Package: eshrink (via r-universe)

September 9, 2024

Title Shrinkage for Effect Estimation

Version 0.1.2

Author Joshua Keller

Maintainer Joshua Keller <joshua.keller@colostate.edu>

Description Computes shrinkage estimators for regression problems. Selects penalty parameter by minimizing bias and variance in the effect estimate, where bias and variance are estimated from the posterior predictive distribution. See Keller and Rice (2017) <doi:10.1093/aje/kwx225> for more details.

Depends R (>= 3.2.0)

Imports MASS, glmnet

License GPL (>=2)

LazyData true

RoxygenNote 7.1.1

Repository https://kpkeller.r-universe.dev

RemoteUrl https://github.com/kpkeller/eshrink

RemoteRef HEAD

RemoteSha 0577237cb59faf52a28e2f23d579b9081d7ee527

Contents

`eshrink-package` *Shrinkage Estimators for Regression*

Description

Computes shrinkage estimators for regression problems. Selects penalty parameter by minimizing bias and variance in the effect estimate, where bias and variance are estimated from the posterior predictive distribution. See Keller and Rice (2017) <doi:10.1093/aje/kwx225> for more details.

`check_CIbound` *Confidence intervals for 'fLoss' estimators*

Description

Compute confidence intervals by 'inverting the test' to determine if a given value should lie in the confidence region.

Usage

```
check_CIbound(
  testBeta,
  obsEst,
  type = c("ridge", "lasso"),
  postParam,
  lambdaseq,
  X,
  nPost,
  ind = 1,
  Bstar = 100,
  B = 500,
  loss = "fMBV",
  lowerBound = TRUE,
  reproducible = TRUE,
  alpha = 0.025,
  returnDist = FALSE,
  ...
)

invertTest(
  interval,
  type = "ridge",
  lower.interval = interval,
  upper.interval = interval,
  ...,
  tol = 0.005,
  fulldetail = FALSE
)
```

Arguments

testBeta	Candidate value of beta to test.
obsEst	Estimate of beta from the observed data for which a confidence interval is desired
type	String indicating "ridge" or "LASSO".
postParam	List of parameters for the posterior distribution of beta. See samplePosterior for expected names.
lambdaseq	Sequence of penalty values
X	deisgn matrix
nPost	Number of posterior samples to use.
ind	Index of parameter to test. Defaults to 1.
Bstar	Number of estimators to compute for comparison distribution. Larger values improve the precision of the procedure but increase computational cost.
B	Passed to festLASSO
loss	Either "fMBV" or "fMSE".
lowerBound	Logical indicating that the test is for a lower bound
reproducible	Should the simulated datasets be reproducible?
alpha	Percentile of the distribution to compare against. See details.
returnDist	If TRUE, then distribution of estimates generated is returned instead of comparison against alpha
...	In invertTest, these are passed to check_CIbound. In check_CIbound, these arguments are passed to samplePosterior.
interval	Interval to check. Used for both upper and lower bound, if they are not provided
lower.interval, upper.interval	Bounding intervals over which to check for lower and upper endpoints of CI
tol	Passed to uniroot
fulldetail	If TRUE, then output from uniroot is included.

Details

This function is used as part of an 'inverting the test' approach to generate confidence intervals for estimators from festRidge. Bstar datasets are generated from slices of the posterior distribution of the model parameters where beta (or other parameter indicated by ind) is fixed at the value testBeta. For each dataset, beta is estimated via festRidge or festLASSO, and the resulting distribution of estimators is compared against the estimate from the observed data (obsEst).

The values of lambdaseq, X, nPost, and loss are passed to festRidge or festLASSO and typically match the values that were used to compute obsEst.

The computational cost of this function is most affected by the values of nPost and Bstar. Large values of the latter are important for adequately representing the distribution of parameter estimates. In some settings, nPost can be reduced without substantially impacting the results. However, each dataset is likely to be different.

Author(s)

Joshua Keller

See Also

festRidge

---

estRidge *Estimate Coefficients for Ridge Regression*

Description

Computes a vector of regression coefficients for a provided ridge penalty.

Usage

```
estRidge(lambda, X, y, penalize, XtX = crossprod(X))
```

Arguments

lambda	ridge penalty factor
X	design matrix for the regression.
y	outcome vector. Unless X contains an intercept column, this should typically be centered.
penalize	vector giving penalty structure. Values must be in [0, 1]. See Details for more information.
XtX	(optional) cross product of the design matrix. If running simulations or other procedure for identical X, providing a pre-computed value can reduce computational cost.

Details

The input penalize is a vector of ridge penalty factors, such that the penalty for covariate j is lambda*penalize[j]. Although its primary purpose is for indicating which variables to penalize (1) and which to not penalize (0), fractional values between 0 and 1 are accepted. Defaults to c(0, rep(1, p-1)), where p is number of columns in X (this penalizes all coefficients but the first).

