted(fm_zip))
coef(fm_zip, model = "count")
coef(fm_zip, model = "zero")
summary(fm_zip)
logLik(fm_zip)

## End(Not run)
```

| | |
|-------------|--|
| pval.zipath | <i>compute p-values from penalized zero-inflated model with multi-split data</i> |
|-------------|--|

Description

compute p-values from penalized zero-inflated Poisson, negative binomial and geometric model with multi-split data

Usage

```
pval.zipath(formula, data, weights, subset, na.action, offset, standardize=TRUE,
family = c("poisson", "negbin", "geometric"),penalty = c("enet", "mnet", "snet"),
gamma.count = 3, gamma.zero = 3, prop=0.5, trace=TRUE, B=10, ...)
```

Arguments

| | |
|-------------|---|
| formula | symbolic description of the model, see details. |
| data | argument controlling formula processing via model.frame . |
| weights | optional numeric vector of weights. If standardize=TRUE, weights are renormalized to weights/sum(weights). If standardize=FALSE, weights are kept as original input |
| subset | subset of data |
| na.action | how to deal with missing data |
| offset | Not implemented yet |
| standardize | logical value, should variables be standardized? |

| | |
|-------------|---|
| family | family to fit zipath |
| penalty | penalty considered as one of enet, mnet, snet. |
| gamma.count | The tuning parameter of the snet or mnet penalty for the count part of model. |
| gamma.zero | The tuning parameter of the snet or mnet penalty for the zero part of model. |
| prop | proportion of data split, default is 50/50 split |
| trace | logical value, if TRUE, print detailed calculation results |
| B | number of repeated multi-split replications |
| ... | Other arguments passing to glmreg_fit |

Details

compute p-values from penalized zero-inflated Poisson, negative binomial and geometric model with multi-split data

Value

| | |
|--------------|-------------------------------------|
| count.pval | raw p-values in the count component |
| zero.pval | raw p-values in the zero component |
| count.pval.q | Q value for the count component |
| zero.pval.q | Q value for the zero component |

Author(s)

Zhu Wang <wangz1@uthscsa.edu>

References

Nicolai Meinshausen, Lukas Meier and Peter Buehlmann (2013) *p-Values for High-Dimensional Regression*, *Journal of the American Statistical Association*, 104(488), 1671–1681.

Zhu Wang, Shuangge Ma, Ching-Yun Wang, Michael Zappitelli, Prasad Devarajan and Chirag R. Parikh (2014) *EM for Regularized Zero Inflated Regression Models with Applications to Postoperative Morbidity after Cardiac Surgery in Children*, *Statistics in Medicine*. 33(29):5192-208.

Zhu Wang, Shuangge Ma and Ching-Yun Wang (2015) *Variable selection for zero-inflated and overdispersed data with application to health care demand in Germany*, *Biometrical Journal*. 57(5):867-84.

sandwichReg

*Making Sandwiches with Bread and Meat for Regularized Estimators***Description**

Constructing sandwich covariance matrix estimators by multiplying bread and meat matrices for regularized regression parameters.

Usage

```
sandwichReg(x, breadreg.=breadReg, meatreg.=meatReg, which, log=FALSE, ...)
```

Arguments

| | |
|-----------|--|
| x | a fitted model object. |
| breadreg. | either a breadReg matrix or a function for computing this via breadreg.(x). |
| meatreg. | either a breadReg matrix or a function for computing this via meatreg.(x, ...). |
| which | which penalty parameters(s) to compute? |
| log | if TRUE, the corresponding element is with respect to log(theta) in negative binomial regression. Otherwise, for theta |
| ... | arguments passed to the meatReg function. |

Details

sandwichReg is a function to compute an estimator for the covariance of the non-zero parameters. It takes a breadReg matrix (i.e., estimator of the expectation of the negative derivative of the penalized estimating functions) and a meatReg matrix (i.e., estimator of the variance of the log-likelihood function) and multiplies them to a sandwich with meat between two slices of bread. By default `breadReg` and `meatReg` are called. Implemented only for zipath object with family="negbin" in the current version.

Value

A matrix containing the sandwich covariance matrix estimate for the non-zero parameters.

Author(s)

Zhu Wang <wangz1@uthscsa.edu>

References

Zhu Wang, Shuangge Ma and Ching-Yun Wang (2015) *Variable selection for zero-inflated and overdispersed data with application to health care demand in Germany*, *Biometrical Journal*. 57(5):867-84.

See Also

[breadReg](#), [meatReg](#)

Examples

