# Package 'envlpaster'

April 19, 2016

**Title** Enveloping the Aster Model

**Version** 0.1-2

**Date** 2016-02-16

**Author** Daniel J. Eck <eckxx049@umn.edu>

**Maintainer** Daniel J. Eck <eckxx049@umn.edu>

**Description** Envelope methodology and aster modeling are combined to provide users with precise estimation of expected Darwinian fitness.

**Depends** R (>= 2.10.0), aster

**Imports** caTools, MASS, aster2

**License** GPL-2

**NeedsCompilation** no

**Repository** CRAN

**Date/Publication** 2016-04-19 08:59:41

## R topics documented:

## eigenboot *eigenboot*

### Description

A parametric bootstrap procedure evaluated at an envelope estimator of the submodel mean-value parameter vector $\tau$ that was obtained using reducing subspaces.

### Usage

```
eigenboot(model, nboot, index, vectors, data,
  quiet = FALSE, m = 100)
```

### Arguments

| | |
|---|---|
| `model` | An aster model object |
| `nboot` | The number of bootstrap iterations desired |
| `index` | The indices denoting which components of the canonical parameter vector are parameters of interest |
| `vectors` | The indices denoting which reducing subspace of inverse Fisher information is desired to construct envelope estimators |
| `data` | An asterdata object |
| `quiet` | A logical argument. If FALSE, the function displays how much time it takes to run `m` iterations |
| `m` | The length of the output interval |

### Details

This function implements the parametric bootstrap procedure given by the algorithm below with respect to the mean-value parameterization. This parametric bootstrap generates resamples from the distribution evaluated at an envelope estimator of $\tau$. The user specifies which vectors are used in order to construct envelope estimators using the reducing subspace approach. When one is using a partial envelope then this function constructs envelope estimators of $\upsilon$ where we write $\tau = (\gamma^T, \upsilon^T)^T$ and $\upsilon$ corresponds to aster model parameters of interest. In applications, candidate reducing subspaces are indices of eigenvectors of $\widehat{\Sigma}_{\upsilon,\upsilon}$ where $\widehat{\Sigma}_{\upsilon,\upsilon}$ is the part of $\widehat{\Sigma}$ corresponding to our parameters of interest. When all of the components of $\tau$ are components of interest, then we write $\widehat{\Sigma}_{\upsilon,\upsilon} = \widehat{\Sigma}$. The algorithm is as follows:

1. Fit the aster model to the data and obtain $\hat{\tau} = (\hat{\gamma}^T, \hat{\upsilon}^T)$ and $\widehat{\Sigma}$ from the aster model fit.
2. Compute the envelope estimator of $\upsilon$ in the original sample, given as $\hat{\upsilon}_{env} = P_{\hat{G}}\hat{\upsilon}$ where $P_{\hat{G}}$ is the projection into the reducing subspace of $\widehat{\Sigma}_{\upsilon,\upsilon}$ specified by `vectors`.

3. Perform a parametric bootstrap by generating resamples from the distribution evaluated at $\hat{\upsilon}_{env}$. For iteration $b = 1, ..., B$ of the procedure:

   (3a) Compute $\hat{\upsilon}^{(b)}$ and $\widehat{\Sigma}^{(b)}_{\upsilon,\upsilon}$ from the aster model fit to the resampled data.

   (3b) Obtain $P^{(b)}_{\hat{G}}$ as done in Step 2.

   (3c) Store $\hat{\upsilon}^{(b)}_{env} = P^{(b)}_{\hat{G}} \hat{\upsilon}^{(b)}$.

A parametric bootstrap generating resamples from the distribution evaluated at the aster model MLE is also conducted by this function.

## Value

`u` The dimension of the envelope space assumed

`table` A table of output. The first two columns display the envelope estimator and its bootstrapped standard error. The next two columns display the MLE and its bootstrapped standard error. The last column displays the ratio of the standard error for the bootstrapped envelope estimator to the standard error for the bootstrapped MLE.

`S` The bootstrap estimator of the variability of the partial envelope estimator.

`S2` The bootstrap estimator of the variability of the MLE.

`env.boot.out` The realizations from the bootstrap procedure using envelope methodology.

`MLE.boot.out` The realizations from the bootstrap procedure using maximum likelihood estimation.

## References

Eck, D. J., Geyer, C. J., and Cook, R. D. (2016). Enveloping the aster model. *in prep*.

## Examples

```
## Not run: set.seed(13)
library(envlpaster)
library(aster2)
data(simdata30nodes)
data <- simdata30nodes.asterdata
nnode <- length(vars)
xnew <- as.matrix(simdata30nodes[,c(1:nnode)])
m1 <- aster(xnew, root, pred, fam, modmat)
target <- 5:9
indices <- c(1,2,4,5)
u <- length(indices)
nboot <- 2000; timer <- nboot/2
bar <- eigenboot(m1, nboot = nboot, index = target,
  u = u, vectors = indices, data = data, m = timer)
bar
## End(Not run)
```

## `eigenbootcanon` *eigenbootcanon*

### Description

A parametric bootstrap procedure evaluated at an envelope estimator of the submodel mean-value parameter vector $\beta$ that was obtained using reducing subspaces.

### Usage

```
eigenbootcanon(model, nboot, index, vectors,
  code, families, quiet = FALSE, m = 100)
```

### Arguments

`model` An aster model object.

`nboot` The number of bootstrap iterations desired.

`index` The indices denoting which components of the canonical parameter vector are parameters of interest.

`vectors` The indices denoting which reducing subspace of inverse Fisher information is desired to construct envelope estimators.

`code` A vector of dimension equal to the number of nodes in the aster graph. This vector specifies which exponential family in the `families` list labels each arrow of the aster graphical structure.

`families` A list of family specifications.

`quiet` A logical argument. If FALSE, the function displays how much time it takes to run `m` iterations.

`m` The length of the output interval.

### Details

This function implements the parametric bootstrap procedure given by the algorithm below with respect to the canonical parameterization. This parametric bootstrap generates resamples from the distribution evaluated at an envelope estimator of $\beta$. The user specifies which vectors are used in order to construct envelope estimators using the reducing subspace approach. When one is using a partial envelope then this function constructs envelope estimators of $\upsilon$ where we write $\beta$ = $(\gamma^T, \upsilon^T)^T$ and $\upsilon$ corresponds to aster model parameters of interest. In applications, candidate reducing subspaces are indices of eigenvectors of $\widehat{\Sigma}^{-1}_{\upsilon,\upsilon}$ where $\widehat{\Sigma}^{-1}_{\upsilon,\upsilon}$ is the part of $\widehat{\Sigma}^{-1}$ corresponding to our parameters of interest. When all of the components of $\beta$ are components of interest, then we write $\widehat{\Sigma}^{-1}_{\upsilon,\upsilon} = \widehat{\Sigma}^{-1}$. The algorithm is as follows:

1. Fit the aster model to the data and obtain $\hat{\beta} = (\hat{\gamma}^T, \hat{\upsilon}^T)$ and $\widehat{\Sigma}^{-1}$ from the aster model fit.
2. Compute the envelope estimator of $\upsilon$ in the original sample, given as $\hat{\upsilon}_{env} = P_{\hat{G}}\hat{\upsilon}$ where $P_{\hat{G}}$ is the projection into the reducing subspace of $\widehat{\Sigma}^{-1}_{\upsilon,\upsilon}$ specified by `vectors`.

