

# Package ‘pengls’

January 23, 2022

**Type** Package

**Title** Fit Penalised Generalised Least Squares models

**Version** 1.0.0

**Description** Combine generalised least squares methodology from the nlme package for dealing with autocorrelation with penalised least squares methods from the glmnet package to deal with high dimensionality. This pengls packages glues them together through an iterative loop. The resulting method is applicable to high dimensional datasets that exhibit autocorrelation, such as spatial or temporal data.

**License** GPL-2

**Encoding** UTF-8

**LazyData** true

**RoxygenNote** 7.1.1

**Imports** glmnet, nlme, stats, BiocParallel

**Suggests** knitr,rmarkdown,testthat

**VignetteBuilder** knitr

**Depends** R (>= 4.0)

**biocViews** Transcriptomics, Regression, TimeCourse

**BugReports** <https://github.com/sthawinke/pengls>

**git\_url** <https://git.bioconductor.org/packages/pengls>

**git\_branch** RELEASE\_3\_14

**git\_last\_commit** 1916a95

**git\_last\_commit\_date** 2021-10-26

**Date/Publication** 2022-01-23

**Author** Stijn Hawinkel [cre, aut] (<<https://orcid.org/0000-0002-4501-5180>>)

**Maintainer** Stijn Hawinkel <[stijn.hawinkel@psb.ugent.be](mailto:stijn.hawinkel@psb.ugent.be)>

## R topics documented:

coef.cv.pengls . . . . .	2
coef.pengls . . . . .	3
cv.pengls . . . . .	3
getCorMat . . . . .	5
makeFolds . . . . .	6
pengls . . . . .	6
predict.pengls . . . . .	8
print.cv.pengls . . . . .	9
print.pengls . . . . .	9

<b>Index</b>	<b>11</b>
--------------	-----------

---

coef.cv.pengls	<i>Extract coefficients from a cv.pengls model</i>
----------------	--

---

### Description

Extract coefficients from a cv.pengls model

### Usage

```
## S3 method for class 'cv.pengls'
coef(object, which = "lambda.1se", ...)
```

### Arguments

object	A cv.pengls object
which	a character string, for which lambda should coefficients be returned
...	further arguments, currently ignored

### Value

The vector of coefficients

---

coef.pengls	<i>Extract coefficients from a pengls model</i>
-------------	---

---

**Description**

Extract coefficients from a pengls model

**Usage**

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

**Arguments**

object	A pengls object
...	further arguments, currently ignored

**Value**

The vector of coefficients

---

cv.pengls	<i>Perform cross-validation pengls</i>
-----------	--

---

**Description**

Perform cross-validation pengls

**Usage**

```
cv.pengls(  
  data,  
  glsSt,  
  xNames,  
  outVar,  
  corMat,  
  nfold,  
  foldid,  
  cvType = "blocked",  
  lambdas,  
  transFun = "identity",  
  transFunArgs = list(),  
  ...  
)
```

**Arguments**

data	A data matrix or data frame
glsSt	a covariance structure, as supplied to nlme::gls as "correlation"
xNames	names of the regressors in data
outVar	name of the outcome variable in data
corMat	a starting value for the correlation matrix. Taken to be a diagonal matrix if missing
nfolds	an integer, the number of folds used in cv.glmnet to find lambda
foldid	An optional vector defining the fold
cvType	A character vector defining the type of cross-validation. Either "random" or "blocked", ignored if foldid is provided
lambdas	an optional lambda sequence
transFun	a transformation function to apply to predictions and outcome in the cross-validation
transFunArgs	Additional arguments passed onto transFun
...	passed onto glmnet::glmnet

**Value**

A list with components

lambda	The series of lambdas
cvm	The vector of mean R <sup>2</sup> 's
cvsd	The standard error of R <sup>2</sup> at the maximum
cvOpt	The R <sup>2</sup> according to the 1 standard error rule
coefs	The matrix of coefficients for every lambda value
lambda.min	Lambda value with maximal R <sup>2</sup>
lambda.1se	Smallest lambda value within 1 standard error from the maximum
foldid	The folds
glsSt	The nlme correlation object

**Examples**

```
library(nlme)
library(BiocParallel)
n <- 50 #Sample size
p <- 100 #Number of features
g <- 10 #Size of the grid
#Generate grid
Grid <- expand.grid("x" = seq_len(g), "y" = seq_len(g))
# Sample points from grid without replacement
GridSample <- Grid[sample(nrow(Grid), n, replace = FALSE),]
#Generate outcome and regressors
b <- matrix(rnorm(p*n), n , p)
```

```

a <- rnorm(n, mean = b %*% rbinom(p, size = 1, p = 0.2)) #20% signal
#Compile to a matrix
df <- data.frame("a" = a, "b" = b, GridSample)
# Define the correlation structure (see ?nlme::gls), with initial nugget 0.5 and range 5
corStruct = corGaus(form = ~ x + y, nugget = TRUE, value = c("range" = 5, "nugget" = 0.5))
#Fit the pengls model, for simplicity for a simple lambda
register(MulticoreParam(3)) #Prepare multithreading
penglsFitCV = cv.pengls(data = df, outVar = "a",
xNames = grep(names(df), pattern = "b", value =TRUE),
glsSt = corStruct, nfolds = 5)
penglsFitCV$lambda.1se #Lambda for 1 standard error rule
penglsFitCV$cvOpt #Corresponding R2
coef(penglsFitCV)
penglsFitCV$foldid #The folds used

```

---

getCorMat

*Get the (square root of the inverse of the) correlation matrix*


---

## Description

Get the (square root of the inverse of the) correlation matrix

## Usage

```
getCorMat(data, glsSt, Coef = c(coef(glsSt)), control, outVar)
```

## Arguments

data	The data frame
glsSt	The correlation object for gls
Coef	optional vector of coefficients to glsSt
control	the list of control arguments for gls
outVar	the name of the outcome variable