```
data("bioChemists", package = "pscl")
fm_zinb <- zipath(art ~ . | ., data = bioChemists, family = "negbin", nlambda=10, maxit.em=1)
sandwichReg(fm_zinb, which=which.min(fm_zinb$bic))
```

se

Standard Error of Regularized Estimators

Description

Generic function for computing standard errors of non-zero regularized estimators

Usage

```
se(x, which, log=TRUE, ...)
```

Arguments

| | |
|-------|--|
| x | a fitted model object. |
| which | which penalty parameter(s)? |
| log | if TRUE, the computed standard error is for log(theta) for negative binomial regression, otherwise, for theta. |
| ... | arguments passed to methods. |

Value

A vector containing standard errors of non-zero regularized estimators.

Author(s)

Zhu Wang <wangz1@uthscsa.edu>

References

Zhu Wang, Shuangge Ma and Ching-Yun Wang (2015) *Variable selection for zero-inflated and overdispersed data with application to health care demand in Germany*, *Biometrical Journal*. 57(5):867-84.

See Also

[zipath](#)

Examples

```
data("bioChemists", package = "pscl")
fm_zinb <- zipath(art ~ . | ., data = bioChemists, family = "negbin", nlambda=10, maxit.em=1)
res <- se(fm_zinb, which=which.min(fm_zinb$bic))
```

stan

standardize variables

Description

Standardize variables. For each column, return mean 0 and mean value of sum of squares = 1.

Usage

```
stan(x, weights)
```

Arguments

| | |
|---------|--|
| x | numeric variables, can be a matrix or vector |
| weights | numeric positive vector of weights |

Value

A list with the following items.

| | |
|-------|---|
| x | standardized variables with each column: mean value 0 and mean value of sum of squares = 1. |
| meanx | a vector of means for each column in the original x |
| normx | a vector of scales for each column in the original x |

Author(s)

Zhu Wang <wangz1@uthscsa.edu>

summary.glmregNB *Summary Method Function for Objects of Class 'glmregNB'*

Description

Summary results of fitted penalized negative binomial regression model

Usage

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

Arguments

| | |
|--------|--|
| object | fitted model object of class glmregNB. |
| ... | arguments passed to or from other methods. |

Details

This function is a method for the generic function `summary()` for class "glmregNB". It can be invoked by calling `summary(x)` for an object `x` of the appropriate class, or directly by calling `summary.glmregNB(x)` regardless of the class of the object.

Value

Summary of fitted penalized negative binomial model

Author(s)

Zhu Wang <wangz1@uthscsa.edu>

References

Zhu Wang, Shuangge Ma, Michael Zappitelli, Chirag Parikh, Ching-Yun Wang and Prasad Devarajan (2014) *Penalized Count Data Regression with Application to Hospital Stay after Pediatric Cardiac Surgery*, *Statistical Methods in Medical Research*. 2014 Apr 17. [Epub ahead of print]

See Also

[summary.glm.nb](#)

| | |
|---------------|--|
| tuning.zipath | <i>find optimal path for penalized zero-inflated model</i> |
|---------------|--|

Description

Fit penalized zero-inflated models, generate multiple paths with varying penalty parameters, therefore determine optimal path with respect to a particular penalty parameter

Usage