3. Perform a parametric bootstrap by generating resamples from the distribution evaluated at $\hat{\upsilon}_{env}$. For iteration $b = 1, ..., B$ of the procedure:

   (3a) Compute $\hat{\upsilon}^{(b)}$ and $\widehat{\Sigma}_{\upsilon,\upsilon}^{(b)^{-1}}$ from the aster model fit to the resampled data.

   (3b) Obtain $P_{\hat{G}}^{(b)}$ as done in Step 2.

   (3c) Store $\hat{\upsilon}_{env}^{(b)} = P_{\hat{G}}^{(b)} \hat{\upsilon}^{(b)}$.

A parametric bootstrap generating resamples from the distribution evaluated at the aster model MLE is also conducted by this function.

## Value

| | |
|---|---|
| u | The dimension of the envelope space assumed. |
| table | A table of output. The first two columns display the envelope estimator and its bootstrapped standard error. The next two columns display the MLE and its bootstrapped standard error. The last column displays the ratio of the standard error for the bootstrapped envelope estimator to the standard error for the bootstrapped MLE. |
| S | The bootstrap estimator of the variability of the partial envelope estimator. |
| S2 | The bootstrap estimator of the variability of the MLE. |
| env.boot.out | The realizations from the bootstrap procedure using envelope methodology. |
| MLE.boot.out | The realizations from the bootstrap procedure using maximum likelihood estimation. |

## References

Eck, D. J., Geyer, C. J., and Cook, R. D. (2016). Enveloping the aster model. *in prep*.

## Examples

```
## Not run: set.seed(13)
library(envlpaster)
library(aster2)
data(generateddata)
m1 <- aster(resp ~ 0 + varb + mass + timing,
  fam = fam, pred = pred, varvar = varb, idvar = id,
  root = root, data = redata)
target <- c(9:10)
nboot <- 2000; timer <- nboot/2
bar <- eigenbootcanon(m1, nboot = nboot, index = target,
  vectors = c(2), u = 1, m = timer)
bar
## End(Not run)
```

## fit.boot *fit.boot*

### Description

A parametric bootstrap procedure evaluated at an envelope estimator of the submodel mean-value parameter vector $\tau$ that was obtained using reducing subspaces.

### Usage

```
fit.boot(model, nboot, index, vectors = NULL, u = NULL,
  data, amat, newdata, modmat.new = NULL, renewdata = NULL,
  fit.name = NULL, method = c("eigen","1d"), quiet = FALSE, m = 100)
```

### Arguments

`model` An aster model object.

`nboot` The number of bootstrap iterations desired.

`index` The indices denoting which components of the canonical parameter vector are parameters of interest.

`vectors` The indices denoting which reducing subspace of Fisher information is desired to construct envelope estimators.

`u` The envelope model dimension.

`data` An asterdata object corresponding to the original data.

`amat` This object can either be an array or a matrix. It specifies a linear combination of mean-value parameters that correspond to expected Darwinian fitness. See the `aster` function help page in the original `aster` package for more details.

`newdata` A dataframe corresponding to hypothetical individuals in which expected Darwinian fitness is to be estimated.

`modmat.new` A model matrix corresponding to hypothetical individuals in which expected Darwinian fitness is to be estimated.

`renewdata` A dataframe in long format corresponding to hypothetical individuals in which expected Darwinian fitness is to be estimated.

`fit.name` An expression that appears in the name of the nodes that correspond to Darwinian fitness. This is only necessary if `renewdata` is not provided.

`method` The procedure used to obtain envelope estimators.

`quiet` A logical argument. If FALSE, the function displays how much time it takes to run `m` iterations.

`m` The length of the output interval.

## Details

This function implements the parametric bootstrap procedure given by the algorithm presented below with respect to the mean-value parameterization. This parametric bootstrap generates resamples from the distribution evaluated at an envelope estimator of $\tau$.

The user specifies which vectors are used in order to construct envelope estimators using the reducing subspace approach. The user also specifies which method is to be used in order to calculate envelope estimators. When one is using a partial envelope, then this function constructs envelope estimators of $\upsilon$ where we write $\tau = (\gamma^T, \upsilon^T)^T$ and $\upsilon$ corresponds to aster model parameters of interest. In applications, candidate reducing subspaces are indices of eigenvectors of $\widehat{\Sigma}_{\upsilon,\upsilon}$ where $\widehat{\Sigma}_{\upsilon,\upsilon}$ is the part of $\hat{\Sigma}$ corresponding to our parameters of interest. These indices are specified by `vectors`. When all of the components of $\tau$ are components of interest, then we write $\widehat{\Sigma}_{\upsilon,\upsilon} = \widehat{\Sigma}$. When data is generated via the parametric bootstrap, it is the indices (not the original reducing subspaces) that are used to construct envelope estimators constructed using the generated data. The algorithm using reducing subspaces is as follows:

1. Fit the aster model to the data and obtain $\hat{\tau} = (\hat{\gamma}^T, \hat{\upsilon}^T)$ and $\hat{\Sigma}$ from the aster model fit.
2. Compute the envelope estimator of $\upsilon$ in the original sample, given as $\hat{\upsilon}_{env} = P_{\hat{G}}\hat{\upsilon}$ where $P_{\hat{G}}$ is the projection into the reducing subspace of $\widehat{\Sigma}_{\upsilon,\upsilon}$ specified by `vectors`.
3. Perform a parametric bootstrap by generating resamples from the distribution evaluated at $\hat{\tau}_{env} = (\hat{\gamma}^T, \hat{\upsilon}_{env}^T)^T$. For iteration $b = 1, ..., B$ of the procedure:
   (3a) Compute $\hat{\upsilon}^{(b)}$ and $\widehat{\Sigma}^{(b)}_{\upsilon,\upsilon}$ from the aster model fit to the resampled data.
   (3b) Obtain $P^{(b)}_{\hat{G}}$ as done in Step 2.
   (3c) Compute $\hat{\upsilon}^{(b)}_{env} = P^{(b)}_{\hat{G}}\hat{\upsilon}^{(b)}$ and $\hat{\tau}^{(b)}_{env} = (\hat{\gamma}^{(b)^T}, \hat{\upsilon}^{(b)^T}_{env})^T$.
   (3d) Store $h\left(\hat{\tau}^{(b)}_{env}\right)$ where $h$ maps $\tau$ to the parameterization of Darwinian fitness as determined by `amat`.

The parametric bootstrap procedure which uses the 1-d algorithm to construct envelope estimators is analogous to the above algorithm. To use the 1-d algorithm, the user specifies a candidate envelope model dimension `u` and specifies `method = "1d"`. A parametric bootstrap generating resamples from the distribution evaluated at the aster model MLE is also conducted by this function.

## Value

`u` The dimension of the envelope space assumed

`table` A table of output. The first two columns display the envelope estimator of expected Darwinian fitness and its bootstrapped standard error. The next two columns display the MLE of expected Darwinian fitness and its bootstrapped standard error. The last column displays the ratio of the standard errors using MLE to those using envelope estimation. Ratios greater than 1 indicate efficiency gains obtained using envelope estimation.

`S` The bootstrap estimator of the variability of the partial envelope estimator.

`S2` The bootstrap estimator of the variability of the MLE.

`env.boot.out` The realizations from the bootstrap procedure using envelope methodology.

