

Package ‘changepointsHD’

March 16, 2019

Type Package

Title Change-Point Estimation for Expensive and High-Dimensional Models

Version 0.3.1

Date 2019-03-16

Author Leland Bybee

Maintainer Leland Bybee <lelandb@umich.edu>

Description This implements the methods developed in, L. Bybee and Y. Atchade. (2018). Contains a series of methods for estimating change-points given user specified black-box models. The methods include binary segmentation for multiple change-point estimation. For estimating each individual change-point the package includes simulated annealing, brute force, and, for Gaussian graphical models, an applications specific rank-one update implementation. Additionally, code for estimating Gaussian graphical models is included. The goal of this package is to allow for the efficient estimation of change-points in complicated models with high dimensional data.

License GPL (>= 2)

Imports Rcpp (>= 0.12.3), methods

LinkingTo RcppArmadillo, Rcpp

RoxygenNote 6.0.1

NeedsCompilation yes

Repository CRAN

Date/Publication 2019-03-16 18:23:19 UTC

R topics documented:

changepointsHD-package	2
bbmod_method	3
binary_segmentation	3
brute_force	5
changepointsMod-class	6
log_likelihood_method	7

prox_gradient_ll	8
prox_gradient_mapping	8
rank_one	9
simulated_annealing	10

Index	12
--------------	-----------

changepointsHD-package

Change-Point Estimation for Expensive and High-Dimensional Models

Description

This implements the methods developed in, L. Bybee and Y. Atchade. (2018). Contains a series of methods for estimating change-points given user specified black-box models. The methods include binary segmentation for multiple change-point estimation. For estimating each individual change-point the package includes simulated annealing, brute force, and, for Gaussian graphical models, an applications specific rank-one update implementation. Additionally, code for estimating Gaussian graphical models is included. The goal of this package is to allow for the efficient estimation of change-points in complicated models with high dimensional data.

Details

Package: changepointsHD
 Type: Package
 Version: 0.3.0
 Date: 2017-11-06
 License: GPL-2

Author(s)

Leland Bybee

Maintainer: Leland Bybee <lelandb@umich.edu>

References

1. L. Bybee and Y. Atchade: "Change-Point Computation for Large Graphical Models: A Scalable Algorithm for Gaussian Graphical Models with Change-Points", 2018; [<http://jmlr.csail.mit.edu/papers/volume19/17-218/17-218.pdf>].

bbmod_method	<i>Wrapper method for black-box estimation.</i>
--------------	---

Description

Applies the black-box estimator to the specified partition given the current tau value. Additionally, this wrapper handles the different data structures possible for `part_values` and `whole_values`.

Usage

```
bbmod_method(object, part, tau)

## S4 method for signature 'changePointsMod'
bbmod_method(object, part, tau)
```

Arguments

<code>object</code>	Corresponding <code>changePointsMod</code> class.
<code>part</code>	Index for current partition, should be 1 or 2.
<code>tau</code>	Current change-point. Should be between <code>buff</code> and <code>N - buff</code> .

Value

An updated version of the change-point model. There are currently three possible updates depending on the form of the `part_values` and `whole_values` provided. 1) If only `part_values` are provided, then we assume the black-box method only updates `part_values`. 2) If only `whole_values` are provided, we assume the black-box method only updates `whole_values`. 3) If both `part_values` and `whole_values` are provided, we assume that both are updated.

Author(s)

Leland Bybee <lelandb@umich.edu>

<code>binary_segmentation</code>	<i>Multiple change-point method.</i>
----------------------------------	--------------------------------------

Description

Estimates multiple change-points using the binary-segmentation method. This does a breadth first search and uses the specified single change-point method for each sub-search.