```
tuning.zipath(formula, data, weights, subset, na.action, offset, standardize=TRUE,
family = c("poisson", "negbin", "geometric"), penalty = c("enet", "mnet", "snet"),
lambdaCountRatio = .0001, lambdaZeroRatio = c(.1, .01, .001),
maxit.theta=1, gamma.count=3, gamma.zero=3, ...)
```

Arguments

| | |
|-----------------------------------|---|
| formula | symbolic description of the model, see details. |
| data | argument controlling formula processing via model.frame . |
| weights | optional numeric vector of weights. If standardize=TRUE, weights are renormalized to weights/sum(weights). If standardize=FALSE, weights are kept as original input |
| subset | subset of data |
| na.action | how to deal with missing data |
| offset | Not implemented yet |
| standardize | logical value, should variables be standardized? |
| family | family to fit |
| penalty | penalty considered as one of enet, mnet, snet. |
| lambdaCountRatio, lambdaZeroRatio | Smallest value for lambda.count and lambda.zero, respectively, as a fraction of lambda.max, the (data derived) entry value (i.e. the smallest value for which all coefficients are zero except the intercepts). This lambda.max can be a surrogate value for penalty="mnet" or "snet" |
| maxit.theta | For family="negbin", the maximum iteration allowed for estimating scale parameter theta. Note, the default value 1 is for computing speed purposes, and is typically too small and less desirable in real data analysis |
| gamma.count | The tuning parameter of the snet or mnet penalty for the count part of model. |
| gamma.zero | The tuning parameter of the snet or mnet penalty for the zero part of model. |
| ... | Other arguments passing to zipath |

Details

From the default lambdaZeroRatio = c(.1, .01, .001) values, find optimal lambdaZeroRatio for penalized zero-inflated Poisson, negative binomial and geometric model

Value

An object of class zipath with the optimal lambdaZeroRatio

Author(s)

Zhu Wang <wangz1@uthscsa.edu>

References

Zhu Wang, Shuangge Ma, Michael Zappitelli, Chirag Parikh, Ching-Yun Wang and Prasad Devarajan (2014) *Penalized Count Data Regression with Application to Hospital Stay after Pediatric Cardiac Surgery*, *Statistical Methods in Medical Research*. 2014 Apr 17. [Epub ahead of print]

Zhu Wang, Shuangge Ma, Ching-Yun Wang, Michael Zappitelli, Prasad Devarajan and Chirag R. Parikh (2014) *EM for Regularized Zero Inflated Regression Models with Applications to Postoperative Morbidity after Cardiac Surgery in Children*, *Statistics in Medicine*. 33(29):5192-208.

Zhu Wang, Shuangge Ma and Ching-Yun Wang (2015) *Variable selection for zero-inflated and overdispersed data with application to health care demand in Germany*, *Biometrical Journal*. 57(5):867-84.

See Also

[zipath](#)

Examples

```
## Not run:
## data
data("bioChemists", package = "pscl")

## inflation with regressors
## ("art ~ . | ." is "art ~ fem + mar + kid5 + phd + ment | fem + mar + kid5 + phd + ment")
fm_zip2 <- tuning.zipath(art ~ . | ., data = bioChemists, nlambd=10)
summary(fm_zip2)
fm_zinb2 <- tuning.zipath(art ~ . | ., data = bioChemists, family = "negbin", nlambd=10)
summary(fm_zinb2)

## End(Not run)
```

zipath

*Fit zero-inflated count data linear model with lasso (or elastic net),
snet or mnet regularization*

Description

Fit zero-inflated regression models for count data via penalized maximum likelihood.

Usage