`MLE.boot.out` The realizations from the bootstrap procedure using maximum likelihood estimation.

### References

Cook, R.D. and Zhang, X. (2014). Foundations for Envelope Models and Methods. *JASA*, In Press.

Cook, R.D. and Zhang, X. (2015). Algorithms for Envelope Estimation. *Journal of Computational and Graphical Statistics*, Published online. doi:~10.1080/10618600.2015.1029577.

Eck, D. J., Geyer, C. J., and Cook, R. D. (2016). Enveloping the aster model. *in prep*.

### Examples

```
## Not run: set.seed(13)
library(envlpaster)
library(aster2)
data(simdata30nodes)
data <- simdata30nodes.asterdata
nnode <- length(vars)
xnew <- as.matrix(simdata30nodes[,c(1:nnode)])
m1 <- aster(xnew, root, pred, fam, modmat)
target <- 5:9
indices <- c(1,2,4,5)
u <- length(indices)
nboot <- 2000; timer <- nboot/2
bar <- eigenboot(m1, nboot = nboot, index = target,
  u = u, vectors = indices, data = data, m = timer)
bar
## End(Not run)
```

## fit.boot.Efron *fit.boot.Efron*

### Description

A parametric bootstrap procedure evaluated at an envelope estimator of the submodel mean-value parameter vector $\tau$ that was obtained using reducing subspaces or the 1d algorithm.

### Usage

```
fit.boot.Efron(model, nboot, index, vectors = NULL, dim = NULL,
  data, amat, newdata, modmat.new = NULL, renewdata = NULL,
  criterion = c("AIC","BIC","LRT"), alpha = 0.05, fit.name = NULL,
  method = c("eigen","1d"), quiet = FALSE)
```

### Arguments

| | |
|---|---|
| `model` | An aster model object. |
| `nboot` | The number of bootstrap iterations desired. |
| `index` | The indices denoting which components of the canonical parameter vector are parameters of interest. |

`vectors` The indices denoting which reducing subspace of Fisher information is desired to construct envelope estimators. Must be specified if `method = "eigen"`.

`dim` The dimension of the envelope space used to construct envelope estimators. Must be specified if `method = "1d"`.

`data` An asterdata object corresponding to the original data.

`amat` This object can either be an array or a matrix. It specifies a linear combination of mean-value parameters that correspond to expected Darwinian fitness. See the `aster` function help page in the original `aster` package for more details.

`newdata` A dataframe corresponding to hypothetical individuals in which expected Darwinian fitness is to be estimated.

`modmat.new` A model matrix corresponding to hypothetical individuals in which expected Darwinian fitness is to be estimated.

`renewdata` A dataframe in long format corresponding to hypothetical individuals in which expected Darwinian fitness is to be estimated.

`criterion` A model selection criterion of choice.

`alpha` The type 1 error rate desired for the LRT.

`fit.name` An expression that appears in the name of the nodes that correspond to Darwinian fitness. This is only necessary if `renewdata` is not provided.

`method` The procedure used to obtain envelope estimators.

`quiet` A logical argument. If FALSE, the function displays how much time it takes to run `m` iterations.

## Details

This function implements the first level of the parametric bootstrap procedure given by either Algorithm 1 or Algorithm 2 in Eck (2015) with respect to the mean-value parameterization. This is detailed in Steps 1 through 3d in the algorithm below. This parametric bootstrap generates resamples from the distribution evaluated at an envelope estimator of $\tau$ adjusting for model selection volatility.

The user specifies a model selection criterion which selects vectors that construct envelope estimators using the reducing subspace approach. The user also specifies which method is to be used in order to calculate envelope estimators. When one is using a partial envelope, then this function constructs envelope estimators of $\upsilon$ where we write $\tau = (\gamma^T, \upsilon^T)^T$ and $\upsilon$ corresponds to aster model parameters of interest. In applications, candidate reducing subspaces are indices of eigenvectors of $\widehat{\Sigma}_{\upsilon,\upsilon}$ where $\widehat{\Sigma}_{\upsilon,\upsilon}$ is the part of $\hat{\Sigma}$ corresponding to our parameters of interest. These indices are specified by `vectors`. When all of the components of $\tau$ are components of interest, then we write $\widehat{\Sigma}_{\upsilon,\upsilon} = \widehat{\Sigma}$. When data is generated via the parametric bootstrap, it is the indices (not the original reducing subspaces) that are used to construct envelope estimators constructed using the generated data. The algorithm using reducing subspaces is as follows:

1. Fit the aster model to the data and obtain $\hat{\tau} = (\hat{\gamma}^T, \hat{\upsilon}^T)$ and $\hat{\Sigma}$ from the aster model fit.
2. Compute the envelope estimator of $\upsilon$ in the original sample, given as $\hat{\upsilon}_{env} = P_{\hat{G}}\hat{\upsilon}$ where $P_{\hat{G}}$ is computed using reducing subspaces and selected via a model selection criterion of choice.
3. Perform a parametric bootstrap by generating resamples from the distribution of the aster submodel evaluated at $\hat{\tau}_{env} = (\hat{\gamma}^T, \hat{\upsilon}_{env}^T)^T$. For iteration $b = 1, ..., B$ of the procedure:

- (3a) Compute $\hat{\tau}^{(b)}$ and $\widehat{\Sigma}_{\upsilon,\upsilon}^{(b)}$ from the aster model fit to the resampled data.
- (3b) Build $P_{\hat{G}}^{(b)}$ using the indices of $\hat{\Sigma}_{\upsilon,\upsilon}^{(b)}$ that are selected using the same model selection criterion as Step 2 to build $\hat{G}$.
- (3c) Compute $\hat{\upsilon}_{env}^{(b)} = P_{\hat{\mathcal{E}}}^{(b)} \hat{\upsilon}^{(b)}$ and $\hat{\tau}_{env}^{(b)} = \left(\hat{\gamma}^{(b)^T}, \hat{\upsilon}_{env}^{(b)^T}\right)^T$.
- (3d) Store $\hat{\tau}_{env}^{(b)}$ and $g\left(\hat{\tau}_{env}^{(b)}\right)$ where $g$ maps $\tau$ to the parameterization of Darwinian fitness.

4. After $B$ steps, the bootstrap estimator of expected Darwinian fitness is the average of the envelope estimators stored in Step 3d. This completes the first part of the bootstrap procedure.
5. We now proceed with the second level of bootstrapping at the $b^{th}$ stored envelope estimator $\hat{\tau}_{env}^{(b)}$. For iteration $k = 1, ..., K$ of the procedure:
    - (5a) Generate data from the distribution of the aster submodel evaluated at $\hat{\tau}_{env}^{(b)}$.
    - (5b) Perform Steps 3a through 3d with respect to the dataset obtained in Step 5a.
    - (5c) Store $\hat{\tau}_{env}^{(b)^{(k)}}$ and $g\left(\hat{\tau}_{env}^{(b)^{(k)}}\right)$.

The parametric bootstrap procedure which uses the 1d algorithm to construct envelope estimators is analogous to the above algorithm. To use the 1d algorithm, the user specifies `method = "1d"`. A parametric bootstrap generating resamples from the distribution evaluated at the aster model MLE is also conducted by this function.

## Value

`env.boot.out` Estimated expected Darwinian fitness from generated data obtained from Steps 3a-3d in the bootstrap procedure using the envelope estimator constructed using reducing subspaces.