The design matrix X is assumed to contain only numeric values, so any factors should be coded according to desired contrast (e.g., via model.matrix)

Author(s)

Joshua Keller

See Also

festRidge, mseRidge

festLASSO *Compute 'Future Loss' Ridge or LASSO Estimates*

Description

Computes a ridge or LASSO estimate for a given regression problem, with penalty parameter chosen to minimize bias and variance.

Usage

```
festLASSO(
  X,
  y,
  loss = c("fMSE", "fMBV", "both"),
  ind = 1,
  lseq,
  B = 500,
  penalize,
  rescale.lambda = TRUE,
  scale = FALSE,
  returnMSE = FALSE,
  postsamp,
  returnPS = FALSE,
  nPost = 1000,
  se.version = c("varExp", "full", "none"),
  ...
)

festRidge(
  X,
  y,
  loss = c("fMSE", "fMBV", "both"),
  ind = 1,
  lseq,
  penalize,
  scale = FALSE,
  returnMSE = FALSE,
  postsamp,
  returnPS = FALSE,
  nPost = 1000,
  se.version = c("varExp", "full", "none"),
  XtXlamIinv = NULL,
  ...
)
```

Arguments

X	Design matrix for the regression. Assumed to contain only numeric values, so any factors should be coded according to desired contrast (e.g., via model.matrix)
y	Outcome vector. Unless X contains an intercept column, this should typically be centered.
loss	Loss function for choosing the penalty parameter. See details.
ind	Vector of integers or logicals indicating which coefficients the loss is to be computed on.
lseq	Sequence of penalty values to consider.
B	Number of future datasets to simulate for each point in posterior sample.
penalize	See estRidge
rescale.lambda	If TRUE, then lambda is rescaled to account for the default re-scaling done by glmnet. Can also be a scalar scaling factor.
scale	Logical indicating whether the design matrix X be scaled. See details.
returnMSE	Logical indicating whether mse object should be returned.
postsamp	List containing posterior sample (from samplePosterior). If missing, then a posterior sample is drawn. Currently checks on the provided postsamp are limited, so use with caution. Designed to facilitate simulations or other scenarios where it may be pre-computed.
returnPS	logical indicating whether or not the full posterior sample should be included in output.
nPost	Size of posterior sample to compute
se.version	String indicating which version of standard errors to use. See vcovfestRidge.
...	Other arguments passed to samplePosterior
XtXlamIinv	explicit value of (X'X + diag(penalty))^-1. Useful for simulations to save computation.

Details

The value of the ridge or LASSO penalty is selected by minimizing the posterior expectation of the loss function, which is chosen by the argument loss. Possible options are fMBV, which uses the loss function $fMBV = max(Bias(\beta)^2, Var(\beta))$ and fMSE, which uses the loss function $fMSE = Bias(\beta)^2 + Var(\beta)$.

To balance the influence of covariates, it is recommended that the design matrix be standardized. This can be done by the user or via the argument scale. If scale=TRUE, then coefficient and standard error estimates are back-transformed.

Use the XtXlamIinv argument with caution. No checks are done on the provided value. Note that lseq is re-ordered to be decreasing, and provided values of XtXlamIinv must account for this.

See Also

mseRidge, vcovfestRidge, simLASSO, check_CIbound

mseRidge *Compute MSE, Bias, and Variance for Ridge Estimator*

Description

Computes the analytic mean-squared error (MSE), bias, and variance for ridge regression estimators given different values of the true `beta` and `sigma2` parameters.

Usage

```
mseRidge(lambda, XtX, beta, sigma2, penalize, ind = 1, XtXlamIinv = NULL)

biasRidge(lambda, XtX, beta, penalize, ind = 1, XtXlamIinv = NULL)

varRidge(lambda, XtX, sigma2 = 1, penalize, ind = 1, XtXlamIinv = NULL)
```

Arguments

`lambda`	penalty parameter value. For `biasRidge` and `varRidge`, this should be a single value. For `mseRidge`, either a single value of a list of values.
`XtX`	Cross product of design matrix. Not needed if `XtXlamIinv` is provided.
`beta`	True parameter values. Either a vector of length `p` or a `p` x `d` matrix.
`sigma2`	Value of the variance parameter
`penalize`	Vector of penalty factors. See `estRidge` for more information.
`ind`	Numerical or logical vector indicating which elements of the bias vector and variance matrix should be returned. Defaults to the first element.
`XtXlamIinv`	Optional explicit value of `(XtX + diag(lambda*penalize))^(-1)`.

Details

The computations assume that all covariates are correctly included in the mean model and bias is due only to penalization. The bias is given by:

$-(X'X + \Lambda)^{-1}\Lambda\beta$

where $\Lambda = diag(\lambda * penalize)$. The variance is given by:

$\sigma^2(X'X + \Lambda)^{-1}X'X(X'X + \Lambda)^{-1}$

If `beta` is provided as a matrix, this will treat each column of `beta` as a different true parameter vector and return a matrix of bias values (or a vector, if `ind` has length 1).

Providing a pre-computed value of `XtXlamIinv` can reduce the computational cost in simulations. However, the user is responsible for assuring that the value of `lambda` provided matches the value used to compute `XtXlamIinv`.