```
## S3 method for class 'formula'
zipath(formula, data, weights, offset=NULL, contrasts=NULL, ... )
## S3 method for class 'matrix'
zipath(X, Z, Y, weights, offsetx=NULL, offsetz=NULL, ...)
## Default S3 method:
zipath(X, ...)
```

Arguments

| | |
|------------------|---|
| formula | symbolic description of the model, see details. |
| data | argument controlling formula processing via <code>model.frame</code> . |
| weights | optional numeric vector of weights. |
| offset | optional numeric vector with an a priori known component to be included in the linear predictor of the count model or zero model. See below for an example. |
| contrasts | a list with elements "count" and "zero" containing the contrasts corresponding to levels from the respective models |
| X | predictor matrix of the count model |
| Z | predictor matrix of the zero model |
| Y | response variable |
| offsetx, offsetz | optional numeric vector with an a priori known component to be included in the linear predictor of the count model (offsetx) or zero model (offsetz). |
| ... | Other arguments which can be passed to <code>glmreg</code> or <code>glmregNB</code> |

Value

An object of class "zipath", i.e., a list with components including

| | |
|---------------|---|
| coefficients | a list with elements "count" and "zero" containing the coefficients from the respective models, |
| residuals | a vector of raw residuals (observed - fitted), |
| fitted.values | a vector of fitted means, |
| weights | the case weights used, |
| terms | a list with elements "count", "zero" and "full" containing the terms objects for the respective models, |
| theta | estimate of the additional θ parameter of the negative binomial model (if a negative binomial regression is used), |
| loglik | log-likelihood of the fitted model, |
| family | character string describing the count distribution used, |
| link | character string describing the link of the zero-inflation model, |
| linkinv | the inverse link function corresponding to link, |

| | |
|-----------|--|
| converged | logical value, TRUE indicating successful convergence of zipath, FALSE indicating otherwise |
| call | the original function call |
| formula | the original formula |
| levels | levels of the categorical regressors |
| contrasts | a list with elements "count" and "zero" containing the contrasts corresponding to levels from the respective models, |
| model | the full model frame (if model = TRUE), |
| y | the response count vector (if y = TRUE), |
| x | a list with elements "count" and "zero" containing the model matrices from the respective models (if x = TRUE), |

Author(s)

Zhu Wang <wangz1@uthscsa.edu>

References

Zhu Wang, Shuangge Ma, Michael Zappitelli, Chirag Parikh, Ching-Yun Wang and Prasad Devarajan (2014) *Penalized Count Data Regression with Application to Hospital Stay after Pediatric Cardiac Surgery*, *Statistical Methods in Medical Research*. 2014 Apr 17. [Epub ahead of print]

Zhu Wang, Shuangge Ma, Ching-Yun Wang, Michael Zappitelli, Prasad Devarajan and Chirag R. Parikh (2014) *EM for Regularized Zero Inflated Regression Models with Applications to Postoperative Morbidity after Cardiac Surgery in Children*, *Statistics in Medicine*. 33(29):5192-208.

Zhu Wang, Shuangge Ma and Ching-Yun Wang (2015) *Variable selection for zero-inflated and overdispersed data with application to health care demand in Germany*, *Biometrical Journal*. 57(5):867-84.

See Also

[zipath_fit](#), [glmreg](#), [glmregNB](#)

Examples

```
## data
data("bioChemists", package = "pscl")
## with simple inflation (no regressors for zero component)
fm_zip <- zipath(art ~ 1 | ., data = bioChemists, nlambd=10)
summary(fm_zip)
fm_zip <- zipath(art ~ . | 1, data = bioChemists, nlambd=10)
summary(fm_zip)
## Not run:
fm_zip <- zipath(art ~ . | 1, data = bioChemists, nlambd=10)
summary(fm_zip)
fm_zinb <- zipath(art ~ . | 1, data = bioChemists, family = "negbin", nlambd=10)
summary(fm_zinb)
## inflation with regressors
## ("art ~ . | ." is "art ~ fem + mar + kid5 + phd + ment | fem + mar + kid5 + phd + ment")
```