`MLE.boot.out` Estimated expected Darwinian fitness from generated data obtained from Steps 3a-3d in the bootstrap procedure using the MLE.

`env.1d.boot.out` Estimated expected Darwinian fitness from generated data obtained from Steps 3a-3d in the bootstrap procedure using the envelope estimator constructed using the 1d algorithm.

`env.tau.boot` Estimated mean-value parameter vectors from generated data obtained from Steps 3a-3d in the bootstrap procedure using the envelope estimator constructed using reducing subspaces.

`MLE.tau.boot` Estimated mean-value parameter vectors from generated data obtained from Steps 3a-3d in the bootstrap procedure using the MLE.

`env.1d.tau.boot` Estimated mean-value parameter vectors from generated data obtained from Steps 3a-3d in the bootstrap procedure using the envelope estimator constructed using the 1d algorithm.

`P.list` A list of all estimated projections into the envelope space constructed from reducing subspaces for Steps 3a-3d in the bootstrap procedure.

`P.1d.list` A list of all estimated projections into the envelope space constructed using the 1d algorithm for Steps 3a-3d in the bootstrap procedure.

| | |
|---|---|
| `vectors.list` | A list of indices of eigenvectors used to build the projections in P.list. These indices are selected using the user specified model selection criterion as indicated in Steps 3a-3d in the bootstrap procedure. |
| `u.1d.list` | A list of indices of eigenvectors used to build the projections in P.list. These indices are selected using the user specified model selection criterion as indicated in Steps 3a-3d in the bootstrap procedure. |

### References

Cook, R.D. and Zhang, X. (2014). Foundations for Envelope Models and Methods. *JASA*, In Press.

Cook, R.D. and Zhang, X. (2015). Algorithms for Envelope Estimation. *Journal of Computational and Graphical Statistics*, Published online. doi:~10.1080/10618600.2015.1029577.

Eck, D. J., Geyer, C. J., and Cook, R. D. (2016). Enveloping the aster model. *in prep*.

Eck, D.~J., Geyer, C.~J., and Cook, R.~D. (2016). Web-based Supplementary Materials for "Enveloping the aster model." *in prep*.

Efron, B. (2014). Estimation and Accuracy After Model Selection. *JASA*, **109:507**, 991-1007.

### Examples

```
### see Web-based Supplementary Materials for ``Enveloping the aster model.'' ###
```

## `generateddata` *A generated aster data set with 8 nodes*

### Description

Simulated data for an aster analysis.

### Usage

```
data(generateddata)
```

### Format

A dataframe with records for 400 organisms over 6 years. The description of objects included follows:

**redata** The data frame in long form.

**data** The `asterdata` object.

**vars** Character vector giving the names of the variables in the graph.

**t** Indicates survival for each of the 6 years.

**u** Indicates flowering for each of the 6 years.

**v** Counts flower heads for each of the 6 years.

**w** Indicates seed production for each of the 6 years.

**y** Counts seeds for each of the 6 years.

**z1** The first trait thought to be associated with Darwinian fitness.

**z2** The second trait thought to be associated with Darwinian fitness.

---

## `get1Dderiv` *get1Dderiv*

---

### Description

The derivative of the objective function for the 1D-algorithm.

### Usage

```
get1Dderiv(w,A,B)
```

### Arguments

| | |
|---|---|
| `w` | A vector of length of $p$. |
| `A` | A $\sqrt{n}$ estimate of an estimator's asymptotic covariance matrix. |
| `B` | A $\sqrt{n}$ estimate of the parameter associated with the space we are enveloping. For our purposes this quantity is either the outer product of the MLE of the mean-value submodel parameter vector with itself or the outer product of the MLE of the canonical submodel parameter vector with itself. |

### Details

This function evaluates the derivative of the objective function for the 1D-algorithm at `w`, `A`, and `B`. This is needed in order to reliably find the maximum of the 1D-algorithm objective function.

### Value

| | |
|---|---|
| `dF` | The value of the derivative of the objective function for the 1D-algorithm evaluated at `w`, `A`, and `B`. |

### References

Cook, R.D. and Zhang, X. (2014). Foundations for Envelope Models and Methods. *JASA*, In Press.

Cook, R.D. and Zhang, X. (2015). Algorithms for Envelope Estimation. *Journal of Computational and Graphical Statistics*, Published online. doi:~10.1080/10618600.2015.1029577.

## Examples

```
## Not run: library(envlpaster)
data(simdata30nodes)
data <- simdata30nodes.asterdata
nnode <- length(vars)
xnew <- as.matrix(simdata30nodes[,c(1:nnode)])
m1 <- aster(xnew, root, pred, fam, modmat)
avar <- m1$fisher
beta <- m1$coef
U <- beta %o% beta
get1Dderiv(w = beta, A = avar, B = U)
## End(Not run)
```

---

`get1Dini` *get1Dini*

---

## Description

Finds appropriate starting values for the 1D-algorithm.

## Usage

```
get1Dini(A,B)
```

## Arguments

| | |
|---|---|
| A | A $\sqrt{n}$ estimate of an estimator's asymptotic covariance matrix. |
| B | A $\sqrt{n}$ estimate of the parameter associated with the space we are enveloping. For our purposes this quantity is either the outer product of the MLE of the mean-value submodel parameter vector with itself or the outer product of the MLE of the canonical submodel parameter vector with itself. |

## Details

The 1D-algorithm is sensitive to starting values. Maximizers of the 1D-algorithm objective function are close to eigenvectors of A or (A+B). The vector, w, is the eigenvector of A or (A+B) which produces the largest value of the 1D-algorithm objective function.

## Value

| | |
|---|---|
| w | An appropriate starting value for the 1D-algorithm. |

## References

Cook, R.D. and Zhang, X. (2014). Foundations for Envelope Models and Methods. *JASA*, In Press.

Cook, R.D. and Zhang, X. (2015). Algorithms for Envelope Estimation. *Journal of Computational and Graphical Statistics*, Published online. doi:~10.1080/10618600.2015.1029577.

### Examples

```
## Not run: library(envlpaster)
data(simdata30nodes)
data <- simdata30nodes.asterdata
nnode <- length(vars)
xnew <- as.matrix(simdata30nodes[,c(1:nnode)])
m1 <- aster(xnew, root, pred, fam, modmat)
avar <- m1$fisher
beta <- m1$coef
U <- beta %o% beta
get1Dini(A = avar, B = U)
## End(Not run)
```

## get1Dobj *get1Dobj*

### Description

The objective function for the 1D-algorithm.

### Usage

```
get1Dobj(w,A,B)
```

### Arguments

`w` A vector of length of `p`.

`A` A $\sqrt{n}$ estimate of an estimator's asymptotic covariance matrix.

`B` A $\sqrt{n}$ estimate of the parameter associated with the space we are enveloping. for our purposes this quantity is either the outer product of the MLE of the mean-value submodel parameter vector with itself or the outer product of the MLE of the canonical submodel parameter vector with itself.

### Details

This function evaluates the objective function for the 1D-algorithm at `w`, `A`, and `B`. The maximizer of this objective function is desired for a problem specific `A` and `B`.

### Value

`Fw` The value of the objective function for the 1D-algorithm evaluated at `w`, `A`, and `B`.

### References

Cook, R.D. and Zhang, X. (2014). Foundations for Envelope Models and Methods. *JASA*, In Press.