## Value

A list with components

corMat	The square root of the inverse correlation matrix
Coef	The coefficients of the correlation object

makeFolds *Divide observations into folds*

---

**Description**

Divide observations into folds

**Usage**

```
makeFolds(nfolds, data, cvType, coords)
```

**Arguments**

nfolds	The number of folds
data	the dataset
cvType	a character vector, indicating the type of cross-validation required, either blocked or random
coords	the names of the coordinates in data

**Value**

the vector of folds

**Examples**

```
nfolds <- 10
data <- expand.grid("x" = seq_len(10), "y" = seq_len(10))
randomFolds <- makeFolds(nfolds = nfolds, data, "random", c("x", "y"))
blockedFolds <- makeFolds(nfolds = nfolds, data, "blocked", c("x", "y"))
```

---

pengls *Iterative estimation of penalised generalised least squares*

---

**Description**

Iterative estimation of penalised generalised least squares

**Usage**

```

pengls(
  data,
  glsSt,
  xNames,
  outVar,
  corMat,
  lambda,
  foldid,
  cvType = c("random", "blocked"),
  maxIter = 30,
  tol = 0.05,
  verbose = FALSE,
  optControl = lmeControl(opt = "optim", maxIter = 500, msVerbose = verbose, msMaxIter
    = 500, niterEM = 1000, msMaxEval = 1000),
  nfolds = 10,
  ...
)

```

**Arguments**

<code>data</code>	A data matrix or data frame
<code>glsSt</code>	a covariance structure, as supplied to <code>nlme::gls</code> as "correlation"
<code>xNames</code>	names of the regressors in data
<code>outVar</code>	name of the outcome variable in data
<code>corMat</code>	a starting value for the correlation matrix. Taken to be a diagonal matrix if missing
<code>lambda</code>	The penalty value for <code>glmnet</code> . If missing, the optimal value of vanilla <code>glmnet</code> without autocorrelation component is used
<code>foldid</code>	An optional vector defining the fold
<code>cvType</code>	A character vector defining the type of cross-validation. Either "random" or "blocked", ignored if <code>foldid</code> is provided
<code>maxIter</code>	maximum number of iterations between <code>glmnet</code> and <code>gls</code>
<code>tol</code>	A convergence tolerance
<code>verbose</code>	a boolean, should output be printed?
<code>optControl</code>	control arguments, passed onto <code>nlme::gls</code> ' control argument
<code>nfolds</code>	an integer, the number of folds used in <code>cv.glmnet</code> to find lambda
<code>...</code>	passed onto <code>glmnet::glmnet</code>

**Details**

This function does not provide cross-validation, but rather fits the model for the lambda penalty value provided, or else the optimal lambda value of the vanilla `glmnet`. Cross-validation needs to be implemented by the user; since the data exhibits autocorrelation this may need to be blocked cross-validation or some other dedicated method.

**Value**

A list with components

corMat	The square root of the inverse correlation matrix
Coef	The coefficients of the correlation object

**Examples**

```
### Example 1: spatial data
# Define the dimensions of the data
library(nlme)
n <- 50 #Sample size
p <- 100 #Number of features
g <- 10 #Size of the grid
#Generate grid
Grid <- expand.grid("x" = seq_len(g), "y" = seq_len(g))
# Sample points from grid without replacement
GridSample <- Grid[sample(nrow(Grid), n, replace = FALSE),]
#Generate outcome and regressors
b <- matrix(rnorm(p*n), n , p)
a <- rnorm(n, mean = b %*% rbinom(p, size = 1, p = 0.2)) #20% signal
#Compile to a matrix
df <- data.frame("a" = a, "b" = b, GridSample)
# Define the correlation structure (see ?nlme::gls), with initial nugget 0.5 and range 5
corStruct <- corGaus(form = ~ x + y, nugget = TRUE, value = c("range" = 5, "nugget" = 0.5))
#Fit the pengls model, for simplicity for a simple lambda
penglsFit <- pengls(data = df, outVar = "a", xNames = grep(names(df), pattern = "b", value =TRUE),
glsSt = corStruct, nfolds = 5)

### Example 2: timecourse data
dfTime <- data.frame("a" = a, "b" = b, "t" = seq_len(50))
corStructTime <- corAR1(form = ~ t, value = 0.5)
penglsFit <- pengls(data = dfTime, outVar = "a",
xNames = grep(names(dfTime), pattern = "b", value =TRUE),
glsSt = corStructTime, nfolds = 5)
```

---

predict.pengls

*Make predictions from a pengls model*

---

**Description**

Make predictions from a pengls model

**Usage**

```
## S3 method for class 'pengls'
predict(object, ...)
```



**Arguments**

object            A pengls object  
...               further arguments, currently ignored

**Value**

A vector with predicted values

---

*print.cv.pengls            Print a summary of a cv.pengls model*

---

**Description**

Print a summary of a cv.pengls model

**Usage**

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

**Arguments**

x                 A cv.pengls object  
...               further arguments, currently ignored

**Value**

Prints output to console

---

*print.pengls                Print a summary of a pengls model*

---

**Description**

Print a summary of a pengls model

**Usage**

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

**Arguments**

x                 A pengls object  
...               further arguments, currently ignored

10

*print.pengls*

**Value**

Prints output to console

# Index

`coef.cv.pengls`, 2  
`coef.pengls`, 3  
`cv.pengls`, 3  
  
`getCorMat`, 5  
  
`makeFolds`, 6  
  
`pengls`, 6  
`predict.pengls`, 8  
`print.cv.pengls`, 9  
`print.pengls`, 9