Value

For `mseRidge`, a list containing the variance, bias, and MSE. For `biasRidge` and `varRidge`, a matrix is returned.

Author(s)

Joshua Keller

`samplePosterior` *Posterior Sample for Bayesian Linear Regression*

Description

Draws a sample from the posterior distribution of parameters from a Bayesian Linear regression model.

Usage

```
samplePosterior(
  X,
  y,
  n,
  a0 = 1,
  b0 = 5e-05,
  v0inv = 1/1000,
  mu0 = 0,
  returnParams = TRUE,
  intercept = FALSE
)
```

Arguments

`X`	Design matrix of size `n` by `p`.
`y`	Outcome variable
`n`	Size of posterior sample to be computed. A value of 0 is accepted.
`a0`, `b0`	Hyperparameters (shape, rate) for inverse gamma distribution of the error variance.
`v0inv`	Prior precision for the error term. Either a single value to be repeated in a diagonal precision matrix, or a `p` by `p` matrix.
`mu0`	Prior mean. Either a single value that will be repeated, or a vector of length `p`. Defaults to zero vector.
`returnParams`	Logical indicating whether the parameters of the posterior distribution are returned.
`intercept`	Logical indicating whether an intercept is included in the model. If `FALSE`, then `y` is centered.

Details

This function draws a sample from the posterior distributions of the coefficient parameter (β) and error variance parameter (σ^2) from a Bayesian linear regression model. The variance parameter is assumed to follow an inverse-gamma distribution. Conditional on the error variance parameter and a specified precision matrix, the coefficient parameters (β) are multivariate normal.

Value

A list containing the following elements:

sigma2	A vector containing the posterior sample of σ^2 values.
beta	Matrix containing the posterior sample of β values.
postMu	Vector containing the posterior mean (if returnParams =TRUE).
postV	Matrix giving the posterior mean (if returnParams =TRUE).
an, bn	Posterior hyperparameters for the inverse gamma distribution of the error variance (if returnParams =TRUE).

Author(s)

Joshua Keller

Examples

```
x <- rnorm(40, mean=2, sd=2)
y <- x + rnorm(40, sd=1)
samplePosterior(X=x, y=y, n=10)
samplePosterior(X=cbind(1, x), y=y, n=10, intercept=TRUE)
samplePosterior(X=cbind(1, x), y=y, n=0, mu=c(3, 3), intercept=TRUE)
```

---

simLASSO *Compute Lasso Estimator for simulated Data*

Description

Simulates data from a regression model and computes the lasso estimate for this data.

Usage

```
simLASSO(lambda, X, beta, sigma, penalize, rescale.lambda = TRUE, ind = 1)
```

Arguments

lambda	Penalty factor to be applied
X	Design matrix of regression problem
beta	true value of parameter vector to simulate from
sigma	true value of square root of variance parameter for simulating.
penalize	Vector giving penalty structure. Supplied to glmnet as 'penalty.factor'. By default, all coefficients except first are penalized.
rescale.lambda	Should lambda be rescaled to account for the default re-scaling done by glmnet?
ind	Index of coefficient to be returned. Value of 0 implies all coefficients (i.e. the full parameter vector estimate)

Details

Simulates data from a regression model with true coefficient parameter `beta` and normal errors with standard deviation `sigma`. Computes the LASSO estimate for the coefficient vector using the `glmnet` function from the package of the same name.

Author(s)

Joshua Keller

`vcovfestRidge` *Standard Error Estimate*

Description

Computes an estimate of the variance-covariance matrix for an 'fLoss' ridge estimator

Usage

```
vcovfestRidge(
  fLoss,
  lambda,
  XtX,
  postBeta,
  postSigma2,
  penalize,
  ind = 1,
  version = c("varExp", "full")
)
```

Arguments

`fLoss`	A matrix of loss function values to be minimized. Assumed structure is the columns correspond to different values of penalty parameter and rows correspond to points in a posterior sample of (beta, sigma).
`lambda`	The sequence of penalty parameter values corresponding to the columns of `fLoss`.
`XtX`	Cross product of the design matrix.
`postBeta`	Matrix containing the posterior sample of beta values. Assumed to be n-by-p, where n is number of samples (and equal to number of rows in fLoss) and p is number of regression parameters in model.
`postSigma2`	Vector containing the posterior sample of variance parameters. Should have same length as postBeta.
`penalize`	Vector indicating which variables are penalized in the regression model. See details for `estRidge` for further details.

ind	Numerical or logical vector indicating which elements of the variance matrix should be returned. Defaults to the (1,1) element
version	Character string indicating which version of standard error to compute. 'varExp' or 'full', corresponding to the variance of the conditional mean of the estimator or that plus the expected value of the conditional variance. In practice, the latter is often too large.

Details

Computes a standard error estimate for an 'fLoss' estimator, where 'fLoss' is typically fMSE or fMBV. Approximates the variance of the estimator using the the variance conditional on the observed data (i.e. using the posterior distribution of parameters). Currently, two different versions are available.

Author(s)

Joshua Keller

See Also

festRidge, samplePosterior

Index