Cook, R.D. and Zhang, X. (2015). Algorithms for Envelope Estimation. *Journal of Computational and Graphical Statistics*, Published online. doi:~10.1080/10618600.2015.1029577.

### Examples

```
## Not run: library(envlpaster)
data(simdata30nodes)
data <- simdata30nodes.asterdata
nnode <- length(vars)
xnew <- as.matrix(simdata30nodes[,c(1:nnode)])
m1 <- aster(xnew, root, pred, fam, modmat)
avar <- m1$fisher
beta <- m1$coef
U <- beta %o% beta
get1Dobj(w = beta, A = avar, B = U)
## End(Not run)
```

---

## `manifold1Dplus` *manifold1Dplus*

---

### Description

The 1D algorithm

### Usage

```
manifold1Dplus(M,U,u)
```

### Arguments

`M` A $\sqrt{n}$ estimate of an estimator's asymptotic covariance matrix.

`U` A $\sqrt{n}$ estimate of the parameter associated with the space we are enveloping. For our purposes this quantity is either the outer product of the MLE of the mean-value submodel parameter vector with itself or the outer product of the MLE of the canonical submodel parameter vector with itself.

`u` The dimension of the envelope space assumed.

### Details

This function calls `get1Dobj`, `get1Dini`, and `get1Dderiv` in order to find

$$\max_{w} \left[\log(w^T M w) + \log(w^T (M+U) w) - 2\log(w^T w)\right]$$

using Polak-Ribiere conjugate gradient in `optim`. This maximization is conducted a total of `u` times and at each iteration a vector belonging to the envelope space is returned. The vector returned at a specific iteration is orthogonal to the vectors returned at previous iterations. When finished, a basis matrix for the envelope space is returned.

### Value

`G` A $\sqrt{n}$ estimator of the basis matrix for the envelope subspace. This matrix has `u` columns

### References

Cook, R.D. and Zhang, X. (2014). Foundations for Envelope Models and Methods. *JASA*, In Press.

Cook, R.D. and Zhang, X. (2015). Algorithms for Envelope Estimation. *Journal of Computational and Graphical Statistics*, Published online. doi:~10.1080/10618600.2015.1029577.

### Examples

```
## Not run: library(envlpaster)
data(simdata30nodes)
data <- simdata30nodes.asterdata
nnode <- length(vars)
xnew <- as.matrix(simdata30nodes[,c(1:nnode)])
m1 <- aster(xnew, root, pred, fam, modmat)
avar <- m1$fisher
beta <- m1$coef
U <- beta %o% beta
manifold1Dplus(M = avar, U = U, u = 1)
## End(Not run)
```

## matpower *matpower*

### Description

Raises a matrix to a power.

### Usage

```
matpower(a,alpha)
```

### Arguments

| | |
|---|---|
| a | An $n \times n$ matrix |
| alpha | The power which to raise the matrix to. |

### Value

| | |
|---|---|
| ai | The matrix raised to the desired power |

### Examples

```
## Not run: library(envlpaster)
data(simdata30nodes)
data <- simdata30nodes.asterdata
nnode <- length(vars)
xnew <- as.matrix(simdata30nodes[,c(1:nnode)])
m1 <- aster(xnew, root, pred, fam, modmat)
avar <- m1$fisher
matpower(a = avar, alpha = -1)
## End(Not run)
```

## Mguttatus *Mguttatus*

### Description

A dataset with observed characteristics on 2313 Mimulus guttatus flowers.

### Usage

```
data(Mguttatus)
```

### Format

A dataset with observed characteristics on 2313 Mimulus guttatus flowers. The description of objects included follows:

**Mguttatus.redata** The data frame in long form.

**Mguttatus** The `asterdata` object.

**quantities** The quantities required to fit the aster model. The components not described in the aster help section are described below.

**fit** The indices of nodes in the `Mguttatus.redata` dataset that correspond to Darwinian fitness.

**gen_bac** The genetic background of the M. guttatus flower.

**site** The field site of the M. guttatus flower.

**inversion** The inversion orientation of the M. guttatus flower.

**ecotype** The ecotype of the M. guttatus flower.

### References

Lowry, D.~B. and Willis, J.~H. (2010). A Widespread Chromosomal Inversion Polymorphism Contributes to a Major Life-History Transition, Local Adaptation, and Reproductive Isolation. *PLoS Biol*, 8(9): e1000500. doi:~10.1371/journal.pbio.1000500.

## projection *projection*

### Description

Computes a projection into the column space of a user specified matrix.

### Usage

```
projection(a)
```

### Arguments

| | |
|---|---|
| a | An $n \times m$ matrix |

### Value

| | |
|---|---|
| pa | The projection into the column space of a |

### Examples

```
## Not run: library(envlpaster)
data(simdata30nodes)
data <- simdata30nodes.asterdata
nnode <- length(vars)
xnew <- as.matrix(simdata30nodes[,c(1:nnode)])
m1 <- aster(xnew, root, pred, fam, modmat)
avar <- m1$fisher
projection(a = avar)
## End(Not run)
```

## scanner *scanner*

### Description

A diagnostic assessing the potential benefits of envelope methodology for a particular aster model.

### Usage

```
scanner(M, coef, u)
```

### Arguments

| | |
|---|---|
| M | A $\sqrt{n}$ estimate of an estimator's asymptotic covariance matrix. |
| coef | The MLE of the parameter of interest. |
| u | The dimension of the envelope space assumed. |

### Details

This function provides users with a rough diagnostic for the performance of an envelope estimator at a specific dimension. We can see how close a particular potential envelope estimator is to the MLE as well as the proportion of variation that will be discarded when using envelope estimation. This amount of variation discarded is optimistic since it does not account for variability associated with estimating the projection into the envelope space.

## Value

`indices` The indices of the `u` most relevant eigenvectors of `M` to the construction of `coef`.

`table` An output table. The first column is the projection of `coef` into the space spanned by the eigenvectors given by the `indices`.

`G` The `u` most relevant eigenvectors of `M` to the construction of `coef`.

`prop` the sum of the remaining eigenvalues of `M` divided by the sum of all of the eigenvalues of `M`.

## Examples

```
## Not run: library(envlpaster)
data(simdata30nodes)
data <- simdata30nodes.asterdata
nnode <- length(vars)
xnew <- as.matrix(simdata30nodes[,c(1:nnode)])
m1 <- aster(xnew, root, pred, fam, modmat)
avar <- m1$fisher
beta <- m1$coef
scanner(M = avar, coef = beta, u = 1)
## End(Not run)
```

---

# `secondboot` *secondboot*

---

## Description

A parametric bootstrap procedure evaluated at an envelope estimator of the submodel mean-value parameter vector $\tau$ that was obtained using eigenstructures or the 1d algorithm.

## Usage

```
secondboot(k, nboot2, out, model, index, data, amat,
  newdata, method = c("eigen","1d"))
```

## Arguments

`k` The index of the top level parametric bootstrap procedure conducted by `fit.boot.Efron` that the second level of bootstrapping is being applied to.

`nboot2` The bootstrap sample size for the second level of parametric bootstrapping.

`out` The output of `fit.boot.Efron`.

`model` An aster model object.

`index` The indices denoting which components of the canonical parameter vector are parameters of interest.

`data` An asterdata object corresponding to the original data.

`amat` This object can either be an array or a matrix. It specifies a linear combination of mean-value parameters that correspond to expected Darwinian fitness. See the `aster` function help page in the original `aster` package for more details.

