Package ‘biotmle’

April 25, 2017

Title Targeted Learning for Biomarker Discovery with Moderated Statistics

Version 1.0.0

Author Nima Hejazi [aut, cre, cph]

Maintainer Nima Hejazi <nhejazi@berkeley.edu>

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 the moderated t-statistic with asymptotically linear statistical parameters estimated via targeted minimum loss-based estimation (TMLE).

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 tmle, limma, foreach, parallel, doParallel, ggplot2, wesanderson, magrittr, stats, dplyr, SummarizedExperiment, superheat, SuperLearner, biotmleData, Matrix

Suggests