Usage

```
binary_segmentation(object, method, thresh = 0, buff = 100,
  method_params = list())

## S4 method for signature 'changepointsMod'
binary_segmentation(object, method, thresh = 0,
  buff = 100, method_params = list())
```

Arguments

object	Corresponding changepointsMod class.
method	changepointHD method for finding single change-point.
thresh	Stopping threshold for cost comparison.
buff	Distance from edge of sample to be maintained during search.
method_params	List of additional parameters for method.

Value

An updated version of the change-point model. The update will effect: 1) An estimate for the current set of change-points. 2) The `mod_list`, this will correspond to all the active single change-point models generated during the binary-segmentation procedure. Active models correspond to models that have not been superseded by more granular models. 3) The `mod_range`, this corresponds to the range of observations covered by each model. It can be used to determine which models are active.

Author(s)

Leland Bybee <lelandb@umich.edu>

Examples

```
set.seed(334)

mcp_data = read.table(system.file("extdata", "mcp.txt", package="changepointsHD"))
mcp_data = as.matrix(mcp_data)

# prox gradient black-box method
cov_est = cov(mcp_data)
init = solve(cov_est)
res_map = prox_gradient_mapping(mcp_data, init, 0.1, 0.99, 0.1, 100, 1e-20)

# prox gradient black-box ll
res_ll = prox_gradient_ll(mcp_data, res_map, 0.1)

prox_gradient_params=list()
prox_gradient_params$update_w = 0.1
prox_gradient_params$update_change = 0.99
prox_gradient_params$regularizer = 0.1
prox_gradient_params$max_iter = 1
prox_gradient_params$tol = 1e-5
```

```

prox_gradient_ll_params=list()
prox_gradient_ll_params$regularizer = 0.1

simulated_annealing_params = list()
simulated_annealing_params$buff=10

changepoints_mod = changepointsMod(bbmod=prox_gradient_mapping,
                                   log_likelihood=prox_gradient_ll,
                                   bbmod_params=prox_gradient_params,
                                   ll_params=prox_gradient_ll_params,
                                   part_values=list(init, init),
                                   data=list(mcp_data))

changepoints_mod = binary_segmentation(changepoints_mod, method=simulated_annealing,
                                       thresh=0, buff=10,
                                       method_params=simulated_annealing_params)

```

brute_force	<i>Single change-point brute force method.</i>
-------------	--

Description

Estimates a single change-point by testing all possible change-points.

Usage

```

brute_force(object, niter = 1, buff = 100)

## S4 method for signature 'changepointsMod'
brute_force(object, niter = 1, buff = 100)

```

Arguments

object	Corresponding changepointsMod class.
niter	Number of iterations at each possible change-point.
buff	Distance from edge of sample to be maintained during search.

Value

An updated version of the change-point model. The update will effect: 1) the `part_values` and/or `whole_values` (depending on the initial values provided). 2) An estimate for the current change-point. 3) The trace for the search.

Author(s)

Leland Bybee <lelandb@umich.edu>

Examples

```

set.seed(334)

scp_data = read.table(system.file("extdata", "scp.txt", package="changepointsHD"))
scp_data = as.matrix(scp_data)

# prox gradient black-box method
cov_est = cov(scp_data)
init = solve(cov_est)
res_map = prox_gradient_mapping(scp_data, init, 0.1, 0.99, 0.1, 100, 1e-20)

# prox gradient black-box ll
res_ll = prox_gradient_ll(scp_data, res_map, 0.1)

prox_gradient_params=list()
prox_gradient_params$update_w = 0.1
prox_gradient_params$update_change = 0.99
prox_gradient_params$regularizer = 0.1
prox_gradient_params$max_iter = 1
prox_gradient_params$tol = 1e-5

prox_gradient_ll_params=list()
prox_gradient_ll_params$regularizer = 0.1

changepoints_mod = changepointsMod(bbmod=prox_gradient_mapping,
                                   log_likelihood=prox_gradient_ll,
                                   bbmod_params=prox_gradient_params,
                                   ll_params=prox_gradient_ll_params,
                                   part_values=list(init, init),
                                   data=list(scp_data))
changepoints_mod = brute_force(changepoints_mod, buff=10)

```

changepointsMod-class *An S4 class corresponding to the change-point model.*

Description

An S4 class corresponding to the change-point model.