`newdata` A dataframe corresponding to hypothetical individuals in which expected Darwinian fitness is to be estimated.

`method` The procedure used to obtain envelope estimators.

## Details

This function implements the second level of the parametric bootstrap procedure given by either Algorithm 1 or Algorithm 2 in Eck (2015) with respect to the mean-value parameterization. This is detailed in Steps 4 through 5c in the algorithm below. At iteration $b$, this parametric bootstrap generates resamples from the distribution evaluated at the envelope estimator ($\hat{\tau}_{env}^{(b)}$) of $\tau$. In this case, the selected indices producing the eigenstructure which was used to construct the envelope estimator $\hat{\tau}_{env}^{(b)}$ are used to construct envelope estimators for the generated data. These resamples are used to estimate the variability of $\hat{\tau}_{env}^{(b)}$. The algorithm using eigenstructures is as follows:

1. Fit the aster model to the data and obtain $\hat{\tau} = (\hat{\gamma}^T, \hat{\upsilon}^T)$ and $\hat{\Sigma}$ from the aster model fit.
2. Compute the envelope estimator of $\upsilon$ in the original sample, given as $\hat{\upsilon}_{env} = P_{\hat{G}}\hat{\upsilon}$ where $P_{\hat{G}}$ is computed using eigenstructures and selected via a model selection criterion of choice.
3. Perform a parametric bootstrap by generating resamples from the distribution of the aster submodel evaluated at $\hat{\tau}_{env} = (\hat{\gamma}^T, \hat{\upsilon}_{env}^T)^T$. For iteration $b = 1, ..., B$ of the procedure:
   (3a) Compute $\hat{\tau}^{(b)}$ and $\widehat{\Sigma}_{\upsilon,\upsilon}^{(b)}$ from the aster model fit to the resampled data.
   (3b) Build $P_{\hat{G}}^{(b)}$ using the indices of $\hat{\Sigma}_{\upsilon,\upsilon}^{(b)}$ that are selected using the same model selection criterion as Step 2 to build $\hat{G}$.
   (3c) Compute $\hat{\upsilon}_{env}^{(b)} = P_{\hat{\mathcal{E}}}^{(b)}\hat{\upsilon}^{(b)}$ and $\hat{\tau}_{env}^{(b)} = \left(\hat{\gamma}^{(b)^T}, \hat{\upsilon}_{env}^{(b)^T}\right)^T$.
   (3d) Store $\hat{\tau}_{env}^{(b)}$ and $g\left(\hat{\tau}_{env}^{(b)}\right)$ where $g$ maps $\tau$ to the parameterization of Darwinian fitness.
4. After $B$ steps, the bootstrap estimator of expected Darwinian fitness is the average of the envelope estimators stored in Step 3d. This completes the first part of the bootstrap procedure.
5. We now proceed with the second level of bootstrapping at the $b^{th}$ stored envelope estimator $\hat{\tau}_{env}^{(b)}$. For iteration $k = 1, ..., K$ of the procedure:
   (5a) Generate data from the distribution of the aster submodel evaluated at $\hat{\tau}_{env}^{(b)}$.
   (5b) Perform Steps 3a through 3d with respect to the dataset obtained in Step 5a.
   (5c) Store $\hat{\tau}_{env}^{(b)^{(k)}}$ and $g\left(\hat{\tau}_{env}^{(b)^{(k)}}\right)$.

When the second level of bootstrapping is completed for all $b = 1, ..., B$ then this function reports the standard deviation of the bootstrapped envelope estimator of expected Darwinian fitness. In this case, the bootstrap procedure accounts for model selection volatility. The bootstrapped envelope estimator is

$$\hat{\mu}_g = \frac{1}{B}\sum_{b=1}^{B} g(\hat{\tau}_{env}^{(b)})$$

where $g(\hat{\tau}_{env}^{(b)})$ are the stored envelope estimators of expected Darwinian fitness in the `env.boot.out` matrix included in the output of `fit.boot.Efron`. The standard deviation of the bootstrapped envelope estimator of expected Darwinian fitness is

$$\sum_{b=1}^{B} \left[ \widehat{cov}^{(b)^T} \hat{V}^{-1} \widehat{cov}^{(b)} \right] / B$$

where $\widehat{cov}^{(b)} = \mathbf{B}^{(b)^T} C^{(b)} / K$ and $\hat{V} = \mathbf{B}^{(b)^T} \mathbf{B}^{(b)} / K$. The matrix $\mathbf{B}^{(b)} \in R^{K \times p}$ has rows given by

$$\hat{\tau}_{env}^{(b)^{(k)}} - \sum_{k=1}^{K} \hat{\tau}_{env}^{(b)^{(k)}} / K$$

and the matrix $C^{(b)} \in R^{K \times d}$ has columns given by

$$g\left(\tau_{env}^{(b)^{(k)}}\right) - g\left(\tau_{env}^{(b)}\right)$$

.

For more details, see Efron (2014) and Eck (2015). The parametric bootstrap procedure which uses the 1d algorithm to construct envelope estimators is analogous to the above algorithm. To use the 1d algorithm, the user specifies `method = "1d"` instead of `method = "eigen"`.

## Value

`sd.Efron` The estimated standard deviation (sd) for estimated expected Darwinian fitness where is estimation is conducted using envelope methodology. This sd accounts for model selection volatility. An eigenvalue decomposition using `eigen` is used internally to calculate this quantity.

`cov` A components needed to construct `sd.Efron` if other numerical methods are desired.

`V` A components needed to construct `sd.Efron` if other numerical methods are desired.

`MLE.tau.boot.subsample`
A components needed to construct `sd.Efron` if other numerical methods are desired.

`est.env.subsample`
A components needed to construct `sd.Efron` if other numerical methods are desired.

## References

Cook, R.D. and Zhang, X. (2014). Foundations for Envelope Models and Methods. *JASA*, In Press.

Cook, R.D. and Zhang, X. (2015). Algorithms for Envelope Estimation. *Journal of Computational and Graphical Statistics*, Published online. doi:~10.1080/10618600.2015.1029577.

Eck, D. J., Geyer, C. J., and Cook, R. D. (2016). Enveloping the aster model. *in prep*.

Eck, D.~J., Geyer, C.~J., and Cook, R.~D. (2016). Web-based Supplementary Materials for "Enveloping the aster model." *in prep*.

Efron, B. (2014). Estimation and Accuracy After Model Selection. *JASA*, **109:507**, 991-1007.

## Examples

```
### Web-based Supplementary Materials for ``Enveloping the aster model.'' ###
```

# selection *selection*

## Description

Finds the envelope eigenspace or dimension that is favored using AIC, BIC, or the LRT at a specified size.

## Usage

```
selection(parm, index, model, data = NULL, alpha = 0.05,
type = c("canonical","mean-value"), method = c("eigen","1d"))
```

## Arguments

`parm` The MLE of the parameter of interest.

`index` The indices denoting which components of the canonical parameter vector are parameters of interest.

`model` An aster model object.

`data` An asterdata object.

`alpha` The desired size of the LRT.

`type` The parameterization of the aster model in which envelope methods are being applied.

`method` The procedure used to obtain envelope estimators.

