# Package ‘biotmle’

April 15, 2019

**Title** Targeted Learning with Moderated Statistics for Biomarker Discovery

**Version** 1.6.0

**Maintainer** Nima Hejazi <nhejazi@berkeley.edu>

**Author** Nima Hejazi [aut, cre, cph], Alan Hubbard [aut, ths], Weixin Cai [ctb]

**Description** This package facilitates the discovery of biomarkers from biological sequencing data (e.g., microarrays, RNA-seq) based on the associations of potential biomarkers with exposure and outcome variables by implementing an estimation procedure that combines a generalization of moderated statistics with targeted minimum loss-based estimates (TMLE) of parameters defined via causal inference (e.g., Average Treatment Effect) whose estimators admit asymptotically linear representations.

**Depends** R (>= 3.4)

**License** file LICENSE

**URL** https://github.com/nhejazi/biotmle

**BugReports** https://github.com/nhejazi/biotmle/issues

**Encoding** UTF-8

**LazyData** true

**Imports** dplyr, ggplot2, ggsci, superheat, doFuture, future, stats, methods, limma, S4Vectors, BiocGenerics, BiocParallel, SummarizedExperiment, tmle

**Suggests** testthat, knitr, rmarkdown, BiocStyle, SuperLearner, Matrix, DBI, biotmleData (>= 1.1.1)

**VignetteBuilder** knitr

**RoxygenNote** 6.1.0.9000

**biocViews** GeneExpression, DifferentialExpression, Sequencing, Microarray, RNASeq, ImmunoOncology

**git_url** https://git.bioconductor.org/packages/biotmle

**git_branch** RELEASE_3_8

**git_last_commit** 46cf1af

**git_last_commit_date** 2018-11-29

**Date/Publication** 2019-04-15

## R topics documented:

---

`biomarkertmle` *Biomarker Evaluation with Targeted Minimum Loss-Based Estimation of the ATE*

---

### Description

Computes the causal target parameter defined as the difference between the biomarker expression values under treatment and those same values under no treatment, using Targeted Minimum Loss-Based Estimation.

### Usage

```
biomarkertmle(se, varInt, ngscounts = FALSE, parallel = TRUE,
  bppar_type = NULL, future_param = NULL, family = "gaussian",
  subj_ids = NULL, g_lib = c("SL.glm", "SL.randomForest", "SL.nnet",
  "SL.polymars", "SL.mean"), Q_lib = c("SL.glm", "SL.randomForest",
  "SL.nnet", "SL.mean"))
```

### Arguments

| | |
|---|---|
| `se` | (SummarizedExperiment) - containing expression or next-generation sequencing data in the "assays" slot and a matrix of phenotype-level data in the "colData" slot. |
| `varInt` | (numeric) - indicating the column of the design matrix corresponding to the treatment or outcome of interest (in the `colData` slot of the `SummarizedExperiment` argument "se"). |
| `ngscounts` | (logical) - whether the data are counts generated from a next-generation sequencing (NGS) experiment (e.g., RNA-seq). The default setting assumes continuous expression measures as generated by platforms that are microarray-type (i.e., so-called "targeted" assays). |
| `parallel` | (logical) - whether or not to use parallelization in the estimation procedure. Invoking parallelization happens through a combination of calls to `future` and `BiocParallel`. If this argument is set to `TRUE`, `future::multiprocess` is used, and if `FALSE`, `future::sequential` is used, alongside `BiocParallel::bplapply`. Other options for evaluation through futures may be invoked by setting the argument `future_param`. |

| | |
|---|---|
| bppar_type | (character) - specifies the type of backend to be used with the parallelization invoked by BiocParallel. Consult the manual page for BiocParallel::BiocParallelParam for possible types and descriptions on their appropriate uses. The default for this argument is NULL, which silently uses BiocParallel::DoparParam. |
| future_param | (character) - specifies the type of parallelization to be invoked when using futures for evaluation. For a list of the available types, please consult the documentation for future::plan. The default setting (this argument set to NULL) silently invokes future::multiprocess. Be careful if changing this setting. |
| family | (character) - specification of error family: "binomial" or "gaussian". |
| subj_ids | (numeric vector) - subject IDs to be passed directly to subject should have the exact same numerical identifier; coerced to class numeric if not provided in the appropriate form. |
| g_lib | (char vector) - library of learning algorithms to be used in fitting the propensity score (the nuisance parameter denoted "g" in the literature on targeted minimum loss-based estimation). |
| Q_lib | (char vector) - library of learning algorithms to be used in fitting the outcome regression (the nuisance parameter denoted "Q" in the literature on targeted minimum loss-based estimation). |

## Value

S4 object of class biotmle, generated by sub-classing SummarizedExperiment, with additional slots containing tmleOut and call, among others, containing TMLE-based estimates of the relationship between a biomarker and exposure or outcome variable and the original call to this function (for user reference), respectively.