Slots

data A list containing the data for the change-point model. The exact structure of the data is dependent on the `bbmod` and `log_likelihood` provided. In cases where the data is fairly simple, it should still be wrapped with a list, e.g. `X = list(X)`, to allow `changepointsMod` to handle it properly.

part_values A list containing the values estimated by `bbmod`. `part_values`, in particular, contain values that are updated independently for each partition (as opposed to `whole_values`).

whole_values A list containing the values estimated by `bbmod`. whole values, in particular, contain values that are shared between partitions (as opposed to `part_values`).
bbmod An R function for performing the black-box estimation.
bbmod_params A list containing any additional parameters for `bbmod`.
log_likelihood An R function for estimating the log-likelihood for the corresponding `bbmod`.
ll_params A list containing any additional parameters for `log_likelihood`.
trace A vector corresponding to the trace of the estimated change-points based on the method used.
changepts A scalar/vector corresponding to the changepoint(s) estimated based on the method used.
mod_list A list corresponding to all the active single change-point models used with `binary_segmentation`.
mod_range A list of the range of observations corresponding to each active model for `binary_segmentation`.

Author(s)

Leland Bybee <lelandb@umich.edu>

log_likelihood_method *Wrapper method for log-likelihood estimation.*

Description

Generates the log-likelihood for the specified partition given the current tau value. Additionally, this wrapper handles the different data structures possible for `part_values` and `whole_values`.

Usage

```
log_likelihood_method(object, part, tau)

## S4 method for signature 'changeptsMod'
log_likelihood_method(object, part, tau)
```

Arguments

<code>object</code>	Corresponding <code>changeptsMod</code> class.
<code>part</code>	Index for current partition, should be 1 or 2.
<code>tau</code>	Current change-point. Should be between <code>buff</code> and <code>N - buff</code> .

Value

The log-likelihood estimate for the current state. There are currently three possible versions depending on the form of the `part_values` and `whole_values` provided. 1) If only `part_values` are provided, then we assume the log-likelihood takes only the `part_values`. 2) If only `whole_values` are provide, we assume the log-likelihood takes only the `whole_values`. 3) If both `part_values` and `whole_values` are provided, we assume that the log-likelihood takes both.

Author(s)

Leland Bybee <lelandb@umich.edu>

prox_gradient_ll *Proximal-gradient log-likelihood estimator.*

Description

Estimates the log-likelihood for the corresponding precision matrix and data set.

Usage

```
prox_gradient_ll(data, theta_i, regularizer)
```

Arguments

data	N x P matrix corresponding to the raw data.
theta_i	Estimate for precision.
regularizer	Regularizing constant, lambda.

Value

Log-likelihood estimate.

Author(s)

Leland Bybee <lelandb@umich.edu>

prox_gradient_mapping *Proximal-gradient mapping method.*

Description

Performs the proximal-gradient mapping operation to estimate a regularized version of the inverse cov. matrix. Follows the procedure described in, http://dept.stat.lsa.umich.edu/~yvesa/sto_prox.pdf

Usage

```
prox_gradient_mapping(data, theta_start, update_w, update_change, regularizer,  
max_iter, tol)
```


Arguments

data	N x P matrix corresponding to the raw data.
theta_start	Initial value for precision estimate.
update_w	Step size for prox-gradient mapping.
update_change	Proportion of update_w to keep when the algorithm fails to successfully estimate precision.
regularizer	Regularizing constant, lambda.
max_iter	Number of mapping iterations.
tol	Tolerance at which the algorithm stops running.

Value

Theta (precision matrix) estimate.

Author(s)

Leland Bybee <lelandb@umich.edu>

rank_one	<i>Rank one update single change-point estimation.</i>
----------	--

Description

This is a method for estimating a single-changepoint which takes advantage of the special structure of the Gaussian graphical model. It cannot take arbitrary black-box models like `simulated_annealing` or `brute_force`. However, it can still be run within `binary_segmentation`.

Usage

```
rank_one(data, theta_init, buff = 10L, regularizer = 1, tau = -1L,
         max_iter = 25L, update_w = 1, update_change = 0.9, mapping_iter = 1L,
         tol = 1e-05)
```

Arguments

data	N x P Matrix corresponding to the raw data.
theta_init	Initial value for theta estimate.
buff	Distance to maintain from edge of sample.
regularizer	Regularizing constant, lambda.
tau	Initial Estimate for change-point.
max_iter	Maximum number of rank-one updates to be run.
update_w	Step size for prox-gradient.
update_change	Proportion of update_w to keep when the algorithm fails to successfully estimate theta.
mapping_iter	Number of mapping iterations.
tol	Tolerance at which the algorithm stops running.

Value

List containing the estimated change-point and theta values.

Author(s)

Leland Bybee <lelandb@umich.edu>

simulated_annealing *Single change-point simulated annealing method*

Description

Estimates a single change-point using the simulated annealing method.

Usage

```
simulated_annealing(object, niter = 500, min_beta = 1e-04, buff = 100)
```

```
## S4 method for signature 'changeptsMod'  
simulated_annealing(object, niter = 500,  
  min_beta = 1e-04, buff = 100)
```

Arguments

object	Corresponding changeptsMod class.
niter	Number of simulated annealing iterations.
min_beta	Lowest temperature.
buff	Distance from edge of sample to be maintained during search.

Value

An updated version of the change-point model. The update will effect: 1) the `part_values` and/or `whole_values` (depending on the initial values provided). 2) An estimate for the current change-point. 3) The trace for the search.

Author(s)

Leland Bybee <lelandb@umich.edu>

Examples

```
set.seed(334)

scp_data = read.table(system.file("extdata", "scp.txt", package="changepointsHD"))
scp_data = as.matrix(scp_data)

# prox gradient black-box method
cov_est = cov(scp_data)
init = solve(cov_est)
res_map = prox_gradient_mapping(scp_data, init, 0.1, 0.99, 0.1, 100, 1e-20)

# prox gradient black-box ll
res_ll = prox_gradient_ll(scp_data, res_map, 0.1)

prox_gradient_params=list()
prox_gradient_params$update_w = 0.1
prox_gradient_params$update_change = 0.99
prox_gradient_params$regularizer = 0.1
prox_gradient_params$max_iter = 1
prox_gradient_params$tol = 1e-5

prox_gradient_ll_params=list()
prox_gradient_ll_params$regularizer = 0.1

changepoints_mod = changepointsMod(bbmod=prox_gradient_mapping,
                                   log_likelihood=prox_gradient_ll,
                                   bbmod_params=prox_gradient_params,
                                   ll_params=prox_gradient_ll_params,
                                   part_values=list(init, init),
                                   data=list(scp_data))
changepoints_mod = simulated_annealing(changepoints_mod, buff=10)
```

Index

*Topic **package**

changepointsHD-package, 2

bbmod_method, 3

bbmod_method, changepointsMod-method
(bbmod_method), 3

binary_segmentation, 3

binary_segmentation, changepointsMod-method
(binary_segmentation), 3

brute_force, 5

brute_force, changepointsMod-method
(brute_force), 5

changepointsHD

(changepointsHD-package), 2

changepointsHD-package, 2

changepointsMod

(changepointsMod-class), 6

changepointsMod-class, 6

log_likelihood_method, 7

log_likelihood_method, changepointsMod-method
(log_likelihood_method), 7

prox_gradient_ll, 8

prox_gradient_mapping, 8

rank_one, 9

simulated_annealing, 10

simulated_annealing, changepointsMod-method
(simulated_annealing), 10