```

fm_zip2 <- zipath(art ~ . | ., data = bioChemists, nlambda=10)
summary(fm_zip2)
fm_zinb2 <- zipath(art ~ . | ., data = bioChemists, family = "negbin", nlambda=10)
summary(fm_zinb2)
### non-penalized regression, compare with zeroinfl
fm_zinb3 <- zipath(art ~ . | ., data = bioChemists, family = "negbin",
lambda.count=0, lambda.zero=0, reltol=1e-12)
summary(fm_zinb3)
library("pscl")
fm_zinb4 <- zeroinfl(art ~ . | ., data = bioChemists, dist = "negbin")
summary(fm_zinb4)
### offset
exposure <- rep(0.5, dim(bioChemists)[1])
fm_zinb <- zipath(art ~ . +offset(log(exposure)) | ., data = bioChemists,
family = "poisson", nlambda=10)
coef <- coef(fm_zinb)
### offset can't be specified in predict function as it has been contained
pred <- predict(fm_zinb)
## without inflation
## ("art ~ ." is "art ~ fem + mar + kid5 + phd + ment")
fm_pois <- glmreg(art ~ ., data = bioChemists, family = "poisson")
coef <- coef(fm_pois)
fm_nb <- glmregNB(art ~ ., data = bioChemists)
coef <- coef(fm_nb)
### high-dimensional
bioChemists <- cbind(matrix(rnorm(915*100), nrow=915), bioChemists)
fm_zinb <- zipath(art ~ . | ., data = bioChemists, family = "negbin", nlambda=10)

## End(Not run)

```

zipath_fit

*Internal function to fit zero-inflated count data linear model with lasso
(or elastic net), snet or mnet regularization*

Description

Fit zero-inflated regression models for count data via penalized maximum likelihood.

Usage

```

zipath_fit(X, Z, Y, weights, offsetx, offsetz, standardize=TRUE,
intercept = TRUE, family = c("poisson", "negbin", "geometric"),
link = c("logit", "probit", "cloglog", "cauchit", "log"),
penalty = c("enet", "mnet", "snet"), start = NULL, y = TRUE, x = FALSE,
nlambda=100, lambda.count=NULL, lambda.zero=NULL,
type.path=c("active", "nonactive"), penalty.factor.count=NULL,
penalty.factor.zero=NULL, lambda.count.min.ratio=.0001,
lambda.zero.min.ratio=.1, alpha.count=1, alpha.zero=alpha.count, gamma.count=3,
gamma.zero=gamma.count, rescale=FALSE, init.theta=NULL, theta.fixed=FALSE,

```