## Examples

```
library(dplyr)
library(biotmleData)
data(illuminaData)
library(SummarizedExperiment)
"%ni%" = Negate("%in%")

colData(illuminaData) <- colData(illuminaData) %>%
     data.frame %>%
     dplyr::mutate(age = as.numeric(age > median(age))) %>%
     DataFrame

varInt_index <- which(names(colData(illuminaData)) %in% "benzene")

biomarkerTMLEout <- biomarkertmle(se = illuminaData[1:2, ],
                                  varInt = varInt_index,
                                  parallel = FALSE,
                                  family = "gaussian",
                                  g_lib = c("SL.mean", "SL.glm"),
                                  Q_lib = "SL.mean"
                                 )
```

## biomarkerTMLE_exposure

*TMLE procedure using ATE for Biomarker Identication from Exposure*

### Description

This function performs influence curve-based estimation of the effect of an exposure on biological expression values associated with a given biomarker, controlling for a user-specified set of baseline covariates.

### Usage

```
biomarkerTMLE_exposure(Y, W, A, a, subj_ids = NULL,
  family = "gaussian", g_lib, Q_lib)
```

### Arguments

| | |
|---|---|
| `Y` | A `numeric` vector of expression values for a single biomarker. |
| `W` | A `Matrix` of `numeric` values corresponding to baseline covariates to be marginalized over in the estimation process. |
| `A` | A `numeric` vector of discretized exposure vector (e.g., from a design matrix whose effect on expression values is of interest. |
| `a` | The `numeric` value indicating levels of `A` above against which comparisons are to be made. |
| `subj_ids` | A `numeric` vector of subject IDs to be passed directly to `tmle::tmle` when there are repeated measures; measurements on the same subject should have the exact same numerical identifier. These values will be coerced to numeric if not provided in the appropriate form (e.g., as `character`). The call to `tmle::tmle` will utilized a corrected version of the variance estimate from the efficient influence function. |
| `family` | (character) - specification of error family: "binomial" or "gaussian" |
| `g_lib` | (char vector) - library of learning algorithms to be used in fitting the "g" step of the standard TMLE procedure. |
| `Q_lib` | (char vector) - library of learning algorithms to be used in fitting the "Q" step of the standard TMLE procedure. |

### Value

TMLE-based estimate of the relationship between biomarker expression and changes in an exposure variable, computed iteratively and saved in the `tmleOut` slot in a `biotmle` object.

## bioTMLE-class *Constructor for class bioTMLE*

### Description

Constructor for class bioTMLE

### Value

class `biotmle` object, sub-classed from SummarizedExperiment.

### Examples

```
library(SummarizedExperiment)
library(biotmleData)
data(illuminaData)

example_biotmle_class <- function(se) {

    call <- match.call(expand.dots = TRUE)
    biotmle <- .biotmle(
          SummarizedExperiment(
             assays = assay(se),
             rowData = rowData(se),
             colData = colData(se)
          ),
          call = call,
          tmleOut = as.data.frame(matrix(NA, 10, 10)),
          topTable = as.data.frame(matrix(NA, 10, 10))
    )
    return(biotmle)
}

example_class <- example_biotmle_class(se = illuminaData)
```

## data.frame_OR_EList-class *S4 class union data.frame_OR_EList*

### Description

Virtual class union containing members of both `data.frame` and `limma::Elist`, used internally to handle situations when a returned object has a type that cannot be guessed from the function call.

### Value

fusion of classes `data.frame` and `EList`, used within `.biotmle` by class `bioTMLE` to handle uncertainty in the object passed to slot "tmleOut".

## heatmap_ic *Heatmap for class biotmle*

### Description

Heatmap of the contributions of a select subset of biomarkers to the variable importance measure changes as assessed by influence curve-based estimation, across all subjects.

### Usage

```
heatmap_ic(x, ..., design, FDRcutoff = 0.05, top = 25)
```

### Arguments

| | |
|---|---|
| `x` | object of class `biotmle` as produced by an appropriate call to `biomarkertmle` |
| `...` | additional arguments passed to `superheat::superheat` as necessary |
| `design` | a vector providing the contrast to be displayed in the heatmap. |
| `FDRcutoff` | cutoff to be used in controlling the False Discovery Rate |
| `top` | number of identified biomarkers to plot in the heatmap |

### Value

heatmap (from the superheat package) using hierarchical clustering to plot the changes in the variable importance measure for all subjects across a specified top number of biomarkers.