## Details

This function provides the user with the envelope model dimension or indices of the eigenspace favored by AIC, BIC, and the likelihood ratio test of size `alpha`. There are four possible combinations of outputs. They are:

1. The specification of `method = "eigen"` and `type = "mean-value"` provides the user with the indices of the eigenspace of estimated Fisher information used to construct an envelope estimator for $\tau$ favored by AIC, BIC, and the LRT of size `alpha`.
2. The specification of `method = "eigen"` and `type = "canonical"` provides the user with the indices of the eigenspace of estimated Fisher information used to construct an envelope estimator for $\beta$ favored by AIC, BIC, and the LRT of size `alpha`.
3. The specification of `method = "1d"` and `type = "mean-value"` provides the user with the envelope model dimension used to construct an envelope estimator for $\tau$ favored by AIC, BIC, and the LRT of size `alpha`.

4. The specification of `method = "1d"` and `type = "canonical"` provides the user with the envelope model dimension used to construct an envelope estimator for $\beta$ favored by AIC, BIC, and the LRT of size `alpha`.

When one is interested in envelope model dimensions or eigenspaces with respect to $\beta$, then an `asterdata` object does not need to be specified. On the other hand, an `asterdata` is needed in order to map the estimated $\tau$ to its corresponding $\beta$ value. This is necessary because of the interface (or lack thereof) between current `aster` and `aster2` software. The way in which aster model log likelihoods are evaluated is incorporated in `aster` software and changing parameterizations is carried out using `aster2` software.

## Value

`aic` The eigenspace or envelope model dimension favored using AIC.

`bic` The eigenspace or envelope model dimension favored using BIC.

`LRT` The eigenspace or envelope model dimension favored using the LRT of size `alpha`.

`out` The output table of all model selection criteria for all envelope estimators considered.

## References

Cook, R.D. and Zhang, X. (2014). Foundations for Envelope Models and Methods. *JASA*, In Press.

Cook, R.D. and Zhang, X. (2015). Algorithms for Envelope Estimation. *Journal of Computational and Graphical Statistics*, Published online. doi:~10.1080/10618600.2015.1029577.

Eck, D. J., Geyer, C. J., and Cook, R. D. (2016). Enveloping the aster model. *in prep*.

## Examples

```
## Not run:
  set.seed(13)
  library(envlpaster)
  library(aster2)
  data(generateddata)
  m.null <- aster(resp ~ 0 + varb, fam = fam, pred = pred,
    varvar = varb, idvar = id, root = root, data = redata)
  m1 <- aster(resp ~ 0 + varb + mass + timing,
    fam = fam, pred = pred, varvar = varb, idvar = id,
    root = root, data = redata)
  m2 <- aster(resp ~ 0 + varb + mass + timing +
    I(mass^2) + I(timing^2) + I(mass*timing),
    fam = fam, pred = pred, varvar = varb, idvar = id,
    root = root, data = redata)
  anova.table <- anova(m.null,m1,m2); anova.table
  beta <- m1$coef
  a <- grepl( "offsp", names(beta))
  a <- a + grepl( "surviv", names(beta))
  b <- which(a == 1)
  target <- c(1:length(beta))[-b]
  nnode <- ncol(m1$x)
  data.aster <- asterdata(data, vars, pred, rep(0,nnode),
    fam, families = list("bernoulli", "poisson",
    fam.zero.truncated.poisson()))
  selection(parm  = beta, index = target, model = m1,
    data = data.aster, alpha = 0.05, type = "canonical",
    method = "eigen")

## End(Not run)
```

## `simdata30nodes` *A generated aster data set with 30 nodes*

### Description

Simulated data for an aster analysis. Loads 7 objects.

### Usage

```
data(simdata30nodes)
```

### Format

The data frame with records for 3000 organisms over 10 years. The dataset corresponding to our aster analysis. The following four descriptions explain the elements of this dataset.

**u** Indicates survival for each of the 10 years.

**w** Counts offspring for each of the 10 years.

**v** Indicates if `w > 0` for each of the 10 years.

**z** A covariate of potential interest, 10 in total.

**variables** Character vector giving the names of the variables in the graph.

**root** The root data. For `aster.default` an `nind` by `nnode` matrix, for `aster.formula` an `nind * nnode` vector.

**modmat** An `nind` by `nnode` by `ncoef` three-dimensional array, the model matrix. `aster.formula` constructs such a modmat from its formula, the data frame data, and the variables in the environment of the formula.

**formula** Necessary for changing to class `aster.formula`.

**xlevels** Necessary for changing to class `aster.formula`.

**terms** Necessary for changing to class `aster.formula`.

**simdata30nodes.asterdata** An object of class `asterdata` corresponding to `simdata30nodes`.

## Details

This object contains an aster data set in wide form, an object of class `asterdata` corresponding to the original data set, and vectors specifying the graphical structure of the aster model.

There are 3000 simulated individuals in this aster analysis. Our data is generated in two parts. The first part follows Technical report 671 (TR 671) on Charlie Geyer's Aster Models for Life History Analysis webpage. For our data, `nind = 3000`, `ntime = 10`, `psurv = 0.95`, `prepr = 0.7`, `mpois = 1`, and the seed is set at `set.seed(13)` which is different from the original simulation setup.

We follow the model construction in TR 671 through `out6`. We then generate a new dataset from the aster model where the components of the submodel mean-value parameter vector $\tau$ corresponding to Darwinian fitness is in the space spanned by the first, second, and fourth eigenvectors of Fisher information.

## References

Geyer, C. J. and Shaw, R. G. (2009). Model Selection in Estimation of Fitness Landscapes. Technical Report No. 671. School of Statistics, University of Minnesota. `http://conservancy.umn.edu/handle/11299/56219`.

# `targetboot` *targetboot*

## Description

A parametric bootstrap procedure evaluated at an envelope estimator of the submodel mean-value parameter vector $\tau$ that was obtained using the 1D algorithm.

## Usage

```
targetboot(model, nboot, index, u, data,
  quiet = FALSE, m = 100)
```

## Arguments

`model` An aster model object

`nboot` The number of bootstrap iterations desired

`index` The indices denoting which components of the canonical parameter vector are parameters of interest

`u` The dimension of the envelope space assumed

`data` An asterdata object

`quiet` A logical argument. If FALSE, the function displays how much time it takes to run `m` iterations

`m` The length of the output interval

## Details

This function implements the parametric bootstrap procedure given by the algorithm below with respect to the mean-value parameterization. This parametric bootstrap generates resamples from the distribution evaluated at an envelope estimator of $\tau$. Envelope estimators are constructed using the 1D algorithm at a user-specified envelope model dimension `u`. When one is using a partial envelope then this function constructs envelope estimators of $\upsilon$ where we write $\tau = (\gamma^T, \upsilon^T)^T$ and $\upsilon$ corresponds to aster model parameters of interest. In the sample, the 1D algorithm uses $M = \widehat{\Sigma}_{\upsilon,\upsilon}$ and $U = \hat{\tau}\hat{\tau}^T$ as inputs where $\widehat{\Sigma}_{\upsilon,\upsilon}$ is the part of $\widehat{\Sigma}$ corresponding to our parameters of interest. When all of the components of $\tau$ are components of interest, then we write $\widehat{\Sigma}_{\upsilon,\upsilon} = \widehat{\Sigma}$. The algorithm is as follows:

1. Fit the aster model to the data and obtain $\hat{\tau} = (\hat{\gamma}^T, \hat{\upsilon}^T)$ and $\widehat{\Sigma}$ from the aster model fit.
2. Compute the envelope estimator of $\upsilon$ in the original sample, given as $\hat{\upsilon}_{env} = P_{\hat{\mathcal{E}}}\hat{\upsilon}$ where $P_{\hat{\mathcal{E}}}$ is obtained from the 1D algorithm.
3. Perform a parametric bootstrap by generating resamples from the distribution evaluated at $\hat{\upsilon}_{env}$. For iteration $b = 1, ..., B$ of the procedure:
   (3a) Compute $\hat{\upsilon}^{(b)}$ and $\widehat{\Sigma}_{\upsilon,\upsilon}^{(b)}$ from the aster model fit to the resampled data.
   (3b) Obtain $P_{\hat{\mathcal{E}}}^{(b)}$ as done in Step 2.
   (3c) Store $\hat{\upsilon}_{env}^{(b)} = P_{\hat{\mathcal{E}}}^{(b)}\hat{\upsilon}^{(b)}$.