```
EM=TRUE, maxit.em=200, convtype=c("count", "both"), maxit= 1000,
maxit.theta =10, reltol = 1e-5, thresh=1e-6, eps.bino=1e-5, shortlist=FALSE,
trace=FALSE, ...)
```

Arguments

| | |
|---|---|
| X | predictor matrix of the count model |
| Z | predictor matrix of the zero model |
| Y | response variable |
| weights | optional numeric vector of weights. |
| offsetx | optional numeric vector with an a priori known component to be included in the linear predictor of the count model. |
| offsetz | optional numeric vector with an a priori known component to be included in the linear predictor of the zero model. |
| intercept | Should intercept(s) be fitted (default=TRUE) or set to zero (FALSE) |
| standardize | Logical flag for x variable standardization, prior to fitting the model sequence. The coefficients are always returned on the original scale. Default is standardize=TRUE. |
| family | character specification of count model family (a log link is always used). |
| link | character specification of link function in the binary zero-inflation model (a binomial family is always used). |
| y, x | logicals. If TRUE the corresponding response and model matrix are returned. |
| penalty | penalty considered as one of enet, mnet, snet. |
| start | starting values for the parameters in the linear predictor. |
| nlambda | number of lambda value, default value is 100. The sequence may be truncated before nlambda is reached if a close to saturated model for the zero component is fitted. |
| lambda.count | A user supplied lambda.count sequence. Typical usage is to have the program compute its own lambda.count and lambda.zero sequence based on nlambda and lambda.min.ratio. |
| lambda.zero | A user supplied lambda.zero sequence. |
| type.path | solution path with default value "active", which is less time computing than "nonactive". If type.path="nonactive", no active set for each element of the lambda sequence and cycle through all the predictor variables. If type.path="active", then cycle through only the active set, then cycle through all the variables for the same penalty parameter. See details below. |
| penalty.factor.count, penalty.factor.zero | These are numeric vectors with the same length as predictor variables. that multiply lambda.count, lambda.zero, respectively, to allow differential shrinkage of coefficients. Can be 0 for some variables, which implies no shrinkage, and that variable is always included in the model. Default is same shrinkage for all variables. |

| | |
|--|--|
| <code>lambda.count.min.ratio</code> , <code>lambda.zero.min.ratio</code> | Smallest value for <code>lambda.count</code> and <code>lambda.zero</code> , respectively, as a fraction of <code>lambda.max</code> , the (data derived) entry value (i.e. the smallest value for which all coefficients are zero except the intercepts). Note, there is a closed formula for <code>lambda.max</code> for <code>penalty="enet"</code> . If <code>rescale=TRUE</code> , <code>lambda.max</code> is the same for <code>penalty="mnet"</code> or <code>"snet"</code> . Otherwise, some modifications are required. In the current implementation, for small gamma value, the square root of the computed <code>lambda.zero[1]</code> is used when <code>penalty="mnet"</code> or <code>"snet"</code> . |
| <code>alpha.count</code> | The elastic net mixing parameter for the count part of model. |
| <code>alpha.zero</code> | The elastic net mixing parameter for the zero part of model. |
| <code>gamma.count</code> | The tuning parameter of the <code>snet</code> or <code>mnet</code> penalty for the count part of model. |
| <code>gamma.zero</code> | The tuning parameter of the <code>snet</code> or <code>mnet</code> penalty for the zero part of model. |
| <code>rescale</code> | logical value, if <code>TRUE</code> , adaptive rescaling |
| <code>init.theta</code> | The initial value of <code>theta</code> for <code>family="negbin"</code> . This is set to <code>NULL</code> since version 0.3-24. |
| <code>theta.fixed</code> | Logical value only used for <code>family="negbin"</code> . If <code>TRUE</code> and <code>init.theta</code> is provided with a numeric value > 0 , then <code>init.theta</code> is not updated. If <code>theta.fixed=FALSE</code> , then <code>init.theta</code> will be updated. In this case, if <code>init.theta=NULL</code> , its initial value is computed with intercept-only zero-inflated <code>negbin</code> model. |
| <code>EM</code> | Using EM algorithm. Not implemented otherwise |
| <code>convtype</code> | convergency type, default is for count component only for speedy computation |
| <code>maxit.em</code> | Maximum number of EM algorithm |
| <code>maxit</code> | Maximum number of coordinate descent algorithm |
| <code>maxit.theta</code> | Maximum number of iterations for estimating <code>theta</code> scaling parameter if <code>family="negbin"</code> . Default value <code>maxit.theta</code> may be increased, yet may slow the algorithm |
| <code>eps.bino</code> | a lower bound of probabilities to be claimed as zero, for computing weights and related values when <code>family="binomial"</code> . |
| <code>reltol</code> | Convergence criteria, default value $1e-5$ may be reduced to make more accurate yet slow |
| <code>thresh</code> | Convergence threshold for coordinate descent. Defaults value is $1e-6$. |
| <code>shortlist</code> | logical value, if <code>TRUE</code> , limited results return |
| <code>trace</code> | If <code>TRUE</code> , progress of algorithm is reported |
| <code>...</code> | Other arguments which can be passed to <code>glmreg</code> or <code>glmregNB</code> |

Details

The algorithm fits penalized zero-inflated count data regression models using the coordinate descent algorithm within the EM algorithm. The returned fitted model object is of class `"zipath"` and is similar to fitted `"glm"` and `"zeroinfl"` objects. For elements such as `"coefficients"` a list is returned with elements for the zero and count component, respectively.

If `type.path="active"`, the algorithm iterates for a pair (`lambda_count`, `lambda_zero`) in a loop:
 Step 1: For initial coefficients `start_count` of the count model and `start_zero` of the zero model, the

EM algorithm is iterated until convergence for the active set with non-zero coefficients determined from `start_count` and `start_zero`, respectively.

Step 2: EM is iterated for all the predict variables once.

Step 3: If active set obtained from Step 2 is the same as in Step 1, stop; otherwise, repeat Step 1 and Step 2.

If `type.path="nonactive"`, the EM algorithm iterates for a pair (`lambda_count`, `lambda_zero`) with all the predict variables until convergence.

A set of standard extractor functions for fitted model objects is available for objects of class "zipath", including methods to the generic functions `print`, `coef`, `logLik`, `residuals`, `predict`. See `predict.zipath` for more details on all methods.

The program may terminate with the following message:

```
Error in: while (j <= maxit.em && !converged) { :
Missing value, where TRUE/FALSE is necessary
Calls: zipath
Additionally: Warning:
In glmreg_fit(Znew,probi,weights = weights,standardize = standardize, :
saturated model, exiting ...
Execution halted
```

One possible reason is that the fitted model is too complex for the data. There are two suggestions to overcome the error. One is to reduce the number of variables. Second, find out what lambda values caused the problem and omit them. Try with other lambda values instead.

Value

An object of class "zipath", i.e., a list with components including

| | |
|----------------------------|---|
| <code>coefficients</code> | a list with elements "count" and "zero" containing the coefficients from the respective models, |
| <code>residuals</code> | a vector of raw residuals (observed - fitted), |
| <code>fitted.values</code> | a vector of fitted means, |
| <code>weights</code> | the case weights used, |
| <code>terms</code> | a list with elements "count", "zero" and "full" containing the terms objects for the respective models, |
| <code>theta</code> | estimate of the additional θ parameter of the negative binomial model (if a negative binomial regression is used), |
| <code>loglik</code> | log-likelihood of the fitted model, |
| <code>family</code> | character string describing the count distribution used, |
| <code>link</code> | character string describing the link of the zero-inflation model, |
| <code>linkinv</code> | the inverse link function corresponding to link, |
| <code>converged</code> | logical value, TRUE indicating successful convergence of zipath, FALSE indicating otherwise |
| <code>call</code> | the original function call |
| <code>formula</code> | the original formula |

| | |
|--------|---|
| levels | levels of the categorical regressors |
| model | the full model frame (if model = TRUE), |
| y | the response count vector (if y = TRUE), |
| x | a list with elements "count" and "zero" containing the model matrices from the respective models (if x = TRUE), |

Author(s)

Zhu Wang <wangz1@uthscsa.edu>

References

Zhu Wang, Shuangge Ma, Michael Zappitelli, Chirag Parikh, Ching-Yun Wang and Prasad Devarajan (2014) *Penalized Count Data Regression with Application to Hospital Stay after Pediatric Cardiac Surgery*, *Statistical Methods in Medical Research*. 2014 Apr 17. [Epub ahead of print]

Zhu Wang, Shuangge Ma, Ching-Yun Wang, Michael Zappitelli, Prasad Devarajan and Chirag R. Parikh (2014) *EM for Regularized Zero Inflated Regression Models with Applications to Postoperative Morbidity after Cardiac Surgery in Children*, *Statistics in Medicine*. 33(29):5192-208.

Zhu Wang, Shuangge Ma and Ching-Yun Wang (2015) *Variable selection for zero-inflated and overdispersed data with application to health care demand in Germany*, *Biometrical Journal*. 57(5):867-84.