### Examples

```
library(dplyr)
library(biotmleData)
library(SummarizedExperiment)
data(illuminaData)
data(biomarkertmleOut)

colData(illuminaData) <- colData(illuminaData) %>%
     data.frame %>%
     dplyr::mutate(age = as.numeric(age > median(age))) %>%
     DataFrame

varInt_index <- which(names(colData(illuminaData)) %in% "benzene")
designVar <- as.data.frame(colData(illuminaData))[, varInt_index]
design <- as.numeric(designVar == max(designVar))

limmaTMLEout <- modtest_ic(biotmle = biomarkerTMLEout)

heatmap_ic(x = limmaTMLEout, design = design, FDRcutoff = 0.05, top = 15)
```

## `modtest_ic` *Moderated Statistical Tests for Influence Functions*

### Description

Performs variance shrinkage via the empirical Bayes procedure of LIMMA on the observed data after a transformation moving the data to influence function space, based on the average treatment effect parameter.

### Usage

```
modtest_ic(biotmle, adjust = "BH")
```

### Arguments

| | |
|---|---|
| `biotmle` | `biotmle` object as generated by `biomarkertmle` |
| `adjust` | the multiple testing correction to be applied to p-values that are generated from the moderated tests. The recommended (and default) method is that of Benjamini and Hochberg. See topTable for a list of appropriate methods. |

### Value

`biotmle` object containing output from `limma::lmFit` and `limma::topTable`

### Examples

```
library(biotmleData)
library(SummarizedExperiment)
data(biomarkertmleOut)

limmaTMLEout <- modtest_ic(biotmle = biomarkerTMLEout)
```

## `plot.bioTMLE` *Plot p-values from moderated statistical tests for class biotmle*

### Description

Histogram of raw or FDR-adjusted p-values from the moderated t-test.

### Usage

```
## S3 method for class 'bioTMLE'
plot(x, ..., type = "pvals_adj")
```

### Arguments

| | |
|---|---|
| `x` | object of class `biotmle` as produced by an appropriate call to `biomarkertmle` |
| `...` | additional arguments passed `plot` as necessary |
| `type` | character describing whether to provide a plot of unadjusted or adjusted p-values (adjustment performed via Benjamini-Hochberg) |

### Value

object of class `ggplot` containing a histogram of the raw or Benjamini-Hochberg corrected p-values (depending on user input).

### Examples

```
library(dplyr)
library(biotmleData)
library(SummarizedExperiment)
data(biomarkertmleOut)

limmaTMLEout <- modtest_ic(biotmle = biomarkerTMLEout)

plot(x = limmaTMLEout, type = "pvals_adj")
```

---

## `rnaseq_ic` *Transformation utility for using "voom" with biomarker TMLE procedure*

---

### Description

This function prepares next-generation sequencing data (counts) for use with the biomarker TMLE procedure by invoking the voom transform of `limma`.

### Usage

```
rnaseq_ic(biotmle, weights = TRUE, ...)
```

### Arguments

| | |
|---|---|
| `biotmle` | (bioTMLE) - subclass of `SummarizedExperiment` containing next-generation sequencing (NGS) count data in the "assays" slot. |
| `weights` | (logical) - whether to return quality weights of samples in the output object. |
| `...` | - other arguments to be passed to functions `limma::voom` or `limma::voomWithQualityWeights` as appropriate. |

### Value

`EList` object containing voom-transformed "expression" measures of count data (actually, the mean-variance trend) in the "E" slot, to be passed into the biomarker TMLE procedure.

## volcano_ic *Volcano plot for class biotmle*

### Description

Volcano plot of the log-changes in the target causal paramter against the log raw p-values from the moderated t-test.

### Usage

```
volcano_ic(biotmle, fc_bound = 3, pval_bound = 0.2)
```

### Arguments

| | |
|---|---|
| biotmle | object of class `biotmle` as produced by an appropriate call to `biomarkertmle` |
| fc_bound | (numeric) - indicates the highest magnitude of the fold to be colored along the x-axis of the volcano plot; this limits the observations to be considered differentially expressed to those in a user-specified interval. |
| pval_bound | (numeric) - indicates the largest corrected p-value to be colored along the y-axis of the volcano plot; this limits observations considered as differentially expressed to those in a user-specified interval. |

### Value

object of class `ggplot` containing a standard volcano plot of the log-fold change in the causal target parameter against the raw log p-value computed from the moderated tests in `modtest_ic`.

### Examples

```
library(dplyr)
library(biotmleData)
library(SummarizedExperiment)
data(biomarkertmleOut)

limmaTMLEout <- modtest_ic(biotmle = biomarkerTMLEout)

volcano_ic(biotmle = limmaTMLEout)
```

# Index