

MATLAB Toolbox **envlp**: Reference Manual

September 10, 2012

Contents	1
1 tools	2
1.1 bootstrapse	2
1.2 bootstrapse_OLS	5
1.3 mfoldcv	7
1.4 modelselectaic	9
1.5 modelselectbic	11
1.6 modelselectlrt	13
1.7 prediction	15
1.8 testcoefficient	18
2 env	20
2.1 aic_env	20
2.2 bic_env	22
2.3 bstrp_env	24
2.4 dF4env	26
2.5 env	27
2.6 F4env	31
2.7 lrt_env	32
2.8 predict_env	34
2.9 testcoefficient_env	37
3 envmean	39
3.1 aic_envmean	39
3.2 bic_envmean	41
3.3 bstrp_envmean	42
3.4 dF4envmean	44
3.5 envmean	45
3.6 F4envmean	48
3.7 lrt_envmean	49
3.8 predict_envmean	51
3.9 testcoefficient_envmean	53
4 envseq	55
4.1 bstrp_envseq	55
4.2 envseq	57
4.3 mfoldcv_envseq	59

5	henv	61
5.1	aic_henv	61
5.2	bic_henv	63
5.3	bstrp_henv	65
5.4	dF4henv	67
5.5	F4henv	68
5.6	henv	69
5.7	lrt_henv	72
5.8	predict_henv	74
5.9	testcoefficient_henv	77
6	ienv	79
6.1	aic_ienv	79
6.2	bic_ienv	81
6.3	bstrp_ienv	82
6.4	dF4ienv	84
6.5	F4ienv	85
6.6	ienv	86
6.7	lrt_ienv	89
6.8	predict_ienv	91
6.9	testcoefficient_ienv	93
7	penv	95
7.1	aic_penv	95
7.2	bic_penv	97
7.3	bstrp_penv	99
7.4	lrt_penv	101
7.5	penv	103
7.6	predict_penv	106
7.7	testcoefficient_penv	109
8	senv	111
8.1	aic_senv	111
8.2	bic_senv	113
8.3	bstrp_senv	115
8.4	dF4senv	117
8.5	F4senv	118
8.6	objfun	119
8.7	predict_senv	120
8.8	senv	122
8.9	testcoefficient_senv	125
9	xenv	127
9.1	aic_xenv	127
9.2	bic_xenv	129
9.3	bstrp_xenv	131
9.4	dF4xenv	133
9.5	F4xenv	134
9.6	lrt_xenv	135
9.7	predict_xenv	137
9.8	testcoefficient_xenv	139
9.9	xenv	141

10 xenvpls	145
10.1 bstrp_xenvpls	145
10.2 mfoldcv_xenvpls	147
10.3 xenvpls	149
11 auxiliary	151
11.1 center	151
11.2 Contr	152
11.3 Expan	153
11.4 fit_OLS	154
11.5 get_envelope	156
11.6 get_Init	157
11.7 get_Init4envmean	158
11.8 get_Init4henv	159
11.9 Kpd	160
11.10 Lmatrix	161
11.11 make_dF	162
11.12 make_F	163
11.13 make_opts	164
11.14 make_parameter	165
11.15 mtest	167

1.1 bootstrapse

Perform bootstrap to estimate actual standard errors for models in the envelope family.

Contents

- Syntax
- Input
- Output
- Description
- Example

Syntax

```
bootse = bootstrapse(X, Y, u, B, modelType)
bootse = bootstrapse(X, Y, u, B, modelType, Opts)
```

Input

X: Predictors. The predictors can be univariate or multivariate, discrete or continuous.

For model type for method 'env', 'envpls', 'henv', 'ienv', 'senv', 'xenv' and 'xenvpls'. X is an n by p matrix, p is the number of predictors.

For model type 'penv', X is A list containing the value of X1 and X2.

- X.X1 (only for 'penv'): Predictors of main interest. An n by p1 matrix, n is the number of observations, and p1 is the number of main predictors. The predictors can be univariate or multivariate, discrete or continuous.
- X.X2 (only for 'penv'): Covariates, or predictors not of main interest. An n by p2 matrix, p2 is the number of covariates.

Y: Multivariate responses. An n by r matrix, r is the number of responses and n is number of observations. The responses must be continuous variables.

u: Dimension of the envelope subspace. The legitimate range of u depends on the model specified.

B: Number of bootstrap samples. A positive integer.

modelType: A string characters indicting the model, choices can be 'env', 'envpls', 'henv', 'ienv', 'penv', 'senv', 'xenv' and 'xenvpls'.

Opts: A list containing the optional input parameters, to control the iterations in `sg_min`. If one or several (even all) fields are not defined, the default settings are used.

- `Opts.maxIter`: Maximum number of iterations. Default value: 300.
- `Opts.ftol`: Tolerance parameter for F. Default value: $1e-10$.
- `Opts.gradtol`: Tolerance parameter for dF. Default value: $1e-7$.
- `Opts.verbose`: Flag for print out the number of bootstrap samples, logical 0 or 1. Default value: 0.

Output

`bootse`: For `'env'`, `'envpls'`, `'henv'`, `'ienv'`, and `'senv'`, an r by p matrix containing the standard errors for elements in β computed by bootstrap. For `'penv'`, an r by p_1 matrix containing the standard errors for β_1 computed by bootstrap. For `'xenv'` and `'xenvpls'`, a p by r matrix containing the standard errors for elements in β computed by bootstrap.

Description

This function computes the bootstrap standard errors for the regression coefficients or for partial envelope model, the main regression coefficients in the specified model by bootstrapping the residuals.

Example

```
load wheatprotein.txt
X = wheatprotein(:, 8);
Y = wheatprotein(:, 1 : 6);
alpha = 0.01;
u = modelselectlrt(X, Y, alpha, 'env');
B = 100;
modelType = 'env';
bootse = bootstrapse(X, Y, u, B, modelType)
```

`bootse =`

```
0.2896
0.4352
0.3189
0.5735
0.2543
0.5840
```

```
modelType = 'envpls';
bootse = bootstrapse(X, Y, u, B, modelType)
```

`bootse =`

```
9.3899
```

```
7.7384
8.3303
9.0875
13.4416
5.1477
```

```
load fiberpaper.dat
Y = fiberpaper(:, 1 : 4);
X.X1 = fiberpaper(:, 7);
X.X2 = fiberpaper(:, 5 : 6);
alpha = 0.01;
u = modelselectlrt(X, Y, alpha, 'penv');
B = 100;
modelType = 'penv';
bootse = bootstrapse(X, Y, u, B, modelType)
```

```
bootse =
```

```
0.0027
0.0012
0.0020
0.0009
```

1.2 bootstrapse_OLS

Compute bootstrap standard error for ordinary least squares.

Contents

- Syntax
- Input
- Output
- Description
- Example

Syntax

```
bootse = bootstrapse_OLS(X, Y, B)
```

Input

X: Predictors, an n by p matrix, p is the number of predictors. The predictors can be univariate or multivariate, discrete or continuous.

Y: Multivariate responses, an n by r matrix, r is the number of responses and n is number of observations. The responses must be continuous variables.

B: Number of bootstrap samples. A positive integer.

Opts: A list containing the optional input parameters. If not defined, the default setting is used.

- Opts.verbose: Flag for print out the number of bootstrap samples, logical 0 or 1. Default value: 0.

Output

bootse: The standard error for elements in β computed by bootstrap. An r by p matrix.

Description

This function computes the bootstrap standard errors for the regression coefficients in ordinary least squares by bootstrapping the residuals.

Example

```
load wheatprotein.txt
X = wheatprotein(:, 8);
Y = wheatprotein(:, 1 : 6);
bootse = bootstrapse_OLS(X, Y, 200)
```

```
bootse =
```

```
10.2168
```

8.3940
9.0503
9.9677
14.5822
5.5874

1.3 mfoldcv

Select the dimension for the envelope family using m-fold cross validation.

Contents

- Syntax
- Input
- Output
- Description
- Example

Syntax

```
u = mfoldcv(X, Y, m, modelType)
u = mfoldcv(X, Y, m, modelType, Opts)
```

Input

X: Predictors. The predictors can be univariate or multivariate, discrete or continuous.

Y: Multivariate responses. An n by r matrix, r is the number of responses and n is number of observations. The responses must be continuous variables.

m: A positive integer that is used to indicate m-fold cross validation.

modelType: A string characters indicting the model, choices can be 'envpls' or 'xenvpls'.

Opts: A list containing the optional input parameters. If one or several (even all) fields are not defined, the default settings are used.

- Opts.verbose: Flag for print out the number of bootstrap samples, logical 0 or 1. Default value: 0.

Output

u: The dimension of the envelope subspace selected by m-fold cross validation.

Description

This function implements m-fold cross validation to select the dimension of the envelope space, based on prediction performance. For each u, the data is partitioned into m parts, each part is in turn used for testing for the prediction performance while the rest m-1 parts are used for training. The dimension is select as the one that minimizes the average prediction errors. If Y is multivariate, the identity inner product is used for computing the prediction errors.

Example

```
load wheatprotein.txt
X = wheatprotein(:, 8);
Y = wheatprotein(:, 1 : 6);
m = 5;
```

```
modelType = 'envpls';  
u = mfoldcv(X, Y, m, modelType)
```

```
u =
```

```
0
```

1.4 modelselectaic

Select the dimension for the envelope family using Akaike information criteria.

Contents

- Syntax
- Input
- Output
- Description
- Example

Syntax

```
u = modelselectaic(X, Y, modelType)
u = modelselectaic(X, Y, modelType, Opts)
```

Input

X: Predictors. The predictors can be univariate or multivariate, discrete or continuous.

For model type for method 'env', 'henv', 'ienv', 'senv', and 'xenv'. X is an n by p matrix, p is the number of predictors.

For model type 'penv', X is A list containing the value of X1 and X2.

- X.X1 (only for 'penv'): Predictors of main interest. An n by p1 matrix, n is the number of observations, and p1 is the number of main predictors. The predictors can be univariate or multivariate, discrete or continuous.
- X.X2 (only for 'penv'): Covariates, or predictors not of main interest. An n by p2 matrix, p2 is the number of covariates.

Y: Multivariate responses. An n by r matrix, r is the number of responses and n is number of observations. The responses must be continuous variables.

modelType: A string characters indicting the model, choices can be 'env', 'henv', 'ienv', 'penv', 'senv' and 'xenv'.

Opts: A list containing the optional input parameters, to control the iterations in sg_min. If one or several (even all) fields are not defined, the default settings are used.

- Opts.maxIter: Maximum number of iterations. Default value: 300.
- Opts.ftol: Tolerance parameter for F. Default value: 1e-10.
- Opts.gradtol: Tolerance parameter for dF. Default value: 1e-7.
- Opts.verbose: Flag for print out dimension selection process, logical 0 or 1. Default value: 0.

Output

u: Dimension of the envelope. An integer between 0 and r.

Description

This function implements the Akaike information criteria (AIC) to select the dimension of the envelope subspace for method 'env', 'henv', 'ienv', 'penv', 'senv', and 'xenv'.

Example

```
load wheatprotein.txt
X = wheatprotein(:, 8);
Y = wheatprotein(:, 1 : 6);
modelType = 'env';
u = modelselectaic(X, Y, modelType)
```

u =

1

```
load fiberpaper.dat
Y = fiberpaper(:, 1 : 4);
X.X1 = fiberpaper(:, 7);
X.X2 = fiberpaper(:, 5 : 6);
modelType = 'penv';
u = modelselectaic(X, Y, modelType)
```

u =

3

1.5 modelselectbic

Select the dimension for the envelope family using Bayesian information criteria.

Contents

- Syntax
- Input
- Output
- Description
- Example

Syntax

```
u = modelselectbic(X, Y, modelType)
u = modelselectbic(X, Y, modelType, Opts)
```

Input

X: Predictors. The predictors can be univariate or multivariate, discrete or continuous.

For model type for method 'env', 'henv', 'ienv', 'senv', and 'xenv'. X is an n by p matrix, p is the number of predictors.

For model type 'penv', X is A list containing the value of X1 and X2.

- X.X1 (only for 'penv'): Predictors of main interest. An n by p1 matrix, n is the number of observations, and p1 is the number of main predictors. The predictors can be univariate or multivariate, discrete or continuous.
- X.X2 (only for 'penv'): Covariates, or predictors not of main interest. An n by p2 matrix, p2 is the number of covariates.

Y: Multivariate responses. An n by r matrix, r is the number of responses and n is number of observations. The responses must be continuous variables.

modelType: A string characters indicting the model, choices can be 'env', 'henv', 'ienv', 'penv', 'senv' and 'xenv'.

Opts: A list containing the optional input parameters, to control the iterations in sg_min. If one or several (even all) fields are not defined, the default settings are used.

- Opts.maxIter: Maximum number of iterations. Default value: 300.
- Opts.ftol: Tolerance parameter for F. Default value: 1e-10.
- Opts.gradtol: Tolerance parameter for dF. Default value: 1e-7.
- Opts.verbose: Flag for print out dimension selection process, logical 0 or 1. Default value: 0.

Output

u: Dimension of the envelope. An integer between 0 and r.

Description

This function implements the Bayesian information criteria (BIC) to select the dimension of the envelope subspace for method 'env', 'henv', 'ienv', 'penv', 'senv', and 'xenv'.

Example

```
load wheatprotein.txt
X = wheatprotein(:, 8);
Y = wheatprotein(:, 1 : 6);
modelType = 'env';
u = modelselectbic(X, Y, modelType)
```

u =

1

```
load fiberpaper.dat
Y = fiberpaper(:, 1 : 4);
X.X1 = fiberpaper(:, 7);
X.X2 = fiberpaper(:, 5 : 6);
modelType = 'penv';
u = modelselectbic(X, Y, modelType)
```

u =

1

1.6 modelselectlrt

Select the dimension for the envelope family using likelihood ratio testing procedure.

Contents

- Syntax
- Input
- Output
- Description
- Example

Syntax

```
u = modelselectlrt(X, Y, alpha, modelType)
u = modelselectlrt(X, Y, alpha, modelType, Opts)
```

Input

X: Predictors. The predictors can be univariate or multivariate, discrete or continuous.

For model type for method 'env', 'henv', 'ienv', and 'xenv'. X is an n by p matrix, p is the number of predictors.

For model type 'penv', X is A list containing the value of X1 and X2.

- X.X1 (only for 'penv'): Predictors of main interest. An n by p1 matrix, n is the number of observations, and p1 is the number of main predictors. The predictors can be univariate or multivariate, discrete or continuous.
- X.X2 (only for 'penv'): Covariates, or predictors not of main interest. An n by p2 matrix, p2 is the number of covariates.

Y: Multivariate responses. An n by r matrix, r is the number of responses and n is number of observations. The responses must be continuous variables.

alpha: Significance level for testing. A real number between 0 and 1, often taken at 0.05 or 0.01.

modelType: A string characters indicting the model, choices can be 'env', 'henv', 'ienv', 'penv' and 'xenv'.

Opts: A list containing the optional input parameters, to control the iterations in sg_min. If one or several (even all) fields are not defined, the default settings are used.

- Opts.maxIter: Maximum number of iterations. Default value: 300.
- Opts.ftol: Tolerance parameter for F. Default value: 1e-10.
- Opts.gradtol: Tolerance parameter for dF. Default value: 1e-7.
- Opts.verbose: Flag for print out dimension selection process, logical 0 or 1. Default value: 0.

Output

u: Dimension of the envelope. An integer between 0 and r.

Description

This function implements the likelihood ratio testing procedure to select the dimension of the envelope subspace for method 'env', 'henv', 'ienv', 'penv', and 'xenv'. The likelihood ratio testing procedure does not support 'senv', because the scaled envelope models are not nested with the standard model.

Example

```
load wheatprotein.txt
X = wheatprotein(:, 8);
Y = wheatprotein(:, 1 : 6);
alpha = 0.01;
modelType = 'env';
u = modelselectlrt(X, Y, alpha, modelType)
```

u =

1

```
load fiberpaper.dat
Y = fiberpaper(:, 1 : 4);
X.X1 = fiberpaper(:, 7);
X.X2 = fiberpaper(:, 5 : 6);
alpha = 0.01;
modelType = 'penv';
u = modelselectlrt(X, Y, alpha, modelType)
```

u =

1

1.7 prediction

Perform estimation or prediction for models in the envelope family.

Contents

- Syntax
- Input
- Output
- Description
- Example

Syntax

```
PredictOutput = prediction(ModelOutput, Xnew, infType, modelType)
```

Input

ModelOutput: A list containing the model outputs from fitting the models.

Xnew: The value of X with which to estimate or predict Y.

For 'env', 'henv', 'ienv', 'senv' and 'xenv', it is a p by 1 vector.

For 'penv', it is a list containing the value of X1 and X2.

* Xnew.X1 (only for 'penv'): A p1 by 1 vector containing the value of X1.

* Xnew.X2 (only for 'penv'): A p2 by 1 vector containing the value of X2.

infType: A string of characters indicting the inference type, the choices can be 'estimation' or 'prediction'.

modelType: A string characters indicting the model, choices can be 'env', 'henv', 'ienv', 'penv', 'senv' and 'xenv'.

Output

PredictOutput: A list containing the results of the inference.

- PredictOutput.value: The fitted value or the prediction value evaluated at Xnew. An r by 1 vector.
- PredictOutput.covMatrix: The covariance matrix of PredictOutput.value. An r by r matrix.
- PredictOutput.SE: The standard error of elements in PredictOutput.value. An r by 1 vector.

Description

This function evaluates the user-specified model, could be 'env', 'henv', 'ienv', 'penv', 'senv' or 'xenv', at new value Xnew. It can perform estimation: find the fitted value when $X = X_{\text{new}}$, or prediction: predict Y when $X = X_{\text{new}}$. The covariance matrix and the standard errors are also provided.

Example

```

load wheatprotein.txt
X = wheatprotein(:, 8);
Y = wheatprotein(:, 1 : 6);
modelType = 'env';
u = modelselectbic(X, Y, modelType);
ModelOutput = env(X, Y, u);
Xnew = X(2, :)';
PredictOutput = prediction(ModelOutput, Xnew, 'estimation', modelType)
[PredictOutput.value, Y(2, :)] % Compare the fitted value with
the observed value

```

PredictOutput =

```

    value: [6x1 double]
 covMatrix: [6x6 double]
        SE: [6x1 double]

```

ans =

```

474.7135  458.0000
127.4740  112.0000
251.2044  236.0000
380.8280  368.0000
380.9473  383.0000
-6.3287  -15.0000

```

```

load fiberpaper.dat
Y = fiberpaper(:, 1 : 4);
X.X1 = fiberpaper(:, 7);
X.X2 = fiberpaper(:, 5 : 6);
modelType = 'penv';
u = modelselectbic(X, Y, modelType);
ModelOutput = penv(X, Y, u);
Xnew.X1 = X.X1(1, :)';
Xnew.X2 = X.X2(1, :)';
PredictOutput = prediction(ModelOutput, Xnew, 'estimation', modelType)
PredictOutput.SE

```

PredictOutput =

```

    value: [4x1 double]
 covMatrix: [4x4 double]
        SE: [4x1 double]

```

```
ans =
```

```
1.4680
```

```
0.4234
```

```
0.7145
```

```
0.3161
```

1.8 testcoefficient

This function tests the null hypothesis $L * \beta * R = A$ versus the alternative hypothesis $L * \beta * R \neq A$, where β is estimated under the model in the envelope family.

Contents

- Syntax
- Input
- Output
- Description
- Example

Syntax

```
TestOutput = testcoefficient(ModelOutput, modelType)
TestOutput = testcoefficient(ModelOutput, modelType, TestInput)
```

Input

ModelOutput: A list containing the model outputs from fitting the models.

modelType: A string characters indicting the model, choices can be 'env', 'henv', 'ienv', 'penv', 'senv' and 'xenv'.

TestInput: A list that specifies the null hypothesis, including L, R, and A. If not provided by the user, default values will be used.

- TestInput.L: The matrix multiplied to β on the left. According to different model, it has different size requirement. Default value will be set if the user does not specify.
- TestInput.R: The matrix multiplied to β on the right. According to different model, it has different size requirement. Default value will be set if the user does not specify.
- TestInput.A: The matrix on the right hand side of the equation. Default value will be set if the user does not specify.

Output

TestOutput: A list containing test statistics, degrees of freedom for the reference chi-squared distribution, the p-value, and the covariance matrix of $\text{vec}(L\beta R)$. At the same time, a table is printed out.

- TestOutput.chisqStatistic: The test statistics. A real number.
- TestOutput.df: The degrees of freedom of the reference chi-squared distribution. A positive integer.
- TestOutput.pValue: p-value of the test. A real number in [0, 1].
- TestOutput.covMatrix: The covariance matrix of $\text{vec}(L\beta R)$. A $d1 * d2$ by $d1 * d2$ matrix.

Description

This function tests for hypothesis $H_0 : L\beta R = A$, versus $H_\alpha : L\beta R \neq A$. The β is estimated by a model in the envelope model. If the user does not specify the values for L, R and A, then the test is equivalent to the standard F test on if $\beta = 0$ (for 'env', 'ienv', 'penv', 'senv' and 'xenv'), or if the group main effects are all zeros (for 'henv'). The test statistics used is $\text{vec}(L\beta R - A) \hat{\Sigma}^{-1} \text{vec}(L\beta R - A)^T$, and the reference distribution is chi-squared distribution with degrees of freedom the same as the length of $\text{vec}(A)$.