See Also

[zipath](#), [glmreg](#), [glmregNB](#)

Index

*Topic **methods**

methods, 30

*Topic **models**

be.zeroinfl, 2

cv.glmreg, 5

cv.glmreg_fit, 9

cv.glmregNB, 7

cv.nclreg, 11

cv.nclreg_fit, 13

cv.zipath, 14

cv.zipath_fit, 17

glmreg, 20

glmreg_fit, 25

glmregNB, 22

ncl, 31

ncl_fit, 38

nclreg, 33

nclreg_fit, 35

plot.glmreg, 39

predict.glmreg, 40

pval.zipath, 43

rzi, 45

summary.glmregNB, 49

tuning.zipath, 50

*Topic **regression**

be.zeroinfl, 2

breadReg, 3

cv.glmreg, 5

cv.glmreg_fit, 9

cv.glmregNB, 7

cv.nclreg, 11

cv.nclreg_fit, 13

cv.zipath, 14

cv.zipath_fit, 17

estfunReg, 19

glmreg, 20

glmreg_fit, 25

glmregNB, 22

hessianReg, 28

meatReg, 29

ncl, 31

ncl_fit, 38

nclreg, 33

nclreg_fit, 35

plot.glmreg, 39

predict.glmreg, 40

predict.zipath, 41

pval.zipath, 43

rzi, 45

sandwichReg, 46

se, 47

tuning.zipath, 50

zipath, 51

zipath_fit, 54

AIC.glmreg (methods), 30

AIC.zipath (methods), 30

be.zeroinfl, 2

BIC.glmreg (methods), 30

BIC.zipath (methods), 30

breadReg, 3, 29, 30, 46, 47

coef, 4, 7, 9, 11, 13, 14, 16, 18, 19, 22, 25, 32, 34, 42, 57

coef.cv.glmreg (cv.glmreg), 5

coef.cv.nclreg (cv.nclreg), 11

coef.cv.zipath (cv.zipath), 14

coef.glmreg (predict.glmreg), 40

coef.zipath (predict.zipath), 41

conv2glmreg, 4

conv2zipath, 5

cv.glmreg, 5, 22

cv.glmreg_fit, 9

cv.glmregNB, 7, 25

cv.nclreg, 11, 34

cv.nclreg_fit, 13

cv.zipath, 14

cv.zipath_fit, 17

deviance.glmreg (glmreg), 20
 estfunReg, 19, 29, 30
 fitted, 42
 fitted.zipath (predict.zipath), 41
 glm.nb, 49
 glmreg, 7, 11, 20, 28, 41, 53, 58
 glmreg_fit, 25
 glmregNB, 9, 22, 53, 58
 glmregNegbin (glmregNB), 22
 hessianReg, 28
 logLik, 57
 logLik.glmreg (glmreg), 20
 logLik.zipath (predict.zipath), 41
 meatReg, 4, 29, 29, 46, 47
 methods, 30
 model.frame, 3, 6, 8, 11, 15, 20, 23, 32, 33, 43, 50, 52
 model.matrix.zipath (predict.zipath), 41
 ncl, 31, 39
 ncl_fit, 38
 nclreg, 13, 14, 33, 38
 nclreg_fit, 35
 plot, 7, 9, 11, 13, 14, 16, 19, 22, 25, 34
 plot.cv.glmreg (cv.glmreg), 5
 plot.cv.nclreg (cv.nclreg), 11
 plot.glmreg, 39
 predict, 7, 9, 11, 13, 14, 16, 19, 22, 25, 32, 34, 42, 57
 predict.cv.glmreg (cv.glmreg), 5
 predict.cv.zipath (cv.zipath), 14
 predict.glmreg, 40
 predict.zipath, 41, 57
 predprob.zipath (predict.zipath), 41
 print, 22, 25, 32, 34, 42, 57
 print.summary.glmregNB
 (summary.glmregNB), 49
 print.summary.zipath (predict.zipath), 41
 pval.zipath, 43
 residuals, 42, 57
 residuals.zipath (predict.zipath), 41
 rzi, 45
 sandwichReg, 4, 30, 46
 se, 47
 stan, 48
 summary, 42, 49
 summary.glmregNB, 49
 summary.zipath (predict.zipath), 41
 terms, 4, 19
 terms.zipath (predict.zipath), 41
 tuning.zipath, 50
 zipath, 16, 19, 20, 42, 43, 47, 51, 51, 58
 zipath_fit, 53, 54