Package ‘cernn’

April 15, 2015

Title Covariance Estimation Regularized by Nuclear Norm Penalties

Description An implementation of the covariance estimation method proposed in Chi and Lange (2014), ``Stable estimation of a covariance matrix guided by nuclear norm penalties,''
Computational Statistics and Data Analysis 80:117-128.

Version 0.1

Maintainer Eric C. Chi <ecchi1105@gmail.com>

Author Eric C. Chi <ecchi1105@gmail.com>

Depends R (>= 3.1.3)

License MIT + file LICENSE

LazyData true

NeedsCompilation no

Repository CRAN

Date/Publication 2015-04-15 12:38:20

R topics documented:

`cernn` *Compute the regularization path for Covariance Estimate Regularized by Nuclear Norms (CERNN)*

Description

cernn performs stable covariance estimation over a grid of regularization parameters.

Usage

```
cernn(X, lambda, alpha)
```

Arguments

X	The data matrix whose rows are observations and columns are covariates.
lambda	vector of regularization parameters controling amount of shrinkage towards the target.
alpha	Parameter that controls mixture between the trace and inverse trace penalties.

References

Eric C. Chi and Kenneth Lange, Stable estimation of a covariance matrix guided by nuclear norm penalties, Computational Statistics and Data Analysis, 80:117-128, 2014.

See Also

get_alpha, shrink_eigen, select_lambda

Examples

```
n <- 10
p <- 5
set.seed(12345)
X <- matrix(rnorm(n*p),n,p)
alpha <- get_alpha(X)
lambda <- 10**(seq(-1,4,length.out=100))
sol_path <- cernn(X,lambda,alpha)
df <- t(sol_path$e)

## Plot regularization paths of eigenvalues
matplot(x=log10(lambda),y=df,type='l',ylab='shrunken eigenvalue')
grand_mean <- (norm(scale(X,center=TRUE,scale=FALSE),'f')**2)/(n*p)
abline(h=grand_mean)
```

get_alpha	*Compute alpha parameter for covariance regularization.*

Description

get_alpha computes the alpha parameter that shrinks eigenvalues of the sample covariance to their grand mean.

Usage

```
get_alpha(X)
```

Arguments

X	The data matrix whose rows are observations and columns are covariates.

Examples

```
n <- 10
p <- 5
set.seed(12345)
X <- matrix(rnorm(n*p),n,p)
get_alpha(X)
```

get_lambda_max	*Compute lambda_max parameter for covariance regularization.*

Description

get_lambda_max computes a maximum lambda value that will shrink eigenvalues nearly to the grand mean.

Usage

```
get_lambda_max(d, alpha, n, eps = 0.01)
```

Arguments

d	Vector of sample eigenvalues to shrink. These must be nonnegative.
alpha	Parameter that controls mixture between the trace and inverse trace penalties.
n	The number of observations.
eps	tolerance

Examples

```
n <- 10
p <- 5
set.seed(12345)
X <- matrix(rnorm(n*p),n,p)
d <- svd(X)$d**2
alpha <- get_alpha(X)
get_lambda_max(d,alpha,n)
```

---

loss_entropy *Entropy Loss*

Description

`loss_entropy` computes the entropy loss, which is also known as Stein's loss.

Usage

```
loss_entropy(S, Sinv)
```

Arguments

`S`	Covariance Estimate
`Sinv`	Reference Precision Matrix

Examples

```
set.seed(12345)
p <- 20
d <- sort(abs(rcauchy(p)),decreasing=TRUE)
sigma <- diag(d)
n <- 20
X <- scale(matrix(rnorm(n*p),n,p),center=FALSE,scale=1/sqrt(d))
alpha <- get_alpha(X)
lambda <- 10**(seq(-2,2,length.out=100))
sol_cv <- select_lambda(X,lambda)
loss_entropy(sol_cv$S,solve(sigma))
```

loss_quadratic — *Quadratic Loss*

Description

`loss_quadratic` computes the quadratic loss.

Usage

```
loss_quadratic(S, Sinv)
```

Arguments

`S`	Covariance Estimate
`Sinv`	Reference Precision Matrix

Examples

```
set.seed(12345)
p <- 20
d <- sort(abs(rcauchy(p)),decreasing=TRUE)
sigma <- diag(d)
n <- 20
X <- scale(matrix(rnorm(n*p),n,p),center=FALSE,scale=1/sqrt(d))
alpha <- get_alpha(X)
lambda <- 10**(seq(-2,2,length.out=100))
sol_cv <- select_lambda(X,lambda)
loss_quadratic(sol_cv$S,solve(sigma))
```

select_lambda — *Selection of penalty parameter based on cross-validation*

Description

`select_lambda` selects the best regularization parameter from a grid of values based on minimal predictive negative log-likelihood.

Usage

```
select_lambda(X, lambda, fold = min(nrow(X), 10))
```

Arguments

`X`	n-by-p data matrix
`lambda`	vector of penalties for cross-validation
`fold`	number of folds for cross-validation

Examples

```
n <- 30
p <- 30
set.seed(12345)
X <- matrix(rnorm(n*p),n,p)
alpha <- get_alpha(X)
lambda_max <- get_lambda_max(svd(X)$d**2,alpha,n)
lambda <- 10**(seq(-1,log10(lambda_max),length.out=100))
sol_path <- cernn(X,lambda,alpha)
df <- t(sol_path$e)

## Plot regularization paths of eigenvalues
matplot(x=log10(lambda),y=df,type='l',ylab='shrunken eigenvalue')
grand_mean <- (norm(scale(X,center=TRUE,scale=FALSE),'f')**2)/(n*p)
abline(h=grand_mean)

## Plot selected lambda
abline(v=log10(select_lambda(X,lambda)$lambda))
```

---

shrink_eigen *Nonlinear shrinkage of sample eigenvalues*

Description

`shrink_eigen` shrinks the sample eigenvalues.

Usage

```
shrink_eigen(d, lambda, alpha, n)
```

Arguments

d	Vector of sample eigenvalues to shrink. These must be nonnegative.
lambda	Regularization parameter controling amount of shrinkage towards the target.
alpha	Parameter that controls mixture between the trace and inverse trace penalties.
n	The number of observations.

Value

Vector of shrunken eigenvalues.

Examples

```
set.seed(12345)
nLambda <- 100
lambda <- 10**seq(-2,2,length.out=nLambda)
alpha <- 0.5
n <- 10
```

```
p <- 5
d <- sort(2*runif(p))
e <- shrink_eigen(d,lambda,alpha,n)

## Plot regularization paths of eigenvalues
matplot(x=log10(lambda),y=t(e),type='l',ylab='shrunken eigenvalue')
```

Index