Example

```
load wheatprotein.txt
X = wheatprotein(:, 8);
Y = wheatprotein(:, 1 : 6);
alpha = 0.01;
u = lrt_env(X, Y, alpha);
ModelOutput = env(X, Y, u);
modelType = 'env';
TestOutout = testcoefficient(ModelOutput, modelType);
```

Test Hypothesis	Chisq Statistic	DF	P-value
L * beta * R = A	116.230	6	0.0000

```
load fiberpaper.dat
Y = fiberpaper(:, 1 : 4);
X.X1 = fiberpaper(:, 7);
X.X2 = fiberpaper(:, 5 : 6);
alpha = 0.01;
u = modelselectlrt(X, Y, alpha, 'penv');
ModelOutput = penv(X, Y, u);
r = size(Y, 2);
p1 = size(X.X1, 2);
TestInput.L = rand(2, r);
TestInput.R = rand(p1, 1);
TestInput.A = zeros(2, 1);
TestOutout = testcoefficient_penv(ModelOutput, TestInput);
```

Test Hypothesis	Chisq Statistic	DF	P-value
L * beta * R = A	12.598	2	0.0018

2.1 aic_env

Select the dimension of the envelope subspace using Akaike information criterion.

Contents

- Syntax
- Input
- Output
- Description
- Example

Syntax

```
u = aic_env(X, Y)
u = aic_env(X, Y, Opts)
```

Input

X: Predictors. An n by p matrix, p is the number of predictors. The predictors can be univariate or multivariate, discrete or continuous.

Y: Multivariate responses. An n by r matrix, r is the number of responses and n is number of observations. The responses must be continuous variables.

Opts: A list containing the optional input parameters, to control the iterations in `sg_min`. If one or several (even all) fields are not defined, the default settings are used.

- `Opts.maxIter`: Maximum number of iterations. Default value: 300.
- `Opts.ftol`: Tolerance parameter for F Default value: $1e-10$.
- `Opts.gradtol`: Tolerance parameter for dF Default value: $1e-7$.
- `Opts.verbose`: Flag for print out dimension selection process, logical 0 or 1. Default value: 0.

Output

u: Dimension of the envelope. An integer between 0 and r .

Description

This function implements the Akaike information criteria (AIC) to select the dimension of the envelope subspace.

Example

```
load wheatprotein.txt
X = wheatprotein(:, 8);
Y = wheatprotein(:, 1 : 6);
u = aic_env(X, Y)
```

u =

1

2.2 bic_env

Select the dimension of the envelope subspace using Bayesian information criterion.

Contents

- Syntax
- Input
- Output
- Description
- Example

Syntax

```
u = bic_env(X, Y)
u = bic_env(X, Y, Opts)
```

Input

X: Predictors. An n by p matrix, p is the number of predictors and n is the number of observations. The predictors can be univariate or multivariate, discrete or continuous.

Y: Multivariate responses. An n by r matrix, r is the number of responses. The responses must be continuous variables.

Opts: A list containing the optional input parameters, to control the iterations in `sg_min`. If one or several (even all) fields are not defined, the default settings are used.

- `Opts.maxIter`: Maximum number of iterations. Default value: 300.
- `Opts.ftol`: Tolerance parameter for F. Default value: $1e-10$.
- `Opts.gradtol`: Tolerance parameter for dF. Default value: $1e-7$.
- `Opts.verbose`: Flag for print out dimension selection process, logical 0 or 1. Default value: 0.

Output

u: Dimension of the envelope. An integer between 0 and r .

Description

This function implements the Bayesian information criteria (BIC) to select the dimension of the envelope subspace.

Example

```
load wheatprotein.txt
X = wheatprotein(:, 8);
Y = wheatprotein(:, 1 : 6);
u = bic_env(X, Y)
```


u =

1

2.3 bstrp_env

Compute bootstrap standard error for the envelope model.

Contents

- Syntax
- Input
- Output
- Description
- Example

Syntax

```
bootse = bstrp_env(X, Y, u, B)
bootse = bstrp_env(X, Y, u, B, Opts)
```

Input

X: Predictors, an n by p matrix, p is the number of predictors. The predictors can be univariate or multivariate, discrete or continuous.

Y: Multivariate responses, an n by r matrix, r is the number of responses and n is number of observations. The responses must be continuous variables.

u: Dimension of the envelope subspace. A positive integer between 0 and r .

B: Number of bootstrap samples. A positive integer.

Opts: A list containing the optional input parameters, to control the iterations in `sg_min`. If one or several (even all) fields are not defined, the default settings are used.

- `Opts.maxIter`: Maximum number of iterations. Default value: 300.
- `Opts.ftol`: Tolerance parameter for F. Default value: $1e-10$.
- `Opts.gradtol`: Tolerance parameter for dF. Default value: $1e-7$.
- `Opts.verbose`: Flag for print out the number of bootstrap samples, logical 0 or 1. Default value: 0.

Output

bootse: The standard error for elements in β computed by bootstrap. An r by p matrix.

Description

This function computes the bootstrap standard errors for the regression coefficients in the envelope model by bootstrapping the residuals.

Example

```
load wheatprotein.txt
X = wheatprotein(:, 8);
Y = wheatprotein(:, 1 : 6);
alpha = 0.01;
```

```
u = modelselectlrt(X, Y, alpha, 'env')

u =

    1

B = 100;
bootse = bstrp_env(X, Y, u, B)

bootse =

    0.2893
    0.4260
    0.3523
    0.5628
    0.1675
    0.6192
```

2.4 dF4env

The first derivative of the objective function for computing the envelope subspace.

Contents

- Syntax
- Input
- Output
- Description

Syntax

```
df = dF4env(R, DataParameter)
```

Input

R: An r by u semi orthogonal matrix, $0 < u \leq r$.

DataParameter: A structure that contains the statistics calculated from the data.

Output

df: An r by u matrix containing the value of the derivative function evaluated at R .

Description

The objective function is derived in Section 4.3 in Cook et al. (2010) by using maximum likelihood estimation. This function is the derivative of the objective function.

2.5 env

Fit the envelope model.

Contents

- Syntax
- Input
- Output
- Description
- References
- Example

Syntax

```
ModelOutput = env(X, Y, u)
ModelOutput = env(X, Y, u, Opts)
```

Input

X: Predictors. An n by p matrix, p is the number of predictors. The predictors can be univariate or multivariate, discrete or continuous.

Y: Multivariate responses. An n by r matrix, r is the number of responses and n is number of observations. The responses must be continuous variables, and r should be strictly greater than p .

u: Dimension of the envelope. An integer between 0 and r .

Opts: A list containing the optional input parameters, to control the iterations in `sg_min`. If one or several (even all) fields are not defined, the default settings are used.

- `Opts.maxIter`: Maximum number of iterations. Default value: 300.
- `Opts.ftol`: Tolerance parameter for F. Default value: $1e-10$.
- `Opts.gradtol`: Tolerance parameter for dF. Default value: $1e-7$.
- `Opts.verbose`: Flag for print out Grassmann manifold optimization process, logical 0 or 1. Default value: 0.
- `Opts.init`: The initial value for the envelope subspace. An r by u matrix. Default value is the one generated by function `get_Init`.

Output

ModelOutput: A list that contains the maximum likelihood estimators and some statistics.

- `ModelOutput.beta`: The envelope estimator of the regression coefficients β . An r by p matrix.
- `ModelOutput.Sigma`: The envelope estimator of the error covariance matrix. An r by r matrix.
- `ModelOutput.Gamma`: The orthogonal basis of the envelope subspace. An r by u semi-orthogonal matrix.
- `ModelOutput.Gamma0`: The orthogonal basis of the complement of the envelope subspace. An r by $r-u$ semi-orthogonal matrix.
- `ModelOutput.eta`: The coordinates of β with respect to `Gamma`. A u by p matrix.

- `ModelOutput.Omega`: The coordinates of Sigma with respect to Gamma. A u by u matrix.
- `ModelOutput.Omega0`: The coordinates of Sigma with respect to Gamma0. An r - u by r - u matrix.
- `ModelOutput.alpha`: The estimated intercept in the envelope model. An r by 1 vector.
- `ModelOutput.l`: The maximized log likelihood function. A real number.
- `ModelOutput.covMatrix`: The asymptotic covariance of $\text{vec}(\beta)$. An rp by rp matrix. The covariance matrix returned are asymptotic. For the actual standard errors, multiply by $1/n$.
- `ModelOutput.asySE`: The asymptotic standard error for elements in β under the envelope model. An r by p matrix. The standard errors returned are asymptotic, for actual standard errors, multiply by $1/\sqrt{n}$.
- `ModelOutput.ratio`: The asymptotic standard error ratio of the standard multivariate linear regression estimator over the envelope estimator, for each element in β . An r by p matrix.
- `ModelOutput.paramNum`: The number of parameters in the envelope model. A positive integer.
- `ModelOutput.n`: The number of observations in the data. A positive integer.

Description

This function fits the envelope model to the responses and predictors, using the maximum likelihood estimation. When the dimension of the envelope is between 1 and $r-1$, we implemented the algorithm in Cook et al. (2010). When the dimension is r , then the envelope model degenerates to the standard multivariate linear regression. When the dimension is 0, it means that X and Y are uncorrelated, and the fitting is different.

References

1. The codes are implemented based on the algorithm in Section 4.3 of Cook et al (2010).
2. The Grassmann manifold optimization step calls the package `sg_min` 2.4.3 by Ross Lipert (<http://web.mit.edu/~ripper/www/sgmin.html>).

Example

The following codes will reconstruct the results in the wheat protein data example in Cook et al. (2010).

```
load wheatprotein.txt
X = wheatprotein(:, 8);
Y = wheatprotein(:, 1 : 6);
alpha = 0.01;
u = modelselectlrt(X, Y, alpha, 'env');
```

`u =`

```
1
```

```
ModelOutput = env(X, Y, u)
```

```
ModelOutput =
```

```

    beta: [6x1 double]
    Sigma: [6x6 double]
    Gamma: [6x1 double]
    Gamma0: [6x5 double]
    eta: 8.5647
    Omega: 7.8762
    Omega0: [5x5 double]
    alpha: [6x1 double]
    l: -850.7592
    covMatrix: [6x6 double]
    asySE: [6x1 double]
    ratio: [6x1 double]
    paramNum: 28
    n: 50

```

```
ModelOutput.Omega
```

```
ans =
```

```
7.8762
```

```
eig(ModelOutput.Omega0)
```

```
ans =
```

```
1.0e+03 *
```

```

6.5166
0.2083
0.0201
0.0004
0.0003

```

```
ModelOutput.ratio
```

```
ans =
```

```
28.0945
```

18.4326
23.6384
16.3211
65.8245
6.4668

2.6 F4env

Objective function for computing the envelope subspace.

Contents

- Syntax
- Input
- Output
- Description

Syntax

```
f = F4env(R, DataParameter)
```

Input

R: An r by u semi orthogonal matrix, $0 < u \leq r$.

DataParameter: A structure that contains the statistics calculated from the data.

Output

f: A scalar containing the value of the objective function evaluated at R .

Description

The objective function is derived in Section 4.3 of Cook et al. (2010) using maximum likelihood estimation. The columns of the semi-orthogonal matrix that minimizes this function span the estimated envelope subspace.

2.7 lrt_env

Select the dimension of the envelope subspace using likelihood ratio testing.

Contents

- Syntax
- Input
- Output
- Description
- Example

Syntax

```
u = lrt_env(X, Y, alpha)
u = lrt_env(X, Y, alpha, Opts)
```

Input

X: Predictors. An n by p matrix, p is the number of predictors. The predictors can be univariate or multivariate, discrete or continuous.

Y: Multivariate responses. An n by r matrix, r is the number of responses and n is number of observations. The responses must be continuous variables.

alpha: Significance level for testing. A real number between 0 and 1, often taken at 0.05 or 0.01.

Opts: A list containing the optional input parameters, to control the iterations in `sg_min`. If one or several (even all) fields are not defined, the default settings are used.

- `Opts.maxIter`: Maximum number of iterations. Default value: 300.
- `Opts.ftol`: Tolerance parameter for F. Default value: $1e-10$.
- `Opts.gradtol`: Tolerance parameter for dF. Default value: $1e-7$.
- `Opts.verbose`: Flag for print out dimension selection process, logical 0 or 1. Default value: 0.

Output

u: Dimension of the envelope. An integer between 0 and r .

Description

This function implements the likelihood ratio testing procedure to select the dimension of the envelope subspace, with pre-specified significance level α .

Example

```
load wheatprotein.txt
X = wheatprotein(:, 8);
Y = wheatprotein(:, 1 : 6);
alpha = 0.01;
u = lrt_env(X, Y, alpha)
```

u =

1

2.8 predict_env

Perform estimation or prediction under the envelope model.

Contents

- Syntax
- Input
- Output
- Description
- Example

Syntax

```
PredictOutput = predict_env(ModelOutput, Xnew, infType)
```

Input

ModelOutput: A list containing the maximum likelihood estimators and other statistics inherited from env.

Xnew: The value of X with which to estimate or predict Y. A p by 1 vector.

infType: A string of characters indicating the inference type, the choices can be 'estimation' or 'prediction'.

Output

PredictOutput: A list containing the results of the inference.

- PredictOutput.value: The fitted value or the prediction value evaluated at Xnew. An r by 1 vector.
- PredictOutput.covMatrix: The covariance matrix of PredictOutput.value. An r by r matrix.
- PredictOutput.SE: The standard error of elements in PredictOutput.value. An r by 1 vector.

Description

This function evaluates the envelope model at new value Xnew. It can perform estimation: find the fitted value when $X = X_{\text{new}}$, or prediction: predict Y when $X = X_{\text{new}}$. The covariance matrix and the standard errors are also provided.

Example

```
load wheatprotein.txt
X = wheatprotein(:, 8);
Y = wheatprotein(:, 1 : 6);
alpha = 0.01;
u = lrt_env(X, Y, alpha);
ModelOutput = env(X, Y, u);
Xnew = X(2, :)';
```

```
PredictOutput = predict_env(ModelOutput, Xnew, 'estimation')
[PredictOutput.value, Y(1, :)] % Compare the fitted value with the data
PredictOutput.SE
```

```
PredictOutput =
```

```
    value: [6x1 double]
  covMatrix: [6x6 double]
        SE: [6x1 double]
```

```
ans =
```

```
474.7135  468.0000
127.4740  123.0000
251.2044  246.0000
380.8280  374.0000
380.9473  386.0000
-6.3287  -11.0000
```

```
ans =
```

```
4.8892
4.0227
4.3237
4.7470
6.8186
2.6948
```

```
PredictOutput = predict_env(ModelOutput, Xnew, 'prediction')
PredictOutput.SE
```

```
PredictOutput =
```

```
    value: [6x1 double]
  covMatrix: [6x6 double]
        SE: [6x1 double]
```

```
ans =
```

```
474.7135
127.4740
251.2044
380.8280
380.9473
-6.3287
```

```
ans =
```

```
34.9161  
28.7280  
30.8775  
33.9006  
48.6945  
19.2448
```

2.9 testcoefficient_env

This function tests the null hypothesis $L * \beta * R = A$ versus the alternative hypothesis $L * \beta * R \neq A$, where β is estimated under the envelope model.

Contents

- Syntax
- Input
- Output
- Description
- Example

Syntax

```
TestOutput = testcoefficient_env(ModelOutput)
TestOutput = testcoefficient_env(ModelOutput, TestInput)
```

Input

ModelOutput: A list containing the maximum likelihood estimators and other statistics inherited from env.

TestInput: A list that specifies the null hypothesis, including L, R, and A. If not provided by the user, default values will be used.

- TestInput.L: The matrix multiplied to β on the left. It is a d1 by r matrix, while d1 is less than or equal to r. Default value: identity matrix I_r .
- TestInput.R: The matrix multiplied to β on the right. It is a p by d2 matrix, while d2 is less than or equal to p. Default value: identity matrix I_p .
- TestInput.A: The matrix on the right hand side of the equation. It is a d1 by d2 matrix. Default value: d1 by d2 zero matrix.

Output

TestOutput: A list containing test statistics, degrees of freedom for the reference chi-squared distribution, the p-value, and the covariance matrix of $\text{vec}(L\beta R)$. At the same time, a table is printed out.

- TestOutput.chisqStatistic: The test statistics. A real number.
- TestOutput.df: The degrees of freedom of the reference chi-squared distribution. A positive integer.
- TestOutput.pValue: p-value of the test. A real number in [0, 1].
- TestOutput.covMatrix: The covariance matrix of $\text{vec}(L\beta R)$. A d1 * d2 by d1 * d2 matrix.

Description

This function tests for hypothesis $H_0 : L\beta R = A$, versus $H_\alpha : L\beta R \neq A$. The β is estimated by the envelope model. If the user does not specify the values for L, R and A, then the test is equivalent to the standard F test on if $\beta = 0$. The test statistics used is $\text{vec}(L\beta R - A) \hat{\Sigma}^{-1} \text{vec}(L\beta R - A)^T$, and the reference distribution is chi-squared distribution with degrees of freedom d1 * d2.

Example

```

load wheatprotein.txt
X = wheatprotein(:, 8);
Y = wheatprotein(:, 1 : 6);
alpha = 0.01;
u = lrt_env(X, Y, alpha);
ModelOutput = env(X, Y, u);
TestOutout = testcoefficient_env(ModelOutput);

```

Test Hypothesis	Chisq Statistic	DF	P-value
L * beta * R = A	116.230	6	0.0000

```

r = size(Y, 2);
p = size(X, 2);
TestInput.L = rand(2, r);
TestInput.R = rand(p, 1);
TestInput.A = zeros(2, 1);
TestOutout = testcoefficient_env(ModelOutput, TestInput);

```

Test Hypothesis	Chisq Statistic	DF	P-value
L * beta * R = A	111.628	2	0.0000

envmean

3.1 aic_envmean

Select the dimension of the envelope subspace using Akaike information criterion.

Contents

- Syntax
- Input
- Output
- Description
- Example

Syntax

```
u = aic_envmean(Y)
u = aic_envmean(Y, Opts)
```

Input

Y: Data matrix. An n by p matrix, p is the dimension of the variable and n is number of observations.

Opts: A list containing the optional input parameters, to control the iterations in `sg_min`. If one or several (even all) fields are not defined, the default settings are used.

- `Opts.maxIter`: Maximum number of iterations. Default value: 300.
- `Opts.ftol`: Tolerance parameter for F. Default value: $1e-10$.
- `Opts.gradtol`: Tolerance parameter for dF. Default value: $1e-7$.
- `Opts.verbose`: Flag for print out dimension selection process, logical 0 or 1. Default value: 0.

Output

u: Dimension of the envelope. An integer between 0 and p .

Description

This function implements the Akaike information criteria (AIC) to select the dimension of the envelope subspace.

Example

```
load Adopted
Y = Adopted(:, 1 : 6);
u = aic_envmean(Y)
```

u =

3

3.2 bic_envmean

Select the dimension of the envelope subspace using Bayesian information criterion.

Contents

- Syntax
- Input
- Output
- Description
- Example

Syntax

```
u = bic_envmean(Y)
u = bic_envmean(Y, Opts)
```

Input

Y: Data matrix. An n by p matrix, p is the dimension of the variable and n is number of observations.

Opts: A list containing the optional input parameters, to control the iterations in `sg_min`. If one or several (even all) fields are not defined, the default settings are used.

- `Opts.maxIter`: Maximum number of iterations. Default value: 300.
- `Opts.ftol`: Tolerance parameter for F. Default value: $1e-10$.
- `Opts.gradtol`: Tolerance parameter for dF. Default value: $1e-7$.
- `Opts.verbose`: Flag for print out dimension selection process, logical 0 or 1. Default value: 0.

Output

u: Dimension of the envelope. An integer between 0 and p .

Description

This function implements the Bayesian information criteria (BIC) to select the dimension of the envelope subspace.

Example

```
load Adopted
Y = Adopted(:, 1 : 6);
u = bic_envmean(Y)
```

u =

3

3.3 bstrp_envmean

Compute bootstrap standard error for the envelope estimator of the multivariate mean.

Contents

- Syntax
- Input
- Output
- Description
- Example

Syntax

```
bootse = bstrp_envmean(Y, u, B)
bootse = bstrp_envmean(Y, u, B, Opts)
```

Input

Y: Data matrix. An n by p matrix, p is the dimension of the variable and n is number of observations.

u: Dimension of the envelope subspace. A positive integer between 0 and p .

B: Number of bootstrap samples. A positive integer.

Opts: A list containing the optional input parameters, to control the iterations in `sg_min`. If one or several (even all) fields are not defined, the default settings are used.

- `Opts.maxIter`: Maximum number of iterations. Default value: 300.
- `Opts.ftol`: Tolerance parameter for F. Default value: $1e-10$.
- `Opts.gradtol`: Tolerance parameter for dF. Default value: $1e-7$.
- `Opts.verbose`: Flag for print out the number of bootstrap samples, logical 0 or 1. Default value: 0.

Output

bootse: The standard error for elements in μ computed by bootstrap. A p dimensional column vector.

Description

This function computes the bootstrap standard errors for the envelope estimator of the multivariate mean by bootstrapping the residuals.

Example

```
load Adopted
Y = Adopted(:, 1 : 6);
u = bic_envmean(Y)
```

```
u =
```

```
3
```

```
B = 100;
```

```
bootse = bstrp_envmean(Y, u, B)
```

```
bootse =
```

```
0.5155
```

```
13.4169
```

```
13.2268
```

```
17.0221
```

```
21.4016
```

```
22.0329
```

3.4 dF4envmean

The first derivative of the objective function for computing the envelope subspace.

Contents

- Syntax
- Input
- Output
- Description

Syntax

```
df = dF4envmean(R, DataParameter)
```

Input

R: A p by u semi orthogonal matrix, $0 < u \leq p$.

DataParameter: A structure that contains the statistics calculated from the data.

Output

df: A p by u matrix containing the value of the derivative function evaluated at R .

Description

The objective function is derived by maximum likelihood estimation. This function is the derivative of the objective function.

3.5 envmean

Provide envelope estimator for the multivariate mean.

Contents

- Syntax
- Input
- Output
- Description
- References
- Example

Syntax

```
ModelOutput = envmean(Y, u)
ModelOutput = envmean(Y, u, Opts)
```

Input

Y: Data matrix. An n by p matrix, p is the dimension of the variable and n is number of observations.

u: Dimension of the envelope. An integer between 0 and p .

Opts: A list containing the optional input parameters, to control the iterations in `sg_min`. If one or several (even all) fields are not defined, the default settings are used.

- `Opts.maxIter`: Maximum number of iterations. Default value: 300.
- `Opts.ftol`: Tolerance parameter for F. Default value: $1e-10$.
- `Opts.gradtol`: Tolerance parameter for dF. Default value: $1e-7$.
- `Opts.verbose`: Flag for print out Grassmann manifold optimization process, logical 0 or 1. Default value: 0.
- `Opts.init`: The initial value for the envelope subspace. A p by u matrix. Default value is the one generated by function `get_Init4envmean`.

Output

ModelOutput: A list that contains the maximum likelihood estimators and some statistics.

- `ModelOutput.mu`: The envelope estimator of the multivariate mean μ . A p dimensional column vector.
- `ModelOutput.Sigma`: The envelope estimator of the error covariance matrix. A p by p matrix.
- `ModelOutput.Gamma`: The orthogonal basis of the envelope subspace. A p by u semi-orthogonal matrix.
- `ModelOutput.Gamma0`: The orthogonal basis of the complement of the envelope subspace. An p by $p-u$ semi-orthogonal matrix.
- `ModelOutput.eta`: The coordinates of μ with respect to `Gamma`. A u dimensional column vector.
- `ModelOutput.Omega`: The coordinates of `Sigma` with respect to `Gamma`. A u by u matrix.
- `ModelOutput.Omega0`: The coordinates of `Sigma` with respect to `Gamma0`. A $p-u$ by $p-u$ matrix.

- `ModelOutput.l`: The maximized log likelihood function. A real number.
- `ModelOutput.covMatrix`: The asymptotic covariance of μ . A p by p matrix. The covariance matrix returned are asymptotic. For the actual standard errors, multiply by $1/n$.
- `ModelOutput.asySE`: The asymptotic standard error for elements in μ under the envelope model. A p dimensional column vector. The standard errors returned are asymptotic, for actual standard errors, multiply by $1/\sqrt{n}$.
- `ModelOutput.ratio`: The asymptotic standard error ratio of the standard multivariate linear regression estimator over the envelope estimator, for each element in μ . A p dimensional column vector.
- `ModelOutput.paramNum`: The number of parameters in the envelope model. A positive integer.
- `ModelOutput.n`: The number of observations in the data. A positive integer.

Description

This function provides an envelope estimator for the multivariate mean, with a given dimension of the envelope subspace u . The estimator is obtained using the maximum likelihood estimation. When the dimension is p , then the envelope model degenerates to the standard sample mean. When the dimension is 0, it means that Y has mean 0.

References

The Grassmann manifold optimization step calls the package `sg_min` 2.4.3 by Ross Lippert (<http://web.mit.edu/~ripper/www.sgmin.html>).

Example

```
load Adopted
Y = Adopted(:, 1 : 6);
u = bic_envmean(Y)
```

`u =`

3

```
ModelOutput = envmean(Y, u)
```

`ModelOutput =`

```
    mu: [6x1 double]
   Sigma: [6x6 double]
   Gamma: [6x3 double]
 Gamma0: [6x3 double]
    eta: [3x1 double]
   Omega: [3x3 double]
 Omega0: [3x3 double]
        l: -1.3492e+03
 covMatrix: [6x6 double]
```



```

      asySE: [6x1 double]
      ratio: [6x1 double]
paramNum: 24
      n: 62

```

ModelOutput.mu

ans =

```

12.3258
85.9841
115.5767
112.1291
114.4862
106.4240

```

ModelOutput.Sigma

ans =

Columns 1 through 5

8.3278	2.9150	-4.0008	0.1057	0.3731
2.9150	235.6587	5.8146	42.1613	59.7492
-4.0008	5.8146	179.2066	91.7228	92.9563
0.1057	42.1613	91.7228	167.1073	114.4203
0.3731	59.7492	92.9563	114.4203	184.3248
-0.8157	71.9394	72.1815	110.2918	161.8752

Column 6

```

-0.8157
71.9394
72.1815
110.2918
161.8752
233.9185

```

3.6 F4envmean

Objective function for computing the envelope subspace.

Contents

- Syntax
- Input
- Output
- Description

Syntax

```
f = F4envmean(R, DataParameter)
```

Input

R: A p by u semi orthogonal matrix, $0 < u \leq p$.

DataParameter: A structure that contains the statistics calculated from the data.

Output

f: A scalar containing the value of the objective function evaluated at **R**.

Description

The objective function is derived by maximum likelihood estimation. The columns of the semi-orthogonal matrix that minimizes this function span the estimated envelope subspace.

3.7 lrt_envmean

Select the dimension of the envelope subspace using likelihood ratio testing.

Contents

- Syntax
- Input
- Output
- Description
- Example

Syntax

```
u = lrt_envmean(Y, alpha)
u = lrt_envmean(Y, alpha, Opts)
```

Input

Y: Data matrix. An n by p matrix, p is the dimension of the variable and n is number of observations.

alpha: Significance level for testing. A real number between 0 and 1, often taken at 0.05 or 0.01.

Opts: A list containing the optional input parameters, to control the iterations in `sg_min`. If one or several (even all) fields are not defined, the default settings are used.

- `Opts.maxIter`: Maximum number of iterations. Default value: 300.
- `Opts.ftol`: Tolerance parameter for F. Default value: $1e-10$.
- `Opts.gradtol`: Tolerance parameter for dF. Default value: $1e-7$.
- `Opts.verbose`: Flag for print out dimension selection process, logical 0 or 1. Default value: 0.

Output

u: Dimension of the envelope. An integer between 0 and p .

Description

This function implements the likelihood ratio testing procedure to select the dimension of the envelope subspace, with pre-specified significance level α .

Example

```
load Adopted
Y = Adopted(:, 1 : 6);
alpha = 0.01;
u = lrt_envmean(Y, alpha)
```

$u =$ 2

3.8 predict_envmean

Perform estimation of the multivariate mean or prediction for a new observation.

Contents

- Syntax
- Input
- Output
- Description
- Example

Syntax

```
PredictOutput = predict_envmean(ModelOutput, infType)
```

Input

ModelOutput: A list containing the maximum likelihood estimators and other statistics inherited from envmean.

infType: A string of characters indicting the inference type, the choices can be 'estimation' or 'prediction'.

Output

PredictOutput: A list containing the results of the inference.

- PredictOutput.value: The estimated multivariate mean or the prediction value. A p dimensional column vector.
- PredictOutput.covMatrix: The covariance matrix of PredictOutput.value. A p by p matrix.
- PredictOutput.SE: The standard error of elements in PredictOutput.value. A p dimensional column vector.

Description

If the inference type is prediction, this function predicts a new observation and gives its covariance matrix and standard errors of its elements. If the inference type is estimation, this function gives the estimation of the multivariate mean, its covariance matrix and standard errors of its elements.

Example

```
load Adopted
Y = Adopted(:, 1 : 6);
u = bic_envmean(Y);
ModelOutput = envmean(Y, u);
PredictOutput = predict_envmean(ModelOutput, 'prediction')
```

```
PredictOutput =
```

```
    value: [6x1 double]  
    covMatrix: [6x6 double]  
    SE: [6x1 double]
```

```
PredictOutput.value
```

```
ans =
```

```
12.3258  
85.9841  
115.5767  
112.1291  
114.4862  
106.4240
```

```
PredictOutput.SE
```

```
ans =
```

```
2.9090  
15.4745  
13.4943  
13.0308  
13.6857  
15.4172
```

3.9 testcoefficient_envmean

This function tests the null hypothesis $L * \mu = A$ versus the alternative hypothesis $L * \mu \neq A$, where μ is the multivariate mean.

Contents

- Syntax
- Input
- Output
- Description
- Example

Syntax

```
TestOutput = testcoefficient_envmean(ModelOutput)
TestOutput = testcoefficient_envmean(ModelOutput, TestInput)
```

Input

ModelOutput: A list containing the maximum likelihood estimators and other statistics inherited from envmean.

TestInput: A list that specifies the null hypothesis, including L and A. If not provided by the user, default values will be used.

- TestInput.L: The matrix multiplied to μ on the left. It is a d1 by p matrix, while d1 is less than or equal to p. Default value: identity matrix I_p .
- TestInput.A: The vector on the right hand side of the equation. It is a d1 dimensional column vector. Default value: d1 by d2 zero matrix.

Output

TestOutput: A list containing test statistics, degrees of freedom for the reference chi-squared distribution, the p-value, and the covariance matrix of $L\mu$. At the same time, a table is printed out.

- TestOutput.chisqStatistic: The test statistics. A real number.
- TestOutput.df: The degrees of freedom of the reference chi-squared distribution. A positive integer.
- TestOutput.pValue: p-value of the test. A real number in [0, 1].
- TestOutput.covMatrix: The covariance matrix of $L\mu$. A d1 dimensional column vector.

Description

This function tests for hypothesis $H_0 : L\mu = A$, versus $H_\alpha : L\mu \neq A$. The μ is estimated by the envelope model. If the user does not specify the values for L and A, then the test is equivalent to the standard F test on if $\mu = 0$. The test statistics used is $(L\mu - A) \hat{\Sigma}^{-1} (L\mu - A)^T$, and the reference distribution is chi-squared distribution with degrees of freedom d1.

Example

```

load Adopted
Y = Adopted(:, 1 : 6);
u = bic_envmean(Y);
ModelOutput = envmean(Y, u);
TestOutout = testcoefficient_envmean(ModelOutput);

```

Test Hypothesis	Chisq Statistic	DF	P-value
L * mu = A	8716.691	6	0.0000

```

p = size(Y, 2);
TestInput.L = rand(2, p);
TestInput.A = zeros(2, 1);
TestOutout = testcoefficient_envmean(ModelOutput, TestInput);

```

Test Hypothesis	Chisq Statistic	DF	P-value
L * mu = A	6543.508	2	0.0000

envseq

4.1 bstrp_envseq

Compute bootstrap standard errors of the envelope model using a sequential algorithm.

Contents

- Syntax
- Input
- Output
- Description
- Example

Syntax

```
bootse = bstrp_envseq(X, Y, u, B)
bootse = bstrp_envseq(X, Y, u, B, Opts)
```

Input

X: Predictors. An n by p matrix, p is the number of predictors. The predictors can be univariate or multivariate, discrete or continuous.

Y: Multivariate responses. An n by r matrix, r is the number of responses and n is number of observations. The responses must be continuous variables, and r should be strictly greater than p .

u: Dimension of the envelope subspace. A positive integer between 0 and p .

B: Number of bootstrap samples. A positive integer.

Opts: A list containing the optional input parameters. If one or several (even all) fields are not defined, the default settings are used.

- **Opts.verbose:** Flag for print out the number of bootstrap samples, logical 0 or 1. Default value: 0.

Output

bootse: The standard error for elements in β computed by bootstrap. An r by p matrix.

Description

This function computes the bootstrap standard errors for the regression coefficients in the envelope model by bootstrapping the residuals. The envelope model is applied for the reduction on X , using a sequential algorithm.

Example

```
load Rohwer
X = Rohwer(:, 4 : 5);
Y = Rohwer(:, 1 : 3);
m = 5;
u = mfoldcv_envseq(X, Y, m)
```

u =

1

```
B = 100;
bootse = bstrp_envseq(X, Y, u, B)
```

bootse =

0.8738	0.6855
0.5191	0.4404
0.0961	0.0654

4.2 envseq

Fit the envelope model using a sequential algorithm.

Contents

- Syntax
- Input
- Output
- Description
- References
- Example

Syntax

```
ModelOutput = envseq(X, Y, u)
```

Input

X: Predictors. An n by p matrix, p is the number of predictors. The predictors can be univariate or multivariate, discrete or continuous.

Y: Multivariate responses. An n by r matrix, r is the number of responses and n is number of observations. The responses must be continuous variables, and r should be strictly greater than p .

u: Dimension of the envelope. An integer between 0 and $(m-1) * n / m-1$.

Output

ModelOutput: A list that contains the maximum likelihood estimators and some statistics.

- ModelOutput.beta: The envelope estimator of the regression coefficients β . An r by p matrix.
- ModelOutput.Sigma: The envelope estimator of the error covariance matrix. An r by r matrix.
- ModelOutput.Gamma: The orthogonal basis of the envelope subspace. An r by u semi-orthogonal matrix.
- ModelOutput.Gamma0: The orthogonal basis of the complement of the envelope subspace. An r by $r-u$ semi-orthogonal matrix.
- ModelOutput.eta: The coordinates of β with respect to Gamma. An u by p matrix.
- ModelOutput.Omega: The coordinates of Sigma with respect to Gamma. An u by u matrix.
- ModelOutput.Omega0: The coordinates of Sigma with respect to Gamma0. An $r-u$ by $r-u$ matrix.
- ModelOutput.alpha: The estimated intercept in the envelope model. An r by 1 vector.
- ModelOutput.paramNum: The number of parameters in the envelope model. A positive integer.
- ModelOutput.n: The number of observations in the data. A positive integer.

Description

This function fits the envelope model to the responses and predictors, using the maximum likelihood estimation. When the dimension of the envelope is between 1 and $r-1$, we implemented the algorithm in Cook et al. (2010). When the dimension is r , then the envelope model degenerates to the standard multivariate linear regression. When the dimension is 0, it means that X and Y are uncorrelated, and the fitting is different.

References

The codes are implemented based on the sequential algorithm in the lecture notes of Cook (2012).

Example

```
load Rohwer
X = Rohwer(:, 4 : 5);
Y = Rohwer(:, 1 : 3);
m = 5;
u = mfoldcv_envseq(X, Y, m)

u =

    1

ModelOutput = envseq(X, Y, u)

ModelOutput =

    beta: [3x2 double]
    Sigma: [3x3 double]
    Gamma: [3x1 double]
    Gamma0: [3x2 double]
    eta: [2.5708 1.1966]
    Omega: 752.8146
    Omega0: [2x2 double]
    alpha: [3x1 double]
    paramNum: 11
    n: 69

ModelOutput.Sigma

ans =

    587.2575    229.9647    37.2716
    229.9647    421.1372    42.9256
    37.2716     42.9256    12.2696
```

4.3 mfoldcv_envseq

Select the dimension of the envelope subspace using m-fold cross validation for envelope model using a sequential algorithm.

Contents

- Syntax
- Input
- Output
- Description
- Example

Syntax

```
u = mfoldcv_envseq(X, Y, m)
u = mfoldcv_envseq(X, Y, m, Opts)
```

Input

X: Predictors. An n by p matrix, p is the number of predictors and n is number of observations.

Y: Responses. An n by r matrix, r is the number of responses. The number of the responses should be greater than the number of the predictors. And they must be continuous variables.

m: A positive integer that is used to indicate m-fold cross validation.

Opts: A list containing the optional input parameters. If one or several (even all) fields are not defined, the default settings are used.

- Opts.verbose: Flag for print out the number of bootstrap samples, logical 0 or 1. Default value: 0.

Output

***u*:** The dimension of the envelope subspace selected by m-fold cross validation.

Description

This function implements m-fold cross validation to select the dimension of the envelope space, based on prediction performance. For each u, the data is partitioned into m parts, each part is in turn used for testing for the prediction performance while the rest m-1 parts are used for training. The dimension is select as the one that minimizes the average prediction errors. If Y is multivariate, the identity inner product is used for computing the prediction errors.

Example

```
load Rohwer
X = Rohwer(:, 4 : 5);
Y = Rohwer(:, 1 : 3);
m = 5;
```

```
u = mfoldcv_envpls(X, Y, m)
```

```
u =
```

```
1
```

henv

5.1 aic_henv

Select the dimension of the envelope subspace using Akaike information criterion for the heteroscedastic envelope model.

Contents

- Syntax
- Input
- Output
- Description
- Example

Syntax

```
u = aic_henv(X, Y)
u = aic_henv(X, Y, Opts)
```

Input

X: Group indicators. A matrix with n rows. X can only have p unique rows, where p is the number of groups. For example, if there are two groups, X can only have 2 different kinds of rows, such as (0, 1) and (1, 0), or (1, 0, 10) and (0, 5, 6). The number of columns is not restricted, as long as X only has p unique rows.

Y: Multivariate responses. An n by r matrix, r is the number of responses and n is number of observations. The responses must be continuous variables, and r should be greater than p .

Opts: A list containing the optional input parameters, to control the iterations in `sg_min`. If one or several (even all) fields are not defined, the default settings are used.

- `Opts.maxIter`: Maximum number of iterations. Default value: 300.
- `Opts.ftol`: Tolerance parameter for F. Default value: $1e-10$.
- `Opts.gradtol`: Tolerance parameter for dF. Default value: $1e-7$.
- `Opts.verbose`: Flag for print out dimension selection process, logical 0 or 1. Default value: 0.

Output

u: Dimension of the envelope. An integer between 0 and r .

Description

This function implements the Akaike information criteria (AIC) to select the dimension of the envelope subspace for the heteroscedastic envelope model.

Example

```
load waterstrider.mat
u = aic_henv(X, Y)
```

u =

6

5.2 bic_henv

Select the dimension of the envelope subspace using Bayesian information criterion for the heteroscedastic envelope model.

Contents

- Syntax
- Input
- Output
- Description
- Example

Syntax

```
u = bic_henv(X, Y)
u = bic_henv(X, Y, Opts)
```

Input

X: Group indicators. A matrix with n rows. X can only have p unique rows, where p is the number of groups. For example, if there are two groups, X can only have 2 different kinds of rows, such as (0, 1) and (1, 0), or (1, 0, 10) and (0, 5, 6). The number of columns is not restricted, as long as X only has p unique rows.

Y: Multivariate responses. An n by r matrix, r is the number of responses and n is number of observations. The responses must be continuous variables, and r should be greater than p .

Opts: A list containing the optional input parameters, to control the iterations in `sg_min`. If one or several (even all) fields are not defined, the default settings are used.

- `Opts.maxIter`: Maximum number of iterations. Default value: 300.
- `Opts.ftol`: Tolerance parameter for F. Default value: $1e-10$.
- `Opts.gradtol`: Tolerance parameter for dF. Default value: $1e-7$.
- `Opts.verbose`: Flag for print out dimension selection process, logical 0 or 1. Default value: 0.

Output

u: Dimension of the envelope. An integer between 0 and r .

Description

This function implements the Bayesian information criteria (BIC) to select the dimension of the envelope subspace for the heteroscedastic envelope model.

Example

```
load waterstrider.mat
u = bic_henv(X, Y)
```

u =

4

5.3 bstrp_henv

Compute bootstrap standard error for the heteroscedastic envelope model.

Contents

- Syntax
- Input
- Output
- Description
- Example

Syntax

```
bootse = bstrp_henv(X, Y, u, B)
bootse = bstrp_henv(X, Y, u, B, Opts)
```

Input

X: Group indicators. A matrix with n rows. X can only have p unique rows, where p is the number of groups. For example, if there are two groups, X can only have 2 different kinds of rows, such as (0, 1) and (1, 0), or (1, 0, 10) and (0, 5, 6). The number of columns is not restricted, as long as X only has p unique rows.

Y: Multivariate responses. An n by r matrix, r is the number of responses and n is number of observations. The responses must be continuous variables, and r should be greater than p .

u: Dimension of the envelope subspace. A positive integer between 0 and r .

B: Number of bootstrap samples. A positive integer.

Opts: A list containing the optional input parameters, to control the iterations in `sg_min`. If one or several (even all) fields are not defined, the default settings are used.