A parametric bootstrap generating resamples from the distribution evaluated at the aster model MLE is also conducted by this function.

## Value

`u` The dimension of the envelope space assumed

`table` A table of output. The first two columns display the envelope estimator and its bootstrapped standard error. The next two columns display the MLE and its bootstrapped standard error. The last column displays the ratio of the standard error for the bootstrapped envelope estimator to the standard error for the bootstrapped MLE.

`S` The bootstrap estimator of the variability of the partial envelope estimator.

`S2` The bootstrap estimator of the variability of the MLE.

`env.boot.out` The realizations from the bootstrap procedure using envelope methodology.

`MLE.boot.out` The realizations from the bootstrap procedure using maximum likelihood estimation.

## References

Cook, R.D. and Zhang, X. (2014). Foundations for Envelope Models and Methods. *JASA*, In Press.

Cook, R.D. and Zhang, X. (2015). Algorithms for Envelope Estimation. *Journal of Computational and Graphical Statistics*, Published online. doi:~10.1080/10618600.2015.1029577.

Eck, D. J., Geyer, C. J., and Cook, R. D. (2016). Enveloping the aster model. *in prep*.

## Examples

```
## Not run: set.seed(13)
library(envlpaster)
library(aster2)
data(simdata30nodes)
data <- simdata30nodes.asterdata
nnode <- length(vars)
xnew <- as.matrix(simdata30nodes[,c(1:nnode)])
m1 <- aster(xnew, root, pred, fam, modmat)
target <- 5:9
indices <- c(1,2,4,5)
u <- length(indices)
nboot <- 2000; timer <- nboot/2
bar <- targetboot(m1, nboot = nboot, index = target,
  u = u, data = data, m = timer)
bar
## End(Not run)
```

---

# targetbootcanon *targetbootcanon*

---

## Description

A parametric bootstrap procedure evaluated at an envelope estimator of the submodel canonical parameter vector $\beta$ that was obtained using the 1D algorithm.

## Usage

```
targetbootcanon(model, nboot, index, u,
  code, families, quiet = FALSE, m = 100)
```

## Arguments

| | |
|---|---|
| model | An aster model object. |
| nboot | The number of bootstrap iterations desired. |
| index | The indices denoting which components of the canonical parameter vector are parameters of interest. |
| u | The dimension of the envelope space assumed. |
| code | A vector of dimension equal to the number of nodes in the aster graph. This vector specifies which exponential family in the `families` list labels each arrow of the aster graphical structure. |
| families | A list of family specifications. |
| quiet | A logical argument. If FALSE, the function displays how much time it takes to run `m` iterations. |
| m | The length of the output interval. |

## Details

This function implements the parametric bootstrap procedure given by the algorithm below with respect to the canonical parameterization. This parametric bootstrap generates resamples from the distribution evaluated at an envelope estimator of $\beta$. Envelope estimators are constructed using the 1D algorithm at a user-specified envelope model dimension `u`. When one is using a partial envelope then this function constructs envelope estimators of $\upsilon$ where we write $\tau = (\gamma^T, \upsilon^T)^T$ and $\upsilon$ corresponds to aster model parameters of interest. In the sample, the 1D algorithm uses $M = \widehat{\Sigma}_{\upsilon,\upsilon}^{-1}$ and $U = \hat{\beta}\hat{\beta}^T$ as inputs where $\widehat{\Sigma}_{\upsilon,\upsilon}^{-1}$ is the part of $\hat{\Sigma}^{-1}$ corresponding to our parameters of interest. When all of the components of $\tau$ are components of interest, then we write $\widehat{\Sigma}_{\upsilon,\upsilon}^{-1} = \widehat{\Sigma}^{-1}$. The algorithm is as follows:

1. Fit the aster model to the data and obtain $\hat{\tau} = (\hat{\gamma}^T, \hat{\upsilon}^T)$ and $\widehat{\Sigma}^{-1}$ from the aster model fit.
2. Compute the envelope estimator of $\upsilon$ in the original sample, given as $\hat{\upsilon}_{env} = P_{\hat{\mathcal{E}}}\hat{\upsilon}$ where $P_{\hat{\mathcal{E}}}$ is obtained from the 1D algorithm.
3. Perform a parametric bootstrap by generating resamples from the distribution evaluated at $\hat{\upsilon}_{env}$. For iteration $b = 1, ..., B$ of the procedure:
   (3a) Compute $\hat{\upsilon}^{(b)}$ and $\widehat{\Sigma}_{\upsilon,\upsilon}^{(b)^{-1}}$ from the aster model fit to the resampled data.
   (3b) Obtain $P_{\hat{\mathcal{E}}}^{(b)}$ as done in Step 2.
   (3c) Store $\hat{\upsilon}_{env}^{(b)} = P_{\hat{\mathcal{E}}}^{(b)}\hat{\upsilon}^{(b)}$.

A parametric bootstrap generating resamples from the distribution evaluated at the aster model MLE is also conducted by this function.

## Value

`u` The dimension of the envelope space assumed.

`table` A table of output. The first two columns display the envelope estimator and its bootstrapped standard error. The next two columns display the MLE and its bootstrapped standard error. The last column displays the ratio of the standard error for the bootstrapped envelope estimator to the standard error for the bootstrapped MLE.

`S` The bootstrap estimator of the variability of the partial envelope estimator.

`S2` The bootstrap estimator of the variability of the MLE.

`env.boot.out` The realizations from the bootstrap procedure using envelope methodology.

`MLE.boot.out` The realizations from the bootstrap procedure using maximum likelihood estimation.

## References

Cook, R.D. and Zhang, X. (2014). Foundations for Envelope Models and Methods. *JASA*, In Press.

Cook, R.D. and Zhang, X. (2015). Algorithms for Envelope Estimation. *Journal of Computational and Graphical Statistics*, Published online. doi:~10.1080/10618600.2015.1029577.

Eck, D. J., Geyer, C. J., and Cook, R. D. (2016). Enveloping the aster model. *in prep*.

## Examples

```
## Not run: set.seed(13)
library(envlpaster)
library(aster2)
data(generateddata)
m1 <- aster(resp ~ 0 + varb + mass + timing,
  fam = fam, pred = pred, varvar = varb, idvar = id,
  root = root, data = redata)
target <- c(9:10)
nboot <- 2000; timer <- nboot/2
bar <- targetbootcanon(m1, nboot = nboot, index = target,
  u = 1, m = timer)
bar
## End(Not run)
```

# Index