- `Opts.maxIter`: Maximum number of iterations. Default value: 300.
- `Opts.ftol`: Tolerance parameter for F. Default value: $1e-10$.
- `Opts.gradtol`: Tolerance parameter for dF. Default value: $1e-7$.
- `Opts.verbose`: Flag for print out the number of bootstrap samples, logical 0 or 1. Default value: 0.

Output

bootse: The standard error for elements in β computed by bootstrap. An r by p matrix.

Description

This function computes the bootstrap standard errors for the regression coefficients in the heteroscedastic envelope model by bootstrapping the residuals.

Example

```
load waterstrider.mat

u = lrt_henv(X, Y, 0.01)

u =

    6

B = 100;
bootse = bstrp_henv(X, Y, u, B)

bootse =

    0.0305    0.0466    0.0647
    0.0309    0.0485    0.0682
    0.0305    0.0432    0.0638
    0.0205    0.0289    0.0425
    0.0385    0.0553    0.0799
    0.0295    0.0427    0.0618
    0.0389    0.0567    0.0819
    0.0321    0.0463    0.0665
```

5.4 dF4henv

The first derivative of the objective function for computing the envelope subspace in the heteroscedastic envelope model.

Contents

- Syntax
- Input
- Output
- Description

Syntax

```
df = dF4henv(R, DataParameter)
```

Input

R: An r by u semi orthogonal matrix, $0 < u \leq r$.

DataParameter: A structure that contains the statistics calculated from the data.

Output

df: An r by u matrix containing the value of the derivative function evaluated at R .

Description

The objective function is derived in Section 2.2 in Su and Cook (2012) by using maximum likelihood estimation. This function is the derivative of the objective function.

5.5 F4henv

Objective function for computing the envelope subspace in heteroscedastic envelope model.

Contents

- Syntax
- Input
- Output
- Description

Syntax

```
f = F4henv(R, DataParameter)
```

Input

R: An r by u semi orthogonal matrix, $0 < u \leq r$.

DataParameter: A structure that contains the statistics calculated from the data.

Output

f: A scalar containing the value of the objective function evaluated at R .

Description

The objective function is derived in Section 2.2 of Su and Cook (2012) using maximum likelihood estimation. The columns of the semi-orthogonal matrix that minimizes this function span the estimated envelope subspace in the heteroscedastic envelope model.

5.6 henv

Fit the heteroscedastic envelope model.

Contents

- Syntax
- Input
- Output
- Description
- References
- Example

Syntax

```
ModelOutput = henv(X, Y, u)
ModelOutput = henv(X, Y, u, Opts)
```

Input

X: Group indicators. A matrix with n rows. X can only have p unique rows, where p is the number of groups. For example, if there are two groups, X can only have 2 different kinds of rows, such as (0, 1) and (1, 0), or (1, 0, 10) and (0, 5, 6). The number of columns is not restricted, as long as X only has p unique rows.

Y: Multivariate responses. An n by r matrix, r is the number of responses and n is number of observations. The responses must be continuous variables, and r should be greater than p .

u: Dimension of the envelope. An integer between 0 and r .

Opts: A list containing the optional input parameters, to control the iterations in `sg_min`. If one or several (even all) fields are not defined, the default settings are used.

- `Opts.maxIter`: Maximum number of iterations. Default value: 300.
- `Opts.ftol`: Tolerance parameter for F. Default value: $1e-10$.
- `Opts.gradtol`: Tolerance parameter for dF. Default value: $1e-7$.
- `Opts.verbose`: Flag for print out Grassmann manifold optimization process, logical 0 or 1. Default value: 0.
- `Opts.init`: The initial value for the heteroscedastic envelope subspace. An r by u matrix. Default value is the one generated by function `get_Init4henv`.

Output

ModelOutput: A list that contains the maximum likelihood estimators and some statistics.

- `ModelOutput.mu`: The heteroscedastic envelope estimator of the grand mean. A r by 1 vector.
- `ModelOutput.mug`: The heteroscedastic envelope estimator of the group mean. A r by p matrix, the i th column of the matrix contains the mean for the i th group.
- `ModelOutput.Yfit`: A n by r matrix, the i th row gives the group mean of the group that the i th observation belongs to. As X is just a group indicator, and is not ordinal, `ModelOutput.mug` alone does not tell which group corresponds to which group mean.

- `ModelOutput.Gamma`: The orthogonal basis of the envelope subspace. An r by u semi-orthogonal matrix.
- `ModelOutput.Gamma0`: The orthogonal basis of the complement of the envelope subspace. An r by $r-u$ semi-orthogonal matrix.
- `ModelOutput.beta`: The heteroscedastic envelope estimator of the group main effect. An r by p matrix, the i th column of the matrix contains the main effect for the i th group.
- `ModelOutput.groupInd`: A matrix containing the unique values of group indicators. The matrix has p rows. The group mean of the i th row is stored in the i th column of `ModelOutput.mug`.
- `ModelOutput.Sigma`: The heteroscedastic envelope estimator of the error covariance matrix. A three dimensional matrix with dimension r , r and p , `ModelOutput.Sigma(:, :, i)` contains the estimated covariance matrix for the i th group.
- `ModelOutput.eta`: The coordinates of β with respect to `Gamma`. An u by p matrix, the i th column contains the coordinates of the main effect of the i th group with respect to `Gamma`.
- `ModelOutput.Omega`: The coordinates of `Sigma` with respect to `Gamma`. An u by u by p matrix, `ModelOutput.Omega(:, :, i)` contains the coordinates of the covariance matrix of the i th group with respect to `Gamma`.
- `ModelOutput.Omega0`: The coordinates of `Sigma` with respect to `Gamma0`. An $r - u$ by $r - u$ matrix.
- `ModelOutput.l`: The maximized log likelihood function. A real number.
- `ModelOutput.paramNum`: The number of parameters in the heteroscedastic envelope model. A positive integer.
- `ModelOutput.covMatrix`: The asymptotic covariance of $(\mu', \text{vec}(\beta'))'$. An $r(p + 1)$ by $r(p + 1)$ matrix. The covariance matrix returned are asymptotic. For the actual standard errors, multiply by $1 / n$.
- `ModelOutput.asyHenv`: The asymptotic standard errors for elements in β under the heteroscedastic envelope model. An r by p matrix. The standard errors returned are asymptotic, for actual standard errors, multiply by $1 / \text{sqrt}(n)$.
- `ModelOutput.ratio`: The asymptotic standard error ratio of the standard multivariate linear regression estimator over the heteroscedastic envelope estimator. An r by p matrix, the (i, j) th element in `ModelOutput.ratio` is the elementwise standard error ratio for the i th element in the j th group mean effect.
- `ModelOutput.ng`: The number of observations in each group. A p by 1 vector.

Description

This function fits the heteroscedastic envelope model to the responses and predictors, using the maximum likelihood estimation. When the dimension of the envelope is between 1 and $r-1$, we implemented the algorithm in Su and Cook (2012). When the dimension is r , then the envelope model degenerates to the standard multivariate linear model for comparing group means. When the dimension is 0, it means there is not any group effect, and the fitting is different.

References

1. The codes are implemented based on the algorithm in Section 2.2 of Su and Cook (2012).
2. The Grassmann manifold optimization step calls the package `sg_min` 2.4.3 by Ross Lipert (<http://web.mit.edu/~ripper/www/sgmin.html>).

Example

The following codes produce the results of the water strider example in Su and Cook (2012).

```

load waterstrider.mat
u = lrt_henv(X, Y, 0.01)

u =

    6

ModelOutput = henv(X, Y, u)
ModelOutput.ratio

ModelOutput =

    mu: [8x1 double]
    mug: [8x3 double]
    Yfit: [90x8 double]
    Gamma: [8x6 double]
    Gamma0: [8x2 double]
    beta: [8x3 double]
    groupInd: [3x2 double]
    Sigma: [8x8x3 double]
    eta: [6x3 double]
    Omega: [6x6x3 double]
    Omega0: [2x2 double]
    paramNum: 98
    l: 1.0051e+03
    covMatrix: [32x32 double]
    asyHenv: [8x3 double]
    ratio: [8x3 double]
    ng: [3x1 double]

ans =

    6.5439    11.2830    6.4954
    4.6325     5.3226    4.7242
    4.4456     5.0741    4.4198
    4.7338     6.2469    5.1937
    8.0377    12.5386    9.4823
    9.5067    11.5974   11.3444
   11.8632    15.6080   12.5611
    6.9792    11.1559   10.1002

```

5.7 lrt_henv

Select the dimension of the envelope subspace using likelihood ratio testing for the heteroscedastic envelope model.

Contents

- Syntax
- Input
- Output
- Description
- Example

Syntax

```
u = lrt_henv(X, Y, alpha)
u = lrt_henv(X, Y, alpha, Opts)
```

Input

X: Group indicators. A matrix with n rows. X can only have p unique rows, where p is the number of groups. For example, if there are two groups, X can only have 2 different kinds of rows, such as (0, 1) and (1, 0), or (1, 0, 10) and (0, 5, 6). The number of columns is not restricted, as long as X only has p unique rows.

Y: Multivariate responses. An n by r matrix, r is the number of responses and n is number of observations. The responses must be continuous variables, and r should be greater than p .

alpha: Significance level for testing. A real number between 0 and 1, often taken at 0.05 or 0.01.

Opts: A list containing the optional input parameters, to control the iterations in `sg_min`. If one or several (even all) fields are not defined, the default settings are used.

- `Opts.maxIter`: Maximum number of iterations. Default value: 300.
- `Opts.ftol`: Tolerance parameter for F. Default value: $1e-10$.
- `Opts.gradtol`: Tolerance parameter for dF. Default value: $1e-7$.
- `Opts.verbose`: Flag for print out dimension selection process, logical 0 or 1. Default value: 0.

Output

u: Dimension of the envelope. An integer between 0 and r .

Description

This function implements the likelihood ratio testing procedure to select the dimension of the envelope subspace in heteroscedastic envelope model, with pre-specified significance level α .

Example

```
load waterstrider.mat  
u = lrt_henv(X, Y, 0.01)
```

u =

6

5.8 predict_henv

Perform estimation or prediction under the heteroscedastic envelope model.

Contents

- Syntax
- Input
- Output
- Description
- Example

Syntax

```
PredictOutput = predict_henv(ModelOutput, Xnew, infType)
```

Input

ModelOutput: A list containing the maximum likelihood estimators and other statistics inherited from henv.

Xnew: A group indicator. It must be a column vector, whose transpose is the same as one of the group indicators from the original data.

infType: A string of characters indicating the inference type, the choices can be 'estimation' or 'prediction'.

Output

PredictOutput: A list containing the results of the inference.

- PredictOutput.value: The fitted value or the prediction value evaluated at Xnew. An r by 1 vector.
- PredictOutput.covMatrix: The covariance matrix of PredictOutput.value. An r by r matrix.
- PredictOutput.SE: The standard error of elements in PredictOutput.value. An r by 1 vector.

Description

This function evaluates the inner envelope model at new value Xnew. It can perform estimation: find the group mean for the group indicated by Xnew, or prediction: predict Y for the group indicated by Xnew. The covariance matrix and the standard errors are also provided.

Example

```
load waterstrider.mat
u = lrt_henv(X, Y, 0.01);
ModelOutput = henv(X, Y, u);
ModelOutput.groupInd
ModelOutput.mug
Xnew = X(1, :).'
```

```

ans =

    -1    -1
     0     1
     1     0

ans =

    -1.1417    -1.1267    -1.0845
    -1.4063    -1.4067    -1.3132
    -1.3314    -1.3336    -1.2152
    -0.3113    -0.1839    -0.1736
     0.4003     0.3847     0.3072
     0.4107     0.3753     0.3735
     0.3467     0.3271     0.3179
    -0.1954    -0.2100    -0.3488

Xnew =

     1
     0

    PredictOutput = predict_henv(ModelOutput, Xnew, 'estimation')
    PredictOutput.value %This is the 3rd group mean
    PredictOutput.SE

PredictOutput =

    value: [8x1 double]
    covMatrix: [8x8 double]
    SE: [8x1 double]

ans =

    -1.0845
    -1.3132
    -1.2152
    -0.1736
     0.3072
     0.3735
     0.3179
    -0.3488

```

```
ans =
```

```
0.0682  
0.0695  
0.0651  
0.0436  
0.0832  
0.0636  
0.0847  
0.0698
```

```
PredictOutput = predict_henv(ModelOutput, Xnew, 'prediction')  
PredictOutput.SE
```

```
PredictOutput =
```

```
value: [8x1 double]  
covMatrix: [8x8 double]  
SE: [8x1 double]
```

```
ans =
```

```
0.3720  
0.3812  
0.3581  
0.2398  
0.4612  
0.3519  
0.4710  
0.3854
```

5.9 testcoefficient_henv

This function tests the null hypothesis $L * \beta * R = A$ versus the alternative hypothesis $L * \beta * R \neq A$, where β is estimated under the heteroscedastic envelope model.

Contents

- Syntax
- Input
- Output
- Description
- Example

Syntax

```
TestOutput = testcoefficient_henv(ModelOutput)
TestOutput = testcoefficient_henv(ModelOutput, TestInput)
```

Input

ModelOutput: A list containing the maximum likelihood estimators and other statistics inherited from henv.

TestInput: A list that specifies the null hypothesis, including L, R, and A. If not provided by the user, default values will be used.

- TestInput.L: The matrix multiplied to β on the left. It is a d1 by r matrix, while d1 is less than or equal to r - 1. Default value: identity matrix I_r .
- TestInput.R: The matrix multiplied to β on the right. It is a p by d2 matrix, while d2 is less than or equal to p. Default value: identity matrix $(I_{p-1}, 0_{(p-1) \times 1})^T$. This is because the columns of β sum to 0. Then we cannot use I_p as default.
- TestInput.A: The matrix on the right hand side of the equation. It is a d1 by d2 matrix. Default value: d1 by d2 zero matrix.

Output

TestOutput: A list containing test statistics, degrees of freedom for the reference chi-squared distribution, the p-value, and the covariance matrix of $\text{vec}(L\beta R)$. At the same time, a table is printed out.

- TestOutput.chisqStatistic: The test statistics. A real number.
- TestOutput.df: The degrees of freedom of the reference chi-squared distribution. A positive integer.
- TestOutput.pValue: p-value of the test. A real number in [0, 1].
- TestOutput.covMatrix: The covariance matrix of $\text{vec}(L\beta R)$. A d1 * d2 by d1 * d2 matrix.

Description

This function tests for hypothesis $H_0 : L\beta R = A$, versus $H_a : L\beta R \neq A$. The β is estimated by the heteroscedastic envelope model. If the user does not specify the values for L, R and A, then the test is equivalent to the standard F test on if all the main group effects are 0. The test statistics used is $\text{vec}(L\beta R - A) \hat{\Sigma}^{-1} \text{vec}(L\beta R - A)^T$, and the reference distribution is chi-squared distribution with degrees of freedom d1 * d2.

Example

```
load waterstrider.mat
u = lrt_henv(X, Y, 0.01);
ModelOutput = henv(X, Y, u);
TestOutout = testcoefficient_henv(ModelOutput);
```

Test Hypothesis	Chisq Statistic	DF	P-value
L * beta * R = A	226.256	16	0.0000

```
r = size(Y, 2);
p = size(ModelOutput.beta, 2);
TestInput.L = rand(2, r);
TestInput.R = rand(p, 1);
TestInput.A = zeros(2, 1);
TestOutout = testcoefficient_henv(ModelOutput, TestInput);
```

Test Hypothesis	Chisq Statistic	DF	P-value
L * beta * R = A	23.429	2	0.0000

6.1 aic_ienv

Select the dimension of the inner envelope subspace using Akaike information criterion.

Contents

- Syntax
- Input
- Output
- Description
- Example

Syntax

```
u = aic_ienv(X, Y)
u = aic_ienv(X, Y, Opts)
```

Input

X: Predictors. An n by p matrix, p is the number of predictors and n is the number of observations. The predictors can be univariate or multivariate, discrete or continuous.

Y: Multivariate responses. An n by r matrix, r is the number of responses. The responses must be continuous variables.

Opts: A list containing the optional input parameters, to control the iterations in `sg_min`. If one or several (even all) fields are not defined, the default settings are used.

- `Opts.maxIter`: Maximum number of iterations. Default value: 300.
- `Opts.ftol`: Tolerance parameter for F. Default value: $1e-10$.
- `Opts.gradtol`: Tolerance parameter for dF. Default value: $1e-7$.
- `Opts.verbose`: Flag for print out dimension selection process, logical 0 or 1. Default value: 0.

Output

u: Dimension of the inner envelope. An integer between 0 and p or equal to r .

Description

This function implements the Akaike information criteria (AIC) to select the dimension of the inner envelope subspace.

Example

```
load irisf.mat
u = aic_ienv(X, Y)
```

u =

1

6.2 bic_ienv

Select the dimension of the inner envelope subspace using Bayesian information criterion.

Contents

- Syntax
- Input
- Output
- Description
- Example

Syntax

```
u = bic_ienv(X, Y)
u = bic_ienv(X, Y, Opts)
```

Input

X: Predictors. An n by p matrix, p is the number of predictors and n is the number of observations. The predictors can be univariate or multivariate, discrete or continuous.

Y: Multivariate responses. An n by r matrix, r is the number of responses. The responses must be continuous variables.

Opts: A list containing the optional input parameters, to control the iterations in sg_min. If one or several (even all) fields are not defined, the default settings are used.

- Opts.maxIter: Maximum number of iterations. Default value: 300.
- Opts.ftol: Tolerance parameter for F. Default value: 1e-10.
- Opts.gradtol: Tolerance parameter for dF. Default value: 1e-7.
- Opts.verbose: Flag for print out dimension selection process, logical 0 or 1. Default value: 0.

Output

u: Dimension of the inner envelope. An integer between 0 and p or equal to r.

Description

This function implements the Bayesian information criteria (BIC) to select the dimension of the inner envelope subspace.

Example

```
load irisf.mat
u = bic_ienv(X, Y)
```

u =

1

6.3 bstrp_ienv

Compute bootstrap standard error for the inner envelope model.

Contents

- Syntax
- Input
- Output
- Description
- Example

Syntax

```
bootse = bstrp_ienv(X, Y, u, B)
bootse = bstrp_ienv(X, Y, u, B, Opts)
```

Input

X: Predictors, an n by p matrix, p is the number of predictors. The predictors can be univariate or multivariate, discrete or continuous.

Y: Multivariate responses, an n by r matrix, r is the number of responses and n is number of observations. The responses must be continuous variables.

u: Dimension of the inner envelope. An integer between 0 and p or equal to r .

B: Number of bootstrap samples. A positive integer.

Opts: A list containing the optional input parameters, to control the iterations in `sg_min`. If one or several (even all) fields are not defined, the default settings are used.

- `Opts.maxIter`: Maximum number of iterations. Default value: 300.
- `Opts.ftol`: Tolerance parameter for F. Default value: $1e-10$.
- `Opts.gradtol`: Tolerance parameter for dF. Default value: $1e-7$.
- `Opts.verbose`: Flag for print out the number of bootstrap samples, logical 0 or 1. Default value: 0.

Output

bootse: The standard error for elements in β computed by bootstrap. An r by p matrix.

Description

This function computes the bootstrap standard errors for the regression coefficients in the inner envelope model by bootstrapping the residuals.

Example

```
load irisf.mat
u = bic_ienv(X, Y)
```

```
u =
```

```
1
```

```
B = 100;  
bootse = bstrp_ienv(X, Y, u, B)
```

```
bootse =
```

```
13.4695    4.9601  
 7.4709    2.7315  
14.9316    5.2913  
 8.7597    3.0853
```

6.4 dF4ienv

First derivative of the objective function for computing the inner envelope subspace.

Contents

- Syntax
- Input
- Output
- Description

Syntax

```
df = dF4ienv(R, DataParameter)
```

Input

R: An r by u semi-orthogonal matrix, $0 < u \leq p$.

DataParameter: A structure that contains the statistics calculated from the data.

Output

df: The first derivative of the objective function for computing the inner envelope subspace. An r by u matrix.

Description

This first derivative of F4ienv obtained by matrix calculus calculations.

6.5 F4ienv

Objective function for computing the inner envelope subspace.

Contents

- Syntax
- Input
- Output
- Description

Syntax

```
f = F4ienv(R, DataParameter)
```

Input

R: An r by u semi orthogonal matrix, $0 < u \leq p$.

DataParameter: A structure that contains the statistics calculated from the data.

Output

f: A scalar containing the value of the objective function evaluated at R .

Description

The objective function is derived in Section 3.3 in Su and Cook (2012) by using maximum likelihood estimation. The columns of the semi-orthogonal matrix that minimizes this function span the estimated inner envelope subspace.

6.6 ienv

Fit the inner envelope model.

Contents

- Syntax
- Input
- Output
- Description
- References
- Example

Syntax

```
ModelOutput = ienv(X, Y, u)
ModelOutput = ienv(X, Y, u, Opts)
```

Input

X: Predictors. An n by p matrix, p is the number of predictors. The predictors can be univariate or multivariate, discrete or continuous.

Y: Multivariate responses. An n by r matrix, r is the number of responses and n is number of observations. The responses must be continuous variables, and r should be strictly greater than p .

u: Dimension of the inner envelope. An integer between 0 and p or equal to r .

Opts: A list containing the optional input parameters, to control the iterations in `sg_min`. If one or several (even all) fields are not defined, the default settings are used.

- `Opts.maxIter`: Maximum number of iterations. Default value: 300.
- `Opts.ftol`: Tolerance parameter for F. Default value: $1e-10$.
- `Opts.gradtol`: Tolerance parameter for dF. Default value: $1e-7$.
- `Opts.verbose`: Flag for print out Grassmann manifold optimization process, logical 0 or 1. Default value: 0.
- `Opts.init`: The initial value for the inner envelope subspace. An r by u matrix. Default value is the one generated by function `get_Init`.

Output

ModelOutput: A list that contains the maximum likelihood estimators and some statistics.

- `ModelOutput.beta`: The envelope estimator of the regression coefficients β . An r by p matrix.
- `ModelOutput.Sigma`: The envelope estimator of the error covariance matrix. An r by r matrix.
- `ModelOutput.Gamma1`: The orthogonal basis of the inner envelope subspace. An r by u semi-orthogonal matrix.
- `ModelOutput.Gamma0`: The orthogonal basis of the complement of the inner envelope subspace. An r by $r-u$ semi-orthogonal matrix.

- `ModelOutput.eta1`: The transpose of the coordinates of β with respect to Γ_1 . An p by u matrix.
- `ModelOutput.B`: An $(r - u)$ by $(p - u)$ semi-orthogonal matrix, so that $(\Gamma_1, \Gamma_0 * B)$ spans β .
- `ModelOutput.eta2`: The transpose of the coordinates of β with respect to Γ_0 . An p by $(p - u)$ matrix.
- `ModelOutput.Omega1`: The coordinates of Σ with respect to Γ_1 . An u by u matrix.
- `ModelOutput.Omega0`: The coordinates of Σ with respect to Γ_0 . An $(r - u)$ by $(r - u)$ matrix.
- `ModelOutput.alpha`: The estimated intercept in the inner envelope model. An r by 1 vector.
- `ModelOutput.l`: The maximized log likelihood function. A real number.
- `ModelOutput.covMatrix`: The asymptotic covariance of $\text{vec}(\beta)$. An rp by rp matrix. The covariance matrix returned are asymptotic. For the actual standard errors, multiply by $1/n$.
- `ModelOutput.asySE`: Asymptotic standard error for elements in β under the inner envelope model. An r by p matrix. The standard errors returned are asymptotic, for actual standard errors, multiply by $1 / \sqrt{n}$.
- `ModelOutput.ratio`: The asymptotic standard error ratio of the standard multivariate linear regression estimator over the inner envelope estimator, for each element in β . An r by p matrix.
- `ModelOutput.paramNum`: The number of parameters in the inner envelope model. A positive integer.
- `ModelOutput.n`: The number of observations in the data. A positive integer.

Description

This function fits the inner envelope model to the responses and predictors, using the maximum likelihood estimation. When the dimension of the envelope is between 1 and $p-1$, we implemented the algorithm in Su and Cook (2012). When the dimension is p , then the inner envelope model degenerates to the standard multivariate linear regression. When the dimension is 0, it means that X and Y are uncorrelated, and the fitting is different.

References

1. The codes are implemented based on the algorithm in Su and Cook (2012).
2. The Grassmann manifold optimization step calls the package `sg_min` 2.4.3 by Ross Lipert (<http://web.mit.edu/~ripper/www/sgmin.html>).

Example

The following codes gives the results of the Fisher's iris data example in Su and Cook (2012).

```
load irisf.mat

d = bic_ienv(X, Y)
```

`d =`

```

ModelOutput = ienv(X, Y, d)
1 - 1 ./ ModelOutput.ratio

```

```

ModelOutput =

```

```

    beta: [4x2 double]
    Sigma: [4x4 double]
    Gamma1: [4x1 double]
    Gamma0: [4x3 double]
    B: [3x1 double]
    eta1: [2x1 double]
    eta2: [2x1 double]
    Omega1: 8.3751
    Omega0: [3x3 double]
    alpha: [4x1 double]
    paramNum: 16
    l: -1.4805e+03
    covMatrix: [8x8 double]
    asySE: [4x2 double]
    ratio: [4x2 double]
    n: 150

```

```

ans =

```

```

    0.0049    0.2122
    0.0020    0.0806
    0.0033    0.1404
    0.0004    0.0138

```

6.7 lrt_ienv

Select the dimension of the inner envelope subspace using likelihood ratio testing.

Contents

- Syntax
- Input
- Output
- Description
- Example

Syntax

```
u = lrt_ienv(X, Y, alpha)
u = lrt_ienv(X, Y, alpha, Opts)
```

Input

X: Predictors. An n by p matrix, p is the number of predictors. The predictors can be univariate or multivariate, discrete or continuous.

Y: Multivariate responses. An n by r matrix, r is the number of responses and n is number of observations. The responses must be continuous variables.

alpha: Significance level for testing. A real number between 0 and 1, often taken at 0.05 or 0.01.

Opts: A list containing the optional input parameters, to control the iterations in `sg_min`. If one or several (even all) fields are not defined, the default settings are used.

- `Opts.maxIter`: Maximum number of iterations. Default value: 300.
- `Opts.ftol`: Tolerance parameter for F. Default value: 1e-10.
- `Opts.gradtol`: Tolerance parameter for dF. Default value: 1e-7.
- `Opts.verbose`: Flag for print out dimension selection process, logical 0 or 1. Default value: 0.

Output

u: Dimension of the inner envelope. An integer between 0 and p or equal to r.

Description

This function implements the likelihood ratio testing procedure to select the dimension of the inner envelope subspace, with pre-specified significance level α .

Example

```
load irisf.mat

alpha = 0.01;
u = lrt_ienv(X, Y, alpha)
```

u =

1

6.8 predict_ienv

Perform estimation or prediction under the inner envelope model.

Contents

- Syntax
- Input
- Output
- Description
- Example

Syntax

```
PredictOutput = predict_ienv(ModelOutput, Xnew, infType)
```

Input

ModelOutput: A list containing the maximum likelihood estimators and other statistics inherited from ienv.

Xnew: The value of X with which to estimate or predict Y. A p by 1 vector.

infType: A string of characters indicating the inference type, the choices can be 'estimation' or 'prediction'.

Output

PredictOutput: A list containing the results of the inference.

- PredictOutput.value: The fitted value or the prediction value evaluated at Xnew. An r by 1 vector.
- PredictOutput.covMatrix: The covariance matrix of PredictOutput.value. An r by r matrix.
- PredictOutput.SE: The standard error of elements in PredictOutput.value. An r by 1 vector.

Description

This function evaluates the inner envelope model at new value Xnew. It can perform estimation: find the fitted value when $X = X_{\text{new}}$, or prediction: predict Y when $X = X_{\text{new}}$. The covariance matrix and the standard errors are also provided.

Example

```
load irisf.mat
d = bic_ienv(X, Y);
ModelOutput = ienv(X, Y, d);
Xnew = X(1, :)';
PredictOutput = predict_ienv(ModelOutput, Xnew, 'estimation')
[PredictOutput.value, Y(1, :)] % Compare the fitted value with the data
PredictOutput.SE
```

```
PredictOutput =
```

```
    value: [4x1 double]
  covMatrix: [4x4 double]
        SE: [4x1 double]
```

```
ans =
```

```
49.9458    51.0000
34.2592    35.0000
14.5771    14.0000
 2.4513     2.0000
```

```
ans =
```

```
1.0978
0.7146
0.9265
0.4357
```

```
PredictOutput = predict_ienv(ModelOutput, Xnew, 'prediction')
PredictOutput.SE
```

```
PredictOutput =
```

```
    value: [4x1 double]
  covMatrix: [4x4 double]
        SE: [4x1 double]
```

```
ans =
```

```
5.2197
3.3897
4.3996
2.0642
```

6.9 testcoefficient_ienv

This function tests the null hypothesis $L * \beta * R = A$ versus the alternative hypothesis $L * \beta * R \neq A$, where β is estimated under the inner envelope model.

Contents

- Syntax
- Input
- Output
- Description
- Example

Syntax

```
TestOutput = testcoefficient_ienv(ModelOutput)
TestOutput = testcoefficient_ienv(ModelOutput, TestInput)
```

Input

ModelOutput: A list containing the maximum likelihood estimators and other statistics inherited from ienv.

TestInput: A list that specifies the null hypothesis, including L, R, and A. If not provided by the user, default values will be used.

- TestInput.L: The matrix multiplied to β on the left. It is a d1 by r matrix, while d1 is less than or equal to r. Default value: identity matrix I_r .
- TestInput.R: The matrix multiplied to β on the right. It is a p by d2 matrix, while d2 is less than or equal to p. Default value: identity matrix I_p .
- TestInput.A: The matrix on the right hand side of the equation. It is a d1 by d2 matrix. Default value: d1 by d2 zero matrix.

Output

TestOutput: A list containing test statistics, degrees of freedom for the reference chi-squared distribution, the p-value, and the covariance matrix of $\text{vec}(L\beta R)$. At the same time, a table is printed out.

- TestOutput.chisqStatistic: The test statistics. A real number.
- TestOutput.df: The degrees of freedom of the reference chi-squared distribution. A positive integer.
- TestOutput.pValue: p-value of the test. A real number in [0, 1].
- TestOutput.covMatrix: The covariance matrix of $\text{vec}(L\beta R)$. A d1 * d2 by d1 * d2 matrix.

Description

This function tests for hypothesis $H_0 : L\beta R = A$, versus $H_\alpha : L\beta R \neq A$. The β is estimated by the inner envelope model. If the user does not specify the values for L, R and A, then the test is equivalent to the standard F test on if $\beta = 0$. The test statistics used is $\text{vec}(L\beta R - A) \hat{\Sigma}^{-1} \text{vec}(L\beta R - A)^T$, and the reference distribution is chi-squared distribution with degrees of freedom d1 * d2.

Example

```
load irisf.mat
d = bic_ienv(X,Y);
ModelOutput = ienv(X,Y,d);
TestOutout = testcoefficient_ienv(ModelOutput);
```

Test Hypothesis	Chisq Statistic	DF	P-value
L * beta * R = A	4642.913	8	0.0000

```
TestInput.L = rand(2, 4);
TestInput.R = rand(2, 1);
TestInput.A = zeros(2, 1);
TestOutout = testcoefficient_ienv(ModelOutput, TestInput);
```

Test Hypothesis	Chisq Statistic	DF	P-value
L * beta * R = A	1834.229	2	0.0000

7.1 aic_penv

Select the dimension of the partial envelope subspace using Akaike information criterion.

Contents

- Syntax
- Input
- Output
- Description
- Example

Syntax

```
u = aic_penv(X, Y)
u = aic_penv(X, Y, Opts)
```

Input

X: A list containing the value of X1 and X2.

- **X.X1:** Predictors of main interest. An n by $p1$ matrix, n is the number of observations, and $p1$ is the number of main predictors. The predictors can be univariate or multivariate, discrete or continuous.
- **X.X2:** Covariates, or predictors not of main interest. An n by $p2$ matrix, $p2$ is the number of covariates.

Y: Multivariate responses. An n by r matrix, r is the number of responses and n is number of observations. The responses must be continuous variables.

Opts: A list containing the optional input parameters, to control the iterations in `sg_min`. If one or several (even all) fields are not defined, the default settings are used.

- **Opts.maxIter:** Maximum number of iterations. Default value: 300.
- **Opts.ftol:** Tolerance parameter for F. Default value: $1e-10$.
- **Opts.gradtol:** Tolerance parameter for dF. Default value: $1e-7$.
- **Opts.verbose:** Flag for print out dimension selection process, logical 0 or 1. Default value: 0.

Output

u: Dimension of the envelope. An integer between 0 and r .

Description

This function implements the Akaike information criteria (AIC) to select the dimension of the partial envelope subspace.

Example

```
load fiberpaper.dat
Y = fiberpaper(:, 1 : 4);
X.X1 = fiberpaper(:, 7);
X.X2 = fiberpaper(:, 5 : 6);
u = aic_penv(X, Y)
```

u =

3

7.2 bic_penv

Select the dimension of the partial envelope subspace using Bayesian information criterion.

Contents

- Syntax
- Input
- Output
- Description
- Example

Syntax

```
u = bic_penv(X, Y)
u = bic_penv(X, Y, Opts)
```

Input

X: A list containing the value of X1 and X2.

- X.X1: Predictors of main interest. An n by p1 matrix, n is the number of observations, and p1 is the number of main predictors. The predictors can be univariate or multivariate, discrete or continuous.
- X.X2: Covariates, or predictors not of main interest. An n by p2 matrix, p2 is the number of covariates.

Y: Multivariate responses. An n by r matrix, r is the number of responses and n is number of observations. The responses must be continuous variables.

Opts: A list containing the optional input parameters, to control the iterations in sg_min. If one or several (even all) fields are not defined, the default settings are used.

- Opts.maxIter: Maximum number of iterations. Default value: 300.
- Opts.ftol: Tolerance parameter for F. Default value: 1e-10.
- Opts.gradtol: Tolerance parameter for dF. Default value: 1e-7.
- Opts.verbose: Flag for print out dimension selection process, logical 0 or 1. Default value: 0.

Output

u: Dimension of the envelope. An integer between 0 and r.

Description

This function implements the Bayesian information criteria (BIC) to select the dimension of the partial envelope subspace.

Example

```
load fiberpaper.dat
Y = fiberpaper(:, 1 : 4);
X.X1 = fiberpaper(:, 7);
X.X2 = fiberpaper(:, 5 : 6);
u = bic_penv(X, Y)
```

u =

1

7.3 bstrp_penv

Compute bootstrap standard error for the partial envelope model.

Contents

- Syntax
- Input
- Output
- Description
- Example

Syntax

```
bootse = bstrp_penv(X, Y, u, B)
bootse = bstrp_penv(X, Y, u, B, Opts)
```

Input

X: A list containing the value of X1 and X2.

- X.X1: Predictors of main interest. An n by p1 matrix, n is the number of observations, and p1 is the number of main predictors. The predictors can be univariate or multivariate, discrete or continuous.
- X.X2: Covariates, or predictors not of main interest. An n by p2 matrix, p2 is the number of covariates.

Y: Multivariate responses, an n by r matrix, r is the number of responses and n is number of observations. The responses must be continuous variables.

u: Dimension of the partial envelope subspace. A positive integer between 0 and r.

B: Number of bootstrap samples. A positive integer.

Opts: A list containing the optional input parameters, to control the iterations in `sg_min`. If one or several (even all) fields are not defined, the default settings are used.

- Opts.maxIter: Maximum number of iterations. Default value: 300.
- Opts.ftol: Tolerance parameter for F. Default value: 1e-10.
- Opts.gradtol: Tolerance parameter for dF. Default value: 1e-7.
- Opts.verbose: Flag for print out the number of bootstrap samples, logical 0 or 1. Default value: 0.

Output

bootse: The standard error for elements in β_1 computed by bootstrap. An r by p1 matrix.

Description

This function computes the bootstrap standard errors for the regression coefficients in the partial envelope model by bootstrapping the residuals.

Example

```
load fiberpaper.dat
Y = fiberpaper(:, 1 : 4);
X.X1 = fiberpaper(:, 7);
X.X2 = fiberpaper(:, 5 : 6);
alpha = 0.01;
u = modelselectlrt(X, Y, alpha, 'penv')
```

u =

1

```
B = 100;
bootse = bstrp_penv(X, Y, u, B)
```

bootse =

```
0.0074
0.0021
0.0043
0.0019
```

7.4 lrt_penv

Select the dimension of the partial envelope subspace using likelihood ratio testing.

Contents

- Syntax
- Input
- Output
- Description
- Example

Syntax

```
u = lrt_penv(X, Y, alpha)
u = lrt_penv(X, Y, alpha, Opts)
```

Input

X: A list containing the value of X1 and X2.

- X.X1: Predictors of main interest. An n by p1 matrix, n is the number of observations, and p1 is the number of main predictors. The predictors can be univariate or multivariate, discrete or continuous.
- X.X2: Covariates, or predictors not of main interest. An n by p2 matrix, p2 is the number of covariates.

Y: Multivariate responses. An n by r matrix, r is the number of responses and n is number of observations. The responses must be continuous variables.

alpha: Significance level for testing. A real number between 0 and 1, often taken at 0.05 or 0.01.

Opts: A list containing the optional input parameters, to control the iterations in sg_min. If one or several (even all) fields are not defined, the default settings are used.

- Opts.maxIter: Maximum number of iterations. Default value: 300.
- Opts.ftol: Tolerance parameter for F. Default value: 1e-10.
- Opts.gradtol: Tolerance parameter for dF. Default value: 1e-7.
- Opts.verbose: Flag for print out dimension selection process, logical 0 or 1. Default value: 0.

Output

u: Dimension of the partial envelope subspace. An integer between 0 and r.

Description

This function implements the likelihood ratio testing procedure to select the dimension of the partial envelope subspace, with pre-specified significance level α .

Example

```
load fiberpaper.dat
Y = fiberpaper(:, 1 : 4);
X.X1 = fiberpaper(:, 7);
X.X2 = fiberpaper(:, 5 : 6);
alpha = 0.01;
u = lrt_penv(X, Y, alpha)
```

u =

1

7.5 penv

Fit the partial envelope model.

Contents

- Syntax
- Input
- Output
- Description
- References
- Example

Syntax

```
ModelOutput = penv(X, Y, u)
ModelOutput = penv(X, Y, u, Opts)
```

Input

X: A list containing the value of X1 and X2.

- X.X1: Predictors of main interest. An n by p1 matrix, n is the number of observations, and p1 is the number of main predictors. The predictors can be univariate or multivariate, discrete or continuous.
- X.X2: Covariates, or predictors not of main interest. An n by p2 matrix, p2 is the number of covariates.

Y: Multivariate responses. An n by r matrix, r is the number of responses and n is number of observations. The responses must be continuous variables, and r should be strictly greater than p1.

u: Dimension of the partial envelope. An integer between 0 and r.

Opts: A list containing the optional input parameters, to control the iterations in sg_min. If one or several (even all) fields are not defined, the default settings are used.

- Opts.maxIter: Maximum number of iterations. Default value: 300.
- Opts.ftol: Tolerance parameter for F. Default value: 1e-10.
- Opts.gradtol: Tolerance parameter for dF. Default value: 1e-7.
- Opts.verbose: Flag for print out Grassmann manifold optimization process, logical 0 or 1. Default value: 0.
- Opts.init: The initial value for the partial envelope subspace. An r by u matrix. Default value is the one generated by function get_Init.

Output

ModelOutput: A list that contains the maximum likelihood estimators and some statistics.

- ModelOutput.beta1: The partial envelope estimator of β_1 , which is the regression coefficients for X1. An r by p1 matrix.
- ModelOutput.beta2: The partial envelope estimator of β_2 , which is the regression coefficients for X2. An r by p2 matrix.

- **ModelOutput.Sigma**: The partial envelope estimator of the error covariance matrix. An r by r matrix.
- **ModelOutput.Gamma**: The orthogonal basis of the partial envelope subspace. An r by u semi-orthogonal matrix.
- **ModelOutput.Gamma0**: The orthogonal basis of the complement of the partial envelope subspace. An r by $r - u$ semi-orthogonal matrix.
- **ModelOutput.eta**: The coordinates of β_1 with respect to Gamma. An u by p_1 matrix.
- **ModelOutput.Omega**: The coordinates of Sigma with respect to Gamma. An u by u matrix.
- **ModelOutput.Omega0**: The coordinates of Sigma with respect to Gamma0. An $r - u$ by $r - u$ matrix.
- **ModelOutput.alpha**: The estimated intercept in the partial envelope model. An r by 1 vector.
- **ModelOutput.l**: The maximized log likelihood function. A real number.
- **ModelOutput.covMatrix**: The asymptotic covariance of $(\text{vec}(\beta_2)', \text{vec}(\beta_1)')'$. An r_p by r_p matrix. The covariance matrix returned are asymptotic. For the actual standard errors, multiply by $1/n$.
- **ModelOutput.asyPenv**: Asymptotic standard error for elements in β_1 under the partial envelope model. An r by p_1 matrix. The standard errors returned are asymptotic, for actual standard errors, multiply by $1/\sqrt{n}$.
- **ModelOutput.ratio**: The asymptotic standard error ratio of the standard multivariate linear regression estimator over the partial envelope estimator, for each element in β_1 . An r by p_1 matrix.
- **ModelOutput.paramNum**: The number of parameters in the envelope model. A positive integer.
- **ModelOutput.n**: The number of observations in the data. A positive integer.

Description

This function fits the partial envelope model to the responses Y and predictors X_1 and X_2 , using the maximum likelihood estimation. When the dimension of the envelope is between 1 and $r - 1$, we implemented the algorithm in Su and Cook (2011). When the dimension is r , then the partial envelope model degenerates to the standard multivariate linear regression with Y as the responses and both X_1 and X_2 as predictors. When the dimension is 0, X_1 and Y are uncorrelated, and the fitting is the standard multivariate linear regression with Y as the responses and X_2 as the predictors.

References

1. The codes are implemented based on the algorithm in Section 3.2 of Su and Cook (2012).
2. The Grassmann manifold optimization step calls the package `sg_min` 2.4.3 by Ross Lipert (<http://web.mit.edu/~ripper/www/sgmin.html>).

Example

The following codes reconstruct the results of the paper and fiber example in Su and Cook (2012).

```
load fiberpaper.dat
Y = fiberpaper(:, 1 : 4);
X.X1 = fiberpaper(:, 7);
X.X2 = fiberpaper(:, 5 : 6);
```

```
alpha = 0.01;
u = modelselectlrt(X, Y, alpha, 'penv')
```

```
u =
```

```
1
```

```
ModelOutput = penv(X, Y, u)
ModelOutput.Omega
eig(ModelOutput.Omega0)
ModelOutput.ratio
```

```
ModelOutput =
```

```
    beta1: [4x1 double]
    beta2: [4x2 double]
    alpha: [4x1 double]
    Gamma: [4x1 double]
    eta: 0.0047
    Omega: 0.0149
    Omega0: [3x3 double]
    Sigma: [4x4 double]
    l: -35.6323
    paramNum: 23
    covMatrix: [12x12 double]
    asyPenv: [4x1 double]
    ratio: [4x1 double]
    n: 62
```

```
ans =
```

```
0.0149
```

```
ans =
```

```
4.9819
0.0999
0.0050
```

```
ans =
```

```
66.0742
6.9326
10.5048
9.6279
```

7.6 predict_penv

Perform estimation or prediction under the partial envelope model.

Contents

- Syntax
- Input
- Output
- Description
- Example

Syntax

```
PredictOutput = predict_penv(ModelOutput, Xnew, infType)
```

Input

ModelOutput: A list containing the maximum likelihood estimators and other statistics inherited from `penv`.

Xnew: A list containing the value of X_1 and X_2 with which to estimate or predict Y .

- * `Xnew.X1`: A p_1 by 1 vector containing the value of X_1 .
- * `Xnew.X2`: A p_2 by 1 vector containing the value of X_2 .

infType: A string of characters indicting the inference type, the choices can be 'estimation' or 'prediction'.

Output

PredictOutput: A list containing the results of the inference.

- `PredictOutput.value`: The fitted value or the prediction value evaluated at `Xnew`. An r by 1 vector.
- `PredictOutput.covMatrix`: The covariance matrix of `PredictOutput.value`. An r by r matrix.
- `PredictOutput.SE`: The standard error of elements in `PredictOutput.value`. An r by 1 vector.

Description

This function evaluates the envelope model at new value `Xnew`. It can perform estimation: find the fitted value when $X = Xnew$, or prediction: predict Y when $X = Xnew$. The covariance matrix and the standard errors are also provided.

Example

```

load fiberpaper.dat
Y = fiberpaper(:, 1 : 4);
X.X1 = fiberpaper(:, 7);
X.X2 = fiberpaper(:, 5 : 6);
alpha = 0.01;
u = modelselectlrt(X, Y, alpha, 'penv');
ModelOutput = penv(X, Y, u);
Xnew.X1 = X.X1(1, :)';
Xnew.X2 = X.X2(1, :)';
PredictOutput = predict_penv(ModelOutput, Xnew, 'estimation')
[PredictOutput.value, Y(1, :)] % Compare the fitted value with the data
PredictOutput.SE

```

```
PredictOutput =
```

```

    value: [4x1 double]
 covMatrix: [4x4 double]
        SE: [4x1 double]

```

```
ans =
```

```

21.1169    21.3120
 7.1173     7.0390
 5.3637     5.3260
 0.8737     0.9320

```

```
ans =
```

```

1.4680
0.4234
0.7145
0.3161

```

```

PredictOutput = predict_penv(ModelOutput, Xnew, 'prediction')
PredictOutput.SE

```

```
PredictOutput =
```

```

    value: [4x1 double]
 covMatrix: [4x4 double]
        SE: [4x1 double]

```

```
ans =
```

```
2.4277
```

0.6982

1.1802

0.5220

7.7 testcoefficient_penv

This function tests the null hypothesis $L * \beta_1 * R = A$ versus the alternative hypothesis $L * \beta_1 * R \neq A$, where β_1 is estimated under the envelope model.

Contents

- Syntax
- Input
- Output
- Description
- Example

Syntax

```
TestOutput = testcoefficient_penv(ModelOutput)
TestOutput = testcoefficient_penv(ModelOutput, TestInput)
```

Input

ModelOutput: A list containing the maximum likelihood estimators and other statistics inherited from `penv`.

TestInput: A list that specifies the null hypothesis, including L , R , and A . If not provided by the user, default values will be used.

- **TestInput.L:** The matrix multiplied to β_1 on the left. It is a d_1 by r matrix, while d_1 is less than or equal to r . Default value: identity matrix I_r .
- **TestInput.R:** The matrix multiplied to β_1 on the right. It is a p_1 by d_2 matrix, while d_2 is less than or equal to p_1 . Default value: identity matrix I_{p_1} .
- **TestInput.A:** The matrix on the right hand side of the equation. It is a d_1 by d_2 matrix. Default value: d_1 by d_2 zero matrix.

Output

TestOutput: A list containing test statistics, degrees of freedom for the reference chi-squared distribution, the p-value, and the covariance matrix of $\text{vec}(L\beta_1 R)$. At the same time, a table is printed out.

- **TestOutput.chisqStatistic:** The test statistics. A real number.
- **TestOutput.df:** The degrees of freedom of the reference chi-squared distribution. A positive integer.
- **TestOutput.pValue:** p-value of the test. A real number in $[0, 1]$.
- **TestOutput.covMatrix:** The covariance matrix of $\text{vec}(L\beta_1 R)$. A $d_1 * d_2$ by $d_1 * d_2$ matrix.

Description

This function tests for hypothesis $H_0 : L\beta_1 R = A$, versus $H_a : L\beta_1 R \neq A$. The β_1 is estimated by the partial envelope model. If the user does not specify the values for L , R and A , then the test is equivalent to the standard F test on if $\beta_1 = 0$. The test statistics used is $\text{vec}(L\beta_1 R - A)^T \hat{\Sigma}^{-1} \text{vec}(L\beta_1 R - A)$, and the reference distribution is chi-squared distribution with degrees of freedom $d_1 * d_2$.

Example

```

load fiberpaper.dat
Y = fiberpaper(:, 1 : 4);
X.X1 = fiberpaper(:, 7);
X.X2 = fiberpaper(:, 5 : 6);
alpha = 0.01;
u = modelselectlrt(X, Y, alpha, 'penv');
ModelOutput = penv(X, Y, u);
TestOutout = testcoefficient_penv(ModelOutput);

```

Test Hypothesis	Chisq Statistic	DF	P-value
L * beta * R = A	12.604	4	0.0134

```

r = size(Y, 2);
p1 = size(X.X1, 2);
TestInput.L = rand(2, r);
TestInput.R = rand(p1, 1);
TestInput.A = zeros(2, 1);
TestOutout = testcoefficient_penv(ModelOutput, TestInput);

```

Test Hypothesis	Chisq Statistic	DF	P-value
L * beta * R = A	11.452	2	0.0033

8.1 aic_senv

Select the dimension of the scaled envelope subspace using Akaike information criterion.

Contents

- Syntax
- Input
- Output
- Description
- Example

Syntax

```
u = aic_senv(X, Y)
u = aic_senv(X, Y, Opts)
```

Input

X: Predictors. An n by p matrix, p is the number of predictors and n is the number of observations. The predictors can be univariate or multivariate, discrete or continuous.

Y: Multivariate responses. An n by r matrix, r is the number of responses. The responses must be continuous variables.

Opts: A list containing the optional input parameters, to control the iterations in `sg_min`. If one or several (even all) fields are not defined, the default settings are used.

- `Opts.maxIter`: Maximum number of iterations. Default value: 300.
- `Opts.ftol`: Tolerance parameter for F Default value: $1e-10$.
- `Opts.gradtol`: Tolerance parameter for dF Default value: $1e-7$.
- `Opts.verbose`: Flag for print out dimension selection process, logical 0 or 1. Default value: 0.

Output

u: Dimension of the inner envelope. An integer between 0 and r .

Description

This function implements the Akaike information criteria (AIC) to select the dimension of the scaled envelope subspace.

Example

```
load('sales.txt')
Y = sales(:, 4 : 7);
X = sales(:, 1 : 3);
u = aic_senv(X, Y)
```

u =

4

8.2 **bic_senv**

Select the dimension of the scaled envelope subspace using Bayesian information criterion.

Contents

- Syntax
- Input
- Output
- Description
- Example

Syntax

```
u = bic_senv(X, Y)
u = bic_senv(X, Y, Opts)
```

Input

X: Predictors. An n by p matrix, p is the number of predictors and n is the number of observations. The predictors can be univariate or multivariate, discrete or continuous.

Y: Multivariate responses. An n by r matrix, r is the number of responses. The responses must be continuous variables.

Opts: A list containing the optional input parameters, to control the iterations in `sg_min`. If one or several (even all) fields are not defined, the default settings are used.

- `Opts.maxIter`: Maximum number of iterations. Default value: 300.
- `Opts.ftol`: Tolerance parameter for F. Default value: $1e-10$.
- `Opts.gradtol`: Tolerance parameter for dF. Default value: $1e-7$.
- `Opts.verbose`: Flag for print out dimension selection process, logical 0 or 1. Default value: 0.

Output

u: Dimension of the inner envelope. An integer between 0 and r .

Description

This function implements the Bayesian information criteria (BIC) to select the dimension of the scaled envelope subspace.

Example

```
load('sales.txt')
Y = sales(:, 4 : 7);
X = sales(:, 1 : 3);
u = bic_senv(X, Y)
```

$$u =$$

$$2$$

8.3 bstrp_senv

Compute bootstrap standard error for the scaled envelope model.

Contents

- Syntax
- Input
- Output
- Description
- Example

Syntax

```
bootse = bstrp_senv(X, Y, u, B)
bootse = bstrp_senv(X, Y, u, B, Opts)
```

Input

X: Predictors, an n by p matrix, p is the number of predictors. The predictors can be univariate or multivariate, discrete or continuous.

Y: Multivariate responses, an n by r matrix, r is the number of responses and n is number of observations. The responses must be continuous variables.

u: Dimension of the envelope subspace. A positive integer between 0 and r .

B: Number of bootstrap samples. A positive integer.

Opts: A list containing the optional input parameters, to control the iterations in `sg_min`. If one or several (even all) fields are not defined, the default settings are used.

- `Opts.maxIter`: Maximum number of iterations. Default value: 300.
- `Opts.ftol`: Tolerance parameter for F. Default value: $1e-10$.
- `Opts.gradtol`: Tolerance parameter for dF. Default value: $1e-7$.
- `Opts.verbose`: Flag for print out the number of bootstrap samples, logical 0 or 1. Default value: 0.

Output

bootse: The standard error for elements in β computed by bootstrap. An r by p matrix.

Description

This function computes the bootstrap standard errors for the regression coefficients in the scaled envelope model by bootstrapping the residuals.

Example

```
load('sales.txt')
Y = sales(:, 4 : 7);
X = sales(:, 1 : 3);
```

```
u = bic_senv(X, Y)
```

```
u =
```

```
2
```

```
B = 20;
```

```
bootse = bstrp_senv(X, Y, u, B)
```

```
bootse =
```

0.0539	0.0472	0.0554
0.0675	0.0912	0.1178
0.0567	0.0781	0.0791
0.0986	0.0944	0.1283

8.4 dF4senv

First derivative of the objective function for computing the envelope subspace in the scaled envelope model.

Contents

- Syntax
- Input
- Output
- Description

Syntax

```
df = dF4senv(R, DataParameter)
```

Input

R: An r by u semi-orthogonal matrix, $0 < u \leq r$.

DataParameter: A structure that contains the statistics calculated from the data.

Output

df: The first derivative of the objective function for computing the envelope subspace. An r by u matrix.

Description

This first derivative of F4senv obtained by matrix calculus calculations.

8.5 F4senv

Objective function for computing the envelope subspace in scaled envelope model.

Contents

- Syntax
- Input
- Output
- Description

Syntax

```
f = F4senv(R, DataParameter)
```

Input

R: An r by u semi orthogonal matrix, $0 < u \leq r$.

DataParameter: A structure that contains the statistics calculated from the data.

Output

f: A scalar containing the value of the objective function evaluated at R .

Description

The objective function is derived in Section 4.1 in Cook and Su (2012) using maximum likelihood estimation. The columns of the semi-orthogonal matrix that minimizes this function span the estimated envelope subspace.

8.6 objfun

Objective function for computing the scales in the scaled envelope model.

Contents

- Syntax
- Input
- Output
- Description

Syntax

```
f = objfun(d, Gamma, DataParameter)
```

Input

d: An $r - 1$ dimensional column vector containing the scales for the 2nd to the r th responses. All the entries in d are positive.

Gamma: A r by u semi-orthogonal matrix that spans the envelope subspace or the estimated envelope subspace.

DataParameter: A structure that contains the statistics calculated from the data.

Output

f: A scalar containing the value of the objective function evaluated at d .

Description

The objective function is derived in Section 4.1 of Su and Cook (2012) using maximum likelihood estimation.

8.7 predict_senv

Perform estimation or prediction under the scaled envelope model.

Contents

- Syntax
- Input
- Output
- Description
- Example

Syntax

```
PredictOutput = predict_senv(ModelOutput, Xnew, infType)
```

Input

ModelOutput: A list containing the maximum likelihood estimators and other statistics inherited from `senv`.

Xnew: The value of X with which to estimate or predict Y . A p by 1 vector.

infType: A string of characters indicating the inference type, the choices can be 'estimation' or 'prediction'.

Output

PredictOutput: A list containing the results of the inference.

- `PredictOutput.value`: The fitted value or the prediction value evaluated at X_{new} . An r by 1 vector.
- `PredictOutput.covMatrix`: The covariance matrix of `PredictOutput.value`. An r by r matrix.
- `PredictOutput.SE`: The standard error of elements in `PredictOutput.value`. An r by 1 vector.

Description

This function evaluates the scaled envelope model at new value X_{new} . It can perform estimation: find the fitted value when $X = X_{\text{new}}$, or prediction: predict Y when $X = X_{\text{new}}$. The covariance matrix and the standard errors are also provided.

Example

```
load('sales.txt')
Y = sales(:, 4 : 7);
X = sales(:, 1 : 3);
u = bic_senv(X, Y);
ModelOutput = senv(X, Y, u);
Xnew = X(1, :);
PredictOutput = predict_senv(ModelOutput, Xnew, 'estimation')
```

```
[PredictOutput.value, Y(1, :)] % Compare the fitted value with the data
PredictOutput.SE
```

```
PredictOutput =
```

```
    value: [4x1 double]
 covMatrix: [4x4 double]
        SE: [4x1 double]
```

```
ans =
```

```
    8.9109    9.0000
   11.5096   12.0000
    9.6063    9.0000
   19.5119   20.0000
```

```
ans =
```

```
    7.7627
    5.8952
    4.2205
    8.0134
```

```
PredictOutput = predict_senv(ModelOutput, Xnew, 'prediction')
PredictOutput.SE
```

```
PredictOutput =
```

```
    value: [4x1 double]
 covMatrix: [4x4 double]
        SE: [4x1 double]
```

```
ans =
```

```
    8.3024
    6.3052
    4.4254
    8.5956
```

8.8 **senv**

Fit the scaled envelope model.

Contents

- Syntax
- Input
- Output
- Description
- References
- Example

Syntax

```
ModelOutput = senv(X, Y, u)
ModelOutput = senv(X, Y, u, Opts)
```

Input

X: Predictors. An n by p matrix, p is the number of predictors. The predictors can be univariate or multivariate, discrete or continuous.

Y: Multivariate responses. An n by r matrix, r is the number of responses and n is number of observations. The responses must be continuous variables, and r should be strictly greater than p .

u: Dimension of the envelope. An integer between 0 and r .

Opts: A list containing the optional input parameters, to control the iterations in `sg_min`. If one or several (even all) fields are not defined, the default settings are used.

- `Opts.maxIter`: Maximum number of iterations. Default value: 300.
- `Opts.ftol`: Tolerance parameter for F. Default value: $1e-10$.
- `Opts.gradtol`: Tolerance parameter for dF. Default value: $1e-7$.
- `Opts.verbose`: Flag for print out number of iterations, logical 0 or 1. Default value: 0.
- `Opts.init`: The initial value for the envelope subspace. An r by u matrix. Default value is the one generated by function `get_Init`.

Output

ModelOutput: A list that contains the maximum likelihood estimators and some statistics.

- `ModelOutput.beta`: The scaled envelope estimator of the regression coefficients β . An r by p matrix.
- `ModelOutput.Sigma`: The scaled envelope estimator of the error covariance matrix. An r by r matrix.
- `ModelOutput.Lambda`: The matrix of estimated scales. An r by r diagonal matrix with the first diagonal element equal to 1 and other diagonal elements being positive.
- `ModelOutput.Gamma`: The orthogonal basis of the envelope subspace. An r by u semi-orthogonal matrix.
- `ModelOutput.Gamma0`: The orthogonal basis of the complement of the envelope subspace. An r by $r - u$ semi-orthogonal matrix.

- `ModelOutput.eta`: The coordinates of β with respect to Γ . An u by p matrix.
- `ModelOutput.Omega`: The coordinates of Σ with respect to Γ . An u by u matrix.
- `ModelOutput.Omega0`: The coordinates of Σ with respect to Γ_0 . An $r - u$ by $r - u$ matrix.
- `ModelOutput.alpha`: The estimated intercept in the scaled envelope model. An r by 1 vector.
- `ModelOutput.l`: The maximized log likelihood function. A real number.
- `ModelOutput.covMatrix`: The asymptotic covariance of $\text{vec}(\beta)$. An rp by rp matrix. The covariance matrix returned are asymptotic. For the actual standard errors, multiply by $1 / n$.
- `ModelOutput.asySenv`: Asymptotic standard error for elements in β under the scaled envelope model. An r by p matrix. The standard errors returned are asymptotic, for actual standard errors, multiply by $1 / \sqrt{n}$.
- `ModelOutput.ratio`: The asymptotic standard error ratio of the standard multivariate linear regression estimator over the scaled envelope estimator, for each element in β . An r by p matrix.
- `ModelOutput.paramNum`: The number of parameters in the scaled envelope model. A positive integer.
- `ModelOutput.n`: The number of observations in the data. A positive integer.

Description

This function fits the scaled envelope model to the responses and predictors, using the maximum likelihood estimation. When the dimension of the envelope is between 1 and $r - 1$, we implemented the algorithm in Cook and Su (2012). When the dimension is r , then the scaled envelope model degenerates to the standard multivariate linear regression. When the dimension is 0, it means that X and Y are uncorrelated, and the fitting is different.

References

1. The codes are implemented based on the algorithm in Section 4.1 of Cook and Su (2012).
2. The Grassmann manifold optimization step calls the package `sg_min` 2.4.3 by Ross Lipert (<http://web.mit.edu/~ripper/www/sgmin.html>).

Example

The following codes produce the results of the test and performance example in Cook and Su (2012).

```
load('sales.txt')
Y = sales(:, 4 : 7);
X = sales(:, 1 : 3);
u = bic_senv(X, Y)
```

`u =`

2

```

ModelOutput = senv(X, Y, u)
ModelOutput.Lambda
1 - 1 ./ ModelOutput.ratio

```

```
ModelOutput =
```

```

    beta: [4x3 double]
    Sigma: [4x4 double]
    Lambda: [4x4 double]
    Gamma: [4x2 double]
    Gamma0: [4x2 double]
    eta: [2x3 double]
    Omega: [2x2 double]
    Omega0: [2x2 double]
    alpha: [4x1 double]
    paramNum: 23
    l: -386.1900
    covMatrix: [12x12 double]
    asySenv: [4x3 double]
    ratio: [4x3 double]
    n: 50

```

```
ans =
```

```

1.0000    0    0    0
    0    0.9729    0    0
    0    0    0.8067    0
    0    0    0    1.7016

```

```
ans =
```

```

0.7020    0.5218    0.6517
0.6137    0.3219    0.5614
0.4439    0.1858    0.3924
0.5375    0.4639    0.5228

```

8.9 testcoefficient_senv

This function tests the null hypothesis $L * \beta * R = A$ versus the alternative hypothesis $L * \beta * R \neq A$, where β is estimated under the scaled envelope model.

Contents

- Syntax
- Input
- Output
- Description
- Example

Syntax

```
TestOutput = testcoefficient_senv(ModelOutput)
TestOutput = testcoefficient_senv(ModelOutput, TestInput)
```

Input

ModelOutput: A list containing the maximum likelihood estimators and other statistics inherited from `senv`.

TestInput: A list that specifies the null hypothesis, including L , R , and A . If not provided by the user, default values will be used.

- TestInput.L: The matrix multiplied to β on the left. It is a $d1$ by r matrix, while $d1$ is less than or equal to r . Default value: identity matrix I_r .
- TestInput.R: The matrix multiplied to β on the right. It is a p by $d2$ matrix, while $d2$ is less than or equal to p . Default value: identity matrix I_p .
- TestInput.A: The matrix on the right hand side of the equation. It is a $d1$ by $d2$ matrix. Default value: $d1$ by $d2$ zero matrix.

Output

TestOutput: A list containing test statistics, degrees of freedom for the reference chi-squared distribution, the p-value, and the covariance matrix of $\text{vec}(L\beta R)$. At the same time, a table is printed out.

- TestOutput.chisqStatistic: The test statistics. A real number.
- TestOutput.df: The degrees of freedom of the reference chi-squared distribution. A positive integer.
- TestOutput.pValue: p-value of the test. A real number in $[0, 1]$.
- TestOutput.covMatrix: The covariance matrix of $\text{vec}(L\beta R)$. A $d1 * d2$ by $d1 * d2$ matrix.

Description

This function tests for hypothesis $H_0 : L\beta R = A$, versus $H_a : L\beta R \neq A$. The β is estimated by the scaled envelope model. If the user does not specify the values for L , R and A , then the test is equivalent to the standard F test on if $\beta = 0$. The test statistics used is $\text{vec}(L\beta R - A)^T \hat{\Sigma}^{-1} \text{vec}(L\beta R - A)$, and the reference distribution is chi-squared distribution with degrees of freedom $d1 * d2$.

Example

```

load('sales.txt')
Y = sales(:,4:7);
X = sales(:,1:3);
u = bic_senv(X,Y)
ModelOutput = senv(X,Y,u);
TestOutout = testcoefficient_senv(ModelOutput);

```

Test Hypothesis	Chisq Statistic	DF	P-value
L * beta * R = A	4827.816	12	0.0000

```

r = size(Y, 2);
p = size(X, 2);
TestInput.L = rand(2, r);
TestInput.R = rand(p, 1);
TestInput.A = zeros(2, 1);
TestOutout = testcoefficient_senv(ModelOutput, TestInput);

```

Test Hypothesis	Chisq Statistic	DF	P-value
L * beta * R = A	1025.948	2	0.0000

9.1 aic_xenv

Use Akaike information criterion to select the dimension of the envelope subspace for the reduction on X .

Contents

- Syntax
- Input
- Output
- Description
- Example

Syntax

```
u = aic_xenv(X, Y)
u = aic_xenv(X, Y, Opts)
```

Input

X: Predictors. An n by p matrix, p is the number of predictors and n is number of observations. The number of predictors should be greater than the number of the responses. And they must be continuous variables.

Y: Responses. An n by r matrix, r is the number of responses. The response can be univariate or multivariate and must be continuous variable.

Opts: A list containing the optional input parameters, to control the iterations in `sg_min`. If one or several (even all) fields are not defined, the default settings are used.

- `Opts.maxIter`: Maximum number of iterations. Default value: 300.
- `Opts.ftol`: Tolerance parameter for F. Default value: $1e-10$.
- `Opts.gradtol`: Tolerance parameter for dF. Default value: $1e-7$.
- `Opts.verbose`: Flag for print out dimension selection process, logical 0 or 1. Default value: 0.

Output

u: Dimension of the envelope. An integer between 0 and p .

Description

This function implements the Akaike information criteria (AIC) to select the dimension of the envelope subspace for the reduction on X .

Example

```
load wheatprotein.txt
X = wheatprotein(:, 1 : 6);
Y = wheatprotein(:, 7);
u = aic_xenv(X, Y)
```

u =

4

9.2 bic_xenv

Use Bayesian information criterion to select the dimension of the envelope subspace for the reduction on X.

Contents

- Syntax
- Input
- Output
- Description
- Example

Syntax

```
u = bic_xenv(X, Y)
u = bic_xenv(X, Y, Opts)
```

Input

X: Predictors. An n by p matrix, p is the number of predictors and n is number of observations. The number of predictors should be greater than the number of the responses. And they must be continuous variables.

Y: Responses. An n by r matrix, r is the number of responses. The response can be univariate or multivariate and must be continuous variable.

Opts: A list containing the optional input parameters, to control the iterations in sg_min. If one or several (even all) fields are not defined, the default settings are used.

- Opts.maxIter: Maximum number of iterations. Default value: 300.
- Opts.ftol: Tolerance parameter for F. Default value: 1e-10.
- Opts.gradtol: Tolerance parameter for dF. Default value: 1e-7.
- Opts.verbose: Flag for print out dimension selection process, logical 0 or 1. Default value: 0.

Output

u: Dimension of the envelope. An integer between 0 and p.

Description

This function implements the Bayesian information criteria (BIC) to select the dimension of the envelope subspace for the reduction on X.

Example

```
load wheatprotein.txt
X = wheatprotein(:, 1 : 6);
Y = wheatprotein(:, 7);
u = bic_xenv(X, Y)
```

u =

4

9.3 bstrp_xenv

Compute bootstrap standard error of the envelope model for the reduction on X.

Contents

- Syntax
- Input
- Output
- Description
- Example

Syntax

```
bootse = bstrp_xenv(X, Y, u, B)
bootse = bstrp_xenv(X, Y, u, B, Opts)
```

Input

X: Predictors. An n by p matrix, p is the number of predictors and n is number of observations. The number of predictors should be greater than the number of the responses. And they must be continuous variables.

Y: Responses. An n by r matrix, r is the number of responses. The response can be univariate or multivariate and must be continuous variable.

u: Dimension of the envelope subspace. A positive integer between 0 and p .

B: Number of bootstrap samples. A positive integer.

Opts: A list containing the optional input parameters, to control the iterations in `sg_min`. If one or several (even all) fields are not defined, the default settings are used.

- `Opts.maxIter`: Maximum number of iterations. Default value: 300.
- `Opts.ftol`: Tolerance parameter for F. Default value: $1e-10$.
- `Opts.gradtol`: Tolerance parameter for dF. Default value: $1e-7$.
- `Opts.verbose`: Flag for print out the number of bootstrap samples, logical 0 or 1. Default value: 0.

Output

bootse: The standard error for elements in β computed by bootstrap. An p by r matrix.

Description

This function computes the bootstrap standard errors for the regression coefficients in the envelope model by bootstrapping the residuals. The envelope model here is for the reduction on X.

Example

```
load wheatprotein.txt
X = wheatprotein(:, 1 : 6);
Y = wheatprotein(:, 7);
alpha = 0.01;
u = lrt_xenv(X, Y, alpha)
```

u =

4

```
B = 100;
bootse = bstrp_xenv(X, Y, u, B)
```

bootse =

```
0.0222
0.0387
0.0413
0.0167
0.0022
0.0087
```

9.4 dF4xenv

The first derivative of the objective function for computing the envelope subspace for the reduction on X.

Contents

- Syntax
- Input
- Output
- Description

Syntax

```
df = dF4xenv(R, DataParameter)
```

Input

R: A p by u semi orthogonal matrix, $0 < u \leq p$.

DataParameter: A structure that contains the statistics calculated from the data.

Output

df: An p by u matrix containing the value of the derivative function evaluated at R.

Description

The objective function is derived in Section 4.5.1 of Cook et al. (2012) by using maximum likelihood estimation. This function is the derivative of the objective function.

9.5 F4xenv

Objective function for computing the envelope subspace for the reduction on X.

Contents

- Syntax
- Input
- Output
- Description

Syntax

```
f = F4xenv(R, DataParameter)
```

Input

R: A p by u semi orthogonal matrix, $0 < u \leq p$.

DataParameter: A structure that contains the statistics calculated from the data.

Output

f: A scalar containing the value of the objective function evaluated at R.

Description

The objective function is derived in Section 4.5.1 of Cook et al. (2012) using maximum likelihood estimation. The columns of the semi-orthogonal matrix that minimizes this function span the estimated envelope subspace.

9.6 lrt_xenv

Use likelihood ratio testing to select the dimension of the envelope subspace for the reduction on X .

Contents

- Syntax
- Input
- Output
- Description
- Example

Syntax

```
u = lrt_xenv(X, Y, alpha)
u = lrt_xenv(X, Y, alpha, Opts)
```

Input

X: Predictors. An n by p matrix, p is the number of predictors and n is number of observations. The number of predictors should be greater than the number of the responses. And they must be continuous variables.

Y: Responses. An n by r matrix, r is the number of responses. The response can be univariate or multivariate and must be continuous variable.

alpha: Significance level for testing. A real number between 0 and 1, often taken at 0.05 or 0.01.

Opts: A list containing the optional input parameters, to control the iterations in `sg_min`. If one or several (even all) fields are not defined, the default settings are used.

- `Opts.maxIter`: Maximum number of iterations. Default value: 300.
- `Opts.ftol`: Tolerance parameter for F. Default value: $1e-10$.
- `Opts.gradtol`: Tolerance parameter for dF. Default value: $1e-7$.
- `Opts.verbose`: Flag for print out dimension selection process, logical 0 or 1. Default value: 0.

Output

u: Dimension of the envelope. An integer between 0 and p .

Description

This function implements the likelihood ratio testing procedure to select the dimension of the envelope subspace for the reduction on X , with pre-specified significance level α .

Example

```
load wheatprotein.txt
X = wheatprotein(:, 1 : 6);
Y = wheatprotein(:, 7);
alpha = 0.01;
u = lrt_xenv(X, Y, alpha)
```

u =

4

9.7 predict_xenv

Perform estimation or prediction under the envelope model for the reduction on X.

Contents

- Syntax
- Input
- Output
- Description
- Example

Syntax

```
PredictOutput = predict_xenv(ModelOutput, Xnew, infType)
```

Input

ModelOutput: A list containing the maximum likelihood estimators and other statistics inherited from xenv.

Xnew: The value of X with which to estimate or predict Y. A p by 1 vector.

infType: A string of characters indicting the inference type, the choices can be 'estimation' or 'prediction'.

Output

PredictOutput: A list containing the results of the inference.

- PredictOutput.value: The fitted value or the prediction value evaluated at Xnew. An r by 1 vector.
- PredictOutput.covMatrix: The covariance matrix of PredictOutput.value. An r by r matrix.
- PredictOutput.SE: The standard error of elements in PredictOutput.value. An r by 1 vector.

Description

This function evaluates the envelope model for the reduction on X at new value Xnew. It can perform estimation: find the fitted value when $X = X_{\text{new}}$, or prediction: predict Y when $X = X_{\text{new}}$. The covariance matrix and the standard errors are also provided.

Example

```
load wheatprotein.txt
X = wheatprotein(:, 1 : 6);
Y = wheatprotein(:, 7);
u = bic_xenv(X, Y);
ModelOutput = xenv(X, Y, u);
Xnew = X(1, :)';
PredictOutput = predict_xenv(ModelOutput, Xnew, 'estimation')
```

```
[PredictOutput.value, Y(1, :)] % Compare the fitted value with the data  
PredictOutput.SE
```

```
PredictOutput =
```

```
value: 9.1751  
covMatrix: 16.8439  
SE: 4.1041
```

```
ans =
```

```
9.1751    9.2300
```

```
ans =
```

```
4.1041
```

```
PredictOutput = predict_xenv(ModelOutput, Xnew, 'prediction')  
PredictOutput.SE
```

```
PredictOutput =
```

```
value: 9.1751  
covMatrix: 16.8760  
SE: 4.1080
```

```
ans =
```

```
4.1080
```

9.8 testcoefficient_xenv

This function tests the null hypothesis $L * \beta * R = A$ versus the alternative hypothesis $L * \beta * R \neq A$, where β is estimated under the envelope model for the reduction on X.

Contents

- Syntax
- Input
- Output
- Description
- Example

Syntax

```
TestOutput = testcoefficient_xenv(ModelOutput)
TestOutput = testcoefficient_xenv(ModelOutput, TestInput)
```

Input

ModelOutput: A list containing the maximum likelihood estimators and other statistics inherited from xenv.

TestInput: A list that specifies the null hypothesis, including L, R, and A. If not provided by the user, default values will be used.

- TestInput.L: The matrix multiplied to β on the left. It is a d1 by p matrix, while d1 is less than or equal to p. Default value: identity matrix I_p .
- TestInput.R: The matrix multiplied to β on the right. It is a r by d2 matrix, while d2 is less than or equal to r. Default value: identity matrix I_r .
- TestInput.A: The matrix on the right hand side of the equation. It is a d1 by d2 matrix. Default value: d1 by d2 zero matrix.

Output

TestOutput: A list containing test statistics, degrees of freedom for the reference chi-squared distribution, the p-value, and the covariance matrix of $\text{vec}(L\beta R)$. At the same time, a table is printed out.

- TestOutput.chisqStatistic: The test statistics. A real number.
- TestOutput.df: The degrees of freedom of the reference chi-squared distribution. A positive integer.
- TestOutput.pValue: p-value of the test. A real number in [0, 1].
- TestOutput.covMatrix: The covariance matrix of $\text{vec}(L\beta R)$. A d1 * d2 by d1 * d2 matrix.

Description

This function tests for hypothesis $H_0 : L\beta R = A$, versus $H_a : L\beta R \neq A$. The β is estimated by the envelope model for the reduction on X. If the user does not specify the values for L, R and A, then the test is equivalent to the standard F test on if $\beta = 0$. The test statistics used is $\text{vec}(L\beta R - A) \hat{\Sigma}^{-1} \text{vec}(L\beta R - A)^T$, and the reference distribution is chi-squared distribution with degrees of freedom d1 * d2.

Example

```
load wheatprotein.txt
X=wheatprotein(:, 1 : 6);
Y=wheatprotein(:, 7);
u = bic_xenv(X, Y);
ModelOutput=xenv(X, Y, u);
TestOutout = testcoefficient_xenv(ModelOutput);
```

Test Hypothesis	Chisq Statistic	DF	P-value
L * beta * R = A	3233.053	6	0.0000

```
r = size(Y, 2);
p = size(X, 2);
TestInput.L = rand(2, p);
TestInput.R = rand(r, 1);
TestInput.A = zeros(2, 1);
TestOutout = testcoefficient_xenv(ModelOutput, TestInput);
```

Test Hypothesis	Chisq Statistic	DF	P-value
L * beta * R = A	33.578	2	0.0000

9.9 xenv

Fit the envelope model for the reduction on X.

Contents

- Syntax
- Input
- Output
- Description
- References
- Example

Syntax

```
ModelOutput = xenv(X, Y, u)
ModelOutput = xenv(X, Y, u, Opts)
```

Input

X: Predictors. An n by p matrix, p is the number of predictors and n is number of observations. The number of predictors should be greater than the number of the responses. And they must be continuous variables.

Y: Responses. An n by r matrix, r is the number of responses. The response can be univariate or multivariate and must be continuous variable.

u: Dimension of the envelope. An integer between 0 and p .

Opts: A list containing the optional input parameters, to control the iterations in `sg_min`. If one or several (even all) fields are not defined, the default settings are used.

- `Opts.maxIter`: Maximum number of iterations. Default value: 300.
- `Opts.ftol`: Tolerance parameter for F. Default value: $1e-10$.
- `Opts.gradtol`: Tolerance parameter for dF. Default value: $1e-7$.
- `Opts.verbose`: Flag for print out Grassmann manifold optimization process, logical 0 or 1. Default value: 0.
- `Opts.init`: The initial value for the envelope subspace. An p by u matrix. Default value is the one generated by function `get_Init`.

Output

ModelOutput: A list that contains the maximum likelihood estimators and some statistics.

- `ModelOutput.beta`: The envelope estimator of the regression coefficients β . An p by r matrix.
- `ModelOutput.SigX`: The envelope estimator of the covariance matrix of X, Σ_X . A p by p matrix.
- `ModelOutput.Gamma`: The orthogonal basis of the envelope subspace. An p by u semi-orthogonal matrix.
- `ModelOutput.Gamma0`: The orthogonal basis of the complement of the envelope subspace. An p by $p-u$ semi-orthogonal matrix.
- `ModelOutput.eta`: The coordinates of β with respect to Gamma. An u by r matrix.

- `ModelOutput.Omega`: The coordinates of Σ_X with respect to Γ . An u by u matrix.
- `ModelOutput.Omega0`: The coordinates of Σ_X with respect to Γ_0 . An $p - u$ by $p - u$ matrix.
- `ModelOutput.mu`: The estimated intercept. An r by 1 vector.
- `ModelOutput.sigYcX`: The estimated conditional covariance matrix of Y given X . An r by r matrix.
- `ModelOutput.l`: The maximized log likelihood function. A real number.
- `ModelOutput.covMatrix`: The asymptotic covariance of $\text{vec}(\beta)$. An pr by pr matrix. The covariance matrix returned are asymptotic. For the actual standard errors, multiply by $1 / n$.
- `ModelOutput.asyXenv`: Asymptotic standard error for elements in β under the envelope model. An r by p matrix. The standard errors returned are asymptotic, for actual standard errors, multiply by $1 / \sqrt{n}$.
- `ModelOutput.ratio`: The asymptotic standard error ratio of the standard multivariate linear regression estimator over the envelope estimator, for each element in β . An p by r matrix.
- `ModelOutput.paramNum`: The number of parameters in the envelope model. A positive integer.
- `ModelOutput.n`: The number of observations in the data. A positive integer.

Description

This function fits the envelope model in the predictor's space, using the maximum likelihood estimation. When the dimension of the envelope is between 1 and $r - 1$, we implemented the algorithm in Cook et al. (2012). When the dimension is r , then the envelope model degenerates to the standard multivariate linear regression. When the dimension is 0, it means that X and Y are uncorrelated, and the fitting is different.

References

1. The codes are implemented based on the algorithm in Section 4.5.1 of Cook et al (2012).
2. The Grassmann manifold optimization step calls the package `sg_min` 2.4.3 by Ross Lippert (<http://web.mit.edu/~ripper/www/sgmin.html>).

Example

```
load wheatprotein.txt
X = wheatprotein(:, 1 : 6);
Y = wheatprotein(:, 7);

p = size(X, 2);
ModelOutput = xenv(X, Y, p);

% When u = p, the envelope model reduces to the ordinary least squares
% regression

temp = fit_OLS(X, Y);
temp.SigmaOLS
ModelOutput.sigYcX
```



```
ans =
```

```
0.0321
```

```
ans =
```

```
0.0321
```

```
temp.betaOLS'
ModelOutput.beta
```

```
ans =
```

```
-0.0416
-0.0490
0.3368
-0.1981
0.0020
-0.0480
```

```
ans =
```

```
-0.0416
-0.0490
0.3368
-0.1981
0.0020
-0.0480
```

```
u = bic_xenv(X, Y);
ModelOutput = xenv(X, Y, u)
```

```
ModelOutput =
```

```
beta: [6x1 double]
SigX: [6x6 double]
Gamma: [6x4 double]
Gamma0: [6x2 double]
eta: [4x1 double]
Omega: [4x4 double]
Omega0: [2x2 double]
mu: 24.8863
sigYcX: 0.0321
l: -865.6407
```

```
covMatrix: [6x6 double]
  asyXenv: [6x1 double]
    ratio: [6x1 double]
paramNum: 27
      n: 50
```

```
% To compare with the results obtained by Partial Least Squares, use the
% plsregress command
[XL, YL, XS, YS, BETA, PCTVAR, MSE, stats] = plsregress(X, Y, u);
ModelOutput.beta
BETA(2 : end, :)
```

```
ans =
```

```
-0.0443
-0.0481
 0.3377
-0.1963
 0.0019
-0.0487
```

```
ans =
```

```
-0.0199
 0.1373
 0.1309
-0.1827
 0.0056
-0.0708
```

xenvpls

10.1 bstrp_xenvpls

Compute bootstrap standard error of the envelope model for the reduction on X using partial least squares algorithm.

Contents

- Syntax
- Input
- Output
- Description
- Example

Syntax

```
bootse = bstrp_xenvpls(X, Y, u, B)
bootse = bstrp_xenvpls(X, Y, u, B, Opts)
```

Input

X: Predictors. An n by p matrix, p is the number of predictors and n is number of observations. The number of predictors should be greater than the number of the responses. And they must be continuous variables.

Y: Responses. An n by r matrix, r is the number of responses. The response can be univariate or multivariate and must be continuous variable.

u: Dimension of the envelope subspace. A positive integer between 0 and p .

B: Number of bootstrap samples. A positive integer.

Opts: A list containing the optional input parameters. If one or several (even all) fields are not defined, the default settings are used.

- **Opts.verbose:** Flag for print out the number of bootstrap samples, logical 0 or 1. Default value: 0.

Output

bootse: The standard error for elements in β computed by bootstrap. A p by r matrix.

Description

This function computes the bootstrap standard errors for the regression coefficients in the envelope model by bootstrapping the residuals. The envelope model is applied for the reduction on X , using the partial least squares algorithm.

Example

```
load VocabGrowth
X = VocabGrowth(:, 1 : 3);
Y = VocabGrowth(:, 4);
m = 5;
u = mfoldcv_xenvpls(X, Y, m)

u =

    1

B = 100;
bootse = bstrp_xenvpls(X, Y, u, B)

bootse =

    0.0230
    0.0235
    0.0273
```

10.2 mfoldcv_xenvpls

Select the dimension of the envelope subspace using m-fold cross validation for envelope model on the reduction on X using partial least squares algorithm.

Contents

- Syntax
- Input
- Output
- Description
- Example

Syntax

```
u = mfoldcv_xenvpls(X, Y, m)
u = mfoldcv_xenvpls(X, Y, m, Opts)
```

Input

X: Predictors. An n by p matrix, p is the number of predictors and n is number of observations. The number of predictors should be greater than the number of the responses. And they must be continuous variables.

Y: Responses. An n by r matrix, r is the number of responses. The response can be univariate or multivariate and must be continuous variable.

m: A positive integer that is used to indicate m-fold cross validation.

Opts: A list containing the optional input parameters. If one or several (even all) fields are not defined, the default settings are used.

- Opts.verbose: Flag for print out the number of bootstrap samples, logical 0 or 1. Default value: 0.

Output

u: The dimension of the envelope subspace selected by m-fold cross validation.

Description

This function implements m-fold cross validation to select the dimension of the envelope space, based on prediction performance. For each u, the data is partitioned into m parts, each part is in turn used for testing for the prediction performance while the rest m-1 parts are used for training. The dimension is select as the one that minimizes the average prediction errors. If Y is multivariate, the identity inner product is used for computing the prediction errors.

Example

```
load VocabGrowth
X = VocabGrowth(:, 1 : 3);
Y = VocabGrowth(:, 4);
```

```
m = 5;  
u = mfoldcv_xenvpls(X, Y, m)
```

```
u =
```

```
1
```

10.3 xenvpls

Fit the envelope model for the reduction on X using partial least squares algorithm.

Contents

- Syntax
- Input
- Output
- Description
- Reference
- Example

Syntax

```
ModelOutput = xenvpls(X, Y, u)
```

Input

X: Predictors. An n by p matrix, p is the number of predictors and n is number of observations. The number of predictors should be greater than the number of the responses. And they must be continuous variables.

Y: Responses. An n by r matrix, r is the number of responses. The response can be univariate or multivariate and must be continuous variable.

u: Dimension of the envelope. An integer between 0 and p.

Output

ModelOutput: A list that contains the maximum likelihood estimators and some statistics.

- ModelOutput.beta: The envelope estimator of the regression coefficients β . An p by r matrix.
- ModelOutput.SigX: The envelope estimator of the covariance matrix of X, Σ_X . A p by p matrix.
- ModelOutput.Gamma: The orthogonal basis of the envelope subspace. An p by u semi-orthogonal matrix.
- ModelOutput.Gamma0: The orthogonal basis of the complement of the envelope subspace. An p by p-u semi-orthogonal matrix.
- ModelOutput.eta: The coordinates of β with respect to Gamma. An u by r matrix.
- ModelOutput.Omega: The coordinates of Σ_X with respect to Gamma. An u by u matrix.
- ModelOutput.Omega0: The coordinates of Σ_X with respect to Gamma0. An p - u by p - u matrix.
- ModelOutput.mu: The estimated intercept. An r by 1 vector.
- ModelOutput.sigYcX: The estimated conditional covariance matrix of Y given X. An r by r matrix.
- ModelOutput.paramNum: The number of parameters in the envelope model. A positive integer.
- ModelOutput.n: The number of observations in the data. A positive integer.

Description

This function fits the envelope model in the predictor's space, by the partial least squares algorithm in Cook et al. (2012). In the population level, this algorithm is equivalent to that in `xenv.m`, which uses the maximum likelihood estimation. In the sample version, the two algorithms are different. And this algorithm is much faster, which provides a root n consistent starting value for the one in `xenv.m`.

Reference

The codes are implemented based on the algorithm in Section 4.3 of Cook et al (2012).

Example

```
load VocabGrowth
X = VocabGrowth(:, 1 : 3);
Y = VocabGrowth(:, 4);
m = 5;
u = mfoldcv_xenvpls(X, Y, m)

u =

    1

ModelOutput = xenvpls(X, Y, u)

ModelOutput =

    beta: [3x1 double]
    SigX: [3x3 double]
    Gamma: [3x1 double]
    Gamma0: [3x2 double]
    eta: 5.2899
    Omega: 10.9286
    Omega0: [2x2 double]
    mu: 1.5683
    sigYcX: 1.0934
    paramNum: 9
    n: 64

ModelOutput.beta

ans =

    0.2573
    0.2741
    0.3049
```


11.1 center

Subtract the mean of each column.

Contents

- Syntax
- Input
- Output
- Description

Syntax

```
XC = center(X)
```

Input

X: A matrix or a column vector.

Output

XC: A matrix or a column vector with the mean for each column equal to 0.

Description

This function centerizes a matrix or a vector, by subtracting each column by its column mean.

11.2 Contr

Compute the contraction matrix of dimension r .

Contents

- Syntax
- Input
- Output
- Description

Syntax

$$C = \text{Contr}(r)$$

Input

r: Dimension of the contraction matrix. A positive integer.

Output

C: Contraction matrix of dimension r . C is an $r(r + 1) / 2$ by r^2 matrix.

Description

The contraction and expansion matrices are links between the "vec" operator and "vech" operator: for an r by r symmetric matrix A , $\text{vech}(A) = \text{Contr}(r) * \text{vec}(A)$, and $\text{vec}(A) = \text{Expan}(r) * \text{vech}(A)$. The "vec" operator stacks the matrix A into an r^2 by 1 vector columnwise. The "vech" operator stacks the lower triangle or the upper triangle of a symmetric matrix into an $r(r+1)/2$ vector. For more details of "vec", "vech", contraction and expansion matrix, refer to Henderson and Searle (1979).

11.3 Expan

Compute the expansion matrix of dimension r .

Contents

- Syntax
- Input
- Output
- Description

Syntax

$$E = \text{Expan}(r)$$

Input

r: Dimension of the expansion matrix. A positive integer.

Output

E: Expansion matrix of dimension r . E is an r^2 by $r(r+1)/2$ matrix.

Description

The contraction and expansion matrices are links between the "vec" operator and "vech" operator: for an r by r symmetric matrix A , $\text{vech}(A) = \text{Contr}(r) * \text{vec}(A)$, and $\text{vec}(A) = \text{Expan}(r) * \text{vech}(A)$. The "vec" operator stacks the matrix A into an r^2 by 1 vector columnwise. The "vech" operator stacks the lower triangle or the upper triangle of a symmetric matrix into an $r(r+1)/2$ vector. For more details of "vec", "vech", contraction and expansion matrix, refer to Henderson and Searle (1979).

11.4 fit_OLS

Multivariate linear regression.

Contents

- Syntax
- Input
- Output
- Description
- Example

Syntax

```
ModelOutput = fit_OLS(X, Y)
```

Input

X: Predictors, an n by p matrix, p is the number of predictors. The predictors can be univariate or multivariate, discrete or continuous.

Y: Multivariate responses, an n by r matrix, r is the number of responses and n is number of observations. The responses must be continuous variables.

Output

ModelOutput: A list that contains the maximum likelihood estimators of regression coefficients and error covariance matrix.

- ModelOutput.betaOLS: An r by p matrix containing estimate of the regression coefficients β .
- ModelOutput.SigmaOLS: An r by r matrix containing estimate of the error covariance matrix.
- ModelOutput.alpha: An r by 1 vector containing estimate of the intercept.
- ModelOutput.n: The number of observations in the data. A positive integer.

Description

In a multivariate linear model, Y and X follows the following relationship: $Y = \alpha + \beta X + \varepsilon$, where ε contains the errors. This function performs the ordinary least squares fit to the inputs, and returns the estimates of β and the covariance matrix of ε .

Example

```
load wheatprotein.txt
X = wheatprotein(:, 8);
Y = wheatprotein(:, 1 : 6);
ModelOutput = fit_OLS(X, Y)
ModelOutput.betaOLS
ModelOutput.SigmaOLS
```

```
ModelOutput =
```

```
    betaOLS: [6x1 double]
    SigmaOLS: [6x6 double]
    alpha: [6x1 double]
    n: 50
```

```
ans =
```

```
    3.2724
    8.0288
    7.5224
   -2.0609
    3.2244
    0.6538
```

```
ans =
```

```
1.0e+03 *
```

1.1905	0.9759	1.0506	1.1524	1.5384	0.6335
0.9759	0.8061	0.8657	0.9432	1.2636	0.5266
1.0506	0.8657	0.9310	1.0164	1.3664	0.5640
1.1524	0.9432	1.0164	1.1228	1.5234	0.6183
1.5384	1.2636	1.3664	1.5234	2.3229	0.8360
0.6335	0.5266	0.5640	0.6183	0.8360	0.3618

11.5 get_envelope

Construct the envelope subspace using a sequential algorithm.

Contents

- Syntax
- Input
- Output
- Description
- Reference

Syntax

```
W = get_envelope(S, M, u)
```

Input

S: An r by p matrix whose columns span the subspace, the rank of S cannot be greater than u .

M: An r by r positive semi-definite matrix.

u: Dimension of the envelope. An integer between 0 and r .

Output

W: An r by u semi-orthogonal matrix that spans the M -envelope of $\text{span}(S)$.

Description

This function constructs the M -envelope of $\text{span}(S)$ using a sequential algorithm similar to partial least squares.

Reference

The codes are implemented based on the algorithm in the lecture notes of Cook (2012).

Example

```
S = [1 2 3]';
S0 = grams(nulbasis(S'));
M = S * S' + S0 * S0';
u = 1;
W = get_envelope(S, M, u)
```

W =

```
0.2673
0.5345
0.8018
```

11.6 get_Init

Starting value for the envelope subspace.

Contents

- Syntax
- Input
- Output
- Description
- Reference

Syntax

```
WInit = get_Init(F, u, DataParameter)
```

Input

F: Objective function of the envelope subspace.

u: Dimension of the envelope. An integer between 1 and $r - 1$.

DataParameter: A list containing commonly used statistics computed from the data.

Output

WInit: The initial estimate of the orthogonal basis of the envelope subspace. An r by u orthogonal matrix.

Description

We compute the eigenvectors for the covariance matrices of Y and the estimated errors, and get $2r$ vectors. Then we get all the combinations of u vectors out of the $2r$ vectors. If the number of $2r$ choose u is small (≤ 50), we search over all the combinations and find out the one that minimizes the objective function F . If that number is large, then we do it iteratively: we pick up any u eigenvectors, fix all of them except the first one. Then we search over all the vectors orthogonal to the fixed ones, and record the one that minimizes F . Next, we fix the first u eigenvectors again but this time search for the second one, then we record the vector. This goes on and on until the last one. We do it for 5 rounds and use the final set as our starting value.

Reference

The codes are implemented based on the algorithm in Section 3.5 of Su and Cook (2011).

11.7 `get_Init4envmean`

Starting value for the envelope subspace in estimating the multiavriate mean.

Contents

- Syntax
- Input
- Output
- Description
- Reference

Syntax

```
WInit = get_Init4envmean(F, u, DataParameter)
```

Input

F: Objective function to get the envelope subspace.

u: Dimension of the envelope. An integer between 1 and $p - 1$.

DataParameter: A list containing commonly used statistics computed from the data.

Output

WInit: The initial estimate of the orthogonal basis of the envelope subspace. A p by u orthogonal matrix.

Description

We compute the eigenvectors for the estimated error covariance matrix, and get p vectors. Then we get all the combinations of u vectors out of the p vectors. If the number of p choose u is small (≤ 50), we search over all the combinations and find out the one that minimizes the objective function F . If that number is large, then we do it iteratively: we pick up any u eigenvectors, fix all of them except the first one. Then we search over all the vectors orthogonal to the fixed ones, and record the one that minimizes F . Next, we fix the first u eigenvectors again but this time search for the second one, then we record the vector. This goes on and on until the last one. We do it for 3 rounds and use the final set as our starting value.

Reference

The codes are implemented based on the algorithm in Section 3.5 of Su and Cook (2011).

11.8 get_Init4henv

Starting value for the heteroscedastic envelope subspace.

Contents

- Syntax
- Input
- Output
- Description
- Reference

Syntax

```
WInit = get_Init4henv(F, u, DataParameter)
```

Input

F: Objective function to get the heteroscedastic envelope subspace.

u: Dimension of the envelope. An integer between 1 and $r - 1$.

DataParameter: A list containing commonly used statistics computed from the data.

Output

WInit: The initial estimate of the orthogonal basis of the heteroscedastic envelope subspace. An r by u orthogonal matrix.

Description

We compute the eigenvectors for the estimated error covariance matrix, and get r vectors. Then we get all the combinations of u vectors out of the r vectors. If the number of r choose u is small (≤ 50), we search over all the combinations and find out the one that minimizes the objective function F . If that number is large, then we do it iteratively: we pick up any u eigenvectors, fix all of them except the first one. Then we search over all the vectors orthogonal to the fixed ones, and record the one that minimizes F . Next, we fix the first u eigenvectors again but this time search for the second one, then we record the vector. This goes on and on until the last one. We do it for 3 rounds and use the final set as our starting value.

Reference

The codes are implemented based on the algorithm in Section 3.5 of Su and Cook (2011).

11.9 Kpd

Compute the communication matrix Kpd.

Contents

- Syntax
- Input
- Output
- Description
- Reference

Syntax

$$k = \text{Kpd}(p, d)$$

Input

p, d: two positive integers represent the dimension parameters for the communication matrix.

Output

k: The communication matrix Kpd. An $p * d$ by $p * d$ matrix.

Description

For a p by d matrix A , $\text{vec}(A') = \text{Kpd} * \text{vec}(A)$, and Kpd is called a communication matrix.

Reference

The codes are implemented based on Definition 3.1 in Magnus and Neudecker (1979).

11.10 Lmatrix

Extract the 2nd to the last diagonal element of a matrix into a vector.

Contents

- Syntax
- Input
- Output
- Description

Syntax

```
L = Lmatrix(r)
```

Input

r: The dimension of the matrix being extracted. The matrix should be an r by r matrix.

Output

L: An r - 1 dimensional vector that contains all the diagonal elements but the first one of the matrix.

Description

Let A be an r by r matrix, and vec be the vector operator, then Lmatrix(r) * vec(A) will give the 2nd to the rth diagonal elements of A, arranged in a column vector.

11.11 make_dF

Generic function to generate the derivative function of the objective function F

Contents

- Syntax
- Input
- Output
- Description

Syntax

```
dF = make_dF(dfun_method_handle, FParameters)
```

Input

- `dfun_method_handle`: A specific model derivative function of the objective function.
- `FParameters`: A structure that contains data parameters as input for the function `dfun_method_handle`.

Output

- `dF`: The generic derivative function of the objective function for computing the envelope subspace.

Description

Generic function to generate the derivative function of the objective function F. The function first sets a handle to the specific model function and fixes the data parameters from the sample needed for its computation. The handle fixed with those parameters is then evaluated at a given value for argument W. A generic derivative function dF is returned.

11.12 make_F

Generic function to generate the objective function F

Contents

- Syntax
- Input
- Output
- Description

Syntax

`F = make_F(fun_method_handle, FParameters)`

Input

- `fun_method_handle`: A specific model objective function.
- `FParameters`: A structure that contains data parameters as input for the function `fun_method_handle`.

Output

- `F`: The generic objective function for computing the envelope subspace.

Description

Generic function to generate the objective function F. The function first sets a handle to the specific model function and fixes the data parameters from the sample needed for its computation. The handle fixed with those parameters is then evaluated at a given value for argument W. A generic objective function F is returned.

11.13 make_opts

Make optional input parameters for running the `sg_min` package.

Contents

- Syntax
- Input
- Output:
- Description

Syntax

```
Opts = make_opts(Opts)
```

Input

Opts: A list containing optional input parameters for `sg_min.m` specified by users. One or several (even all) fields could be empty.

- `Opts.maxIter`: Maximum number of iterations.
- `Opts.ftol`: Tolerance parameter for F.
- `Opts.gradtol`: Tolerance parameter for dF.
- `Opts.verbose`: Flag for print out output, logical 0 or 1.

Output:

Opts: A list containing optional input parameters for `sg_min.m`, specified by users or the default values are used.

- `Opts.maxIter`: Maximum number of iterations. Default value: 300.
- `Opts.ftol`: Tolerance parameter for F. Default value: $1e-10$.
- `Opts.gradtol`: Tolerance parameter for dF. Default value: $1e-7$.
- `Opts.verbose`: Flag for print out output, logical 0 or 1. Default value: 0.

Description

The `sg_min` function has some optional input parameters that control the iteration process. These parameters include maximum number of iteration, tolerance parameters for convergence of the objective function F and the derivative of the objective function dF, and the print out of the iteration process. The user can set one or all of parameters, if not, default values will be used.

11.14 make_parameter

Compute summary statistics from the data.

Contents

- Syntax
- Input
- Output
- Description

Syntax

```
DataParameter = make_parameter(X, Y, method)
```

Input

X: Predictors. An n by p matrix, p is the number of predictors. The predictors can be univariate or multivariate, discrete or continuous.

Y: Multivariate responses. An n by r matrix, r is the number of responses and n is number of observations. The responses must be continuous variables, and r should be strictly greater than p .

method: A string of characters indicating which member of the envelope family to be used, the choices can be 'env', 'ienv', 'henv', 'senv' or 'xenv'.

Output

DataParameter: A list that contains summary statistics computed from the data. The output list can vary from method to method.

- **DataParameter.n:** The number of observations in the data. A positive integer.
- **DataParameter.ng:** A p by 1 vector containing the number of observations in each group. p is the number of groups. Only for 'henv'.
- **DataParameter.ncum:** A p by 1 vector containing the total number of observations till this group. Only for 'henv'.
- **DataParameter.ind:** An n by 1 vector indicating the sequence of the observations after sorted by groups.
- **DataParameter.p:** The number of predictors or number of groups for 'henv'. A positive integer.
- **DataParameter.r:** The number of responses. A positive integer.
- **DataParameter.XC:** Centered predictors. An n by p matrix with the i th row being the i th observation of X subtracted by the mean of X . Only for 'env' and 'ienv'.
- **DataParameter.YC:** Centered responses. An n by r matrix with the i th row being the i th observation of Y subtracted by the mean of Y . Only for 'env' and 'ienv'.
- **DataParameter.mX:** The mean of predictors. A p by 1 vector. For all method except 'henv'.
- **DataParameter.mY:** The mean of responses. An r by 1 vector.
- **DataParameter.mYg:** An r by p matrix with the i th column being the sample mean of the i th group.
- **DataParameter.sigX:** The sample covariance matrix of X . A p by p matrix.
- **DataParameter.sigY:** The sample covariance matrix of Y . An r by r matrix.

- `DataParameter.sigRes`: For 'env', 'senv', 'ienv': The sample covariance matrix of the residuals from the ordinary least squares regression of Y on X. An r by r matrix. For 'henv', an r by r by p three dimensional matrix with the ith depth is the ith sample covariance matrix for the ith group.
- `DataParameter.sigFit`: The sample covariance matrix of the fitted value from the ordinary least squares regression of Y on X. An r by r matrix. Only for method 'ienv'.
- `DataParameter.betaOLS`: The regression coefficients from the ordinary least squares regression of Y on X. An r by p matrix. For all method except 'henv'.

Description

This function computes statistics that will be used frequently in the estimation for each method.

11.15 mtest

Perform Box's M test to check the homogeneity of the covariance matrices.

Contents

- Syntax
- Input
- Output
- Description
- References
- Example

Syntax

```
TestOutput = mtest(X, Y, alpha)
```

Input

X: Group indicators. A matrix with n rows. X can only have p unique rows, where p is the number of groups. For example, if there are two groups, X can only have 2 different kinds of rows, such as (0, 1) and (1, 0), or (1, 0, 10) and (0, 5, 6). The number of columns is not restricted, as long as X only has p unique rows.

Y: Multivariate responses. An n by r matrix, r is the number of responses and n is number of observations. The responses must be continuous variables, and r should be greater than p.

alpha: Significance level for testing. A real number between 0 and 1, often taken at 0.05 or 0.01.

Output

TestOutput: A list containing the Box's M statistic, the approximation test statistic, degrees of freedom for the approximation statistic test, and the p-value. At the same time, a table is printed out.

- TestOutput.mStatistic: The Box's M statistic. A real number.
- TestOutput.approxStatistic: The approximation test statistic.
- TestOutput.df: The degrees of freedom of the approximation statistic test. A positive integer.
- TestOutput.pValue: p-value of the test. A real number in [0, 1].

Description

This function performs the Box's M test for homogeneity of the covariance matrices for different groups, indicated by X. If the groups sample-size is at least 20 (sufficiently large), Box's M test takes a Chi-square approximation; otherwise it takes an F approximation.

References

The codes are implemented based on

Trujillo-Ortiz, A., R. Hernandez-Walls, K. Castro-Morales, A. Espinoza-Tenorio, A. Guia-Ramirez and R. Carmona-Pina. (2002). MBoxtest: Multivariate Statistical Testing for the Homogeneity of Covariance Matrices by the Box's M. A MATLAB file. [WWW document]. URL:

<http://www.mathworks.com/matlabcentral/fileexchange/loadFile.do?objectId=2733&objectType=FILE>

Example

```
load waterstrider.mat
alpha = 0.01;
TestOutput = mtest(X, Y, alpha);
```

MBox	Chi-sqr.	df	P
157.5977	137.3361	72	0.0000

Covariance matrices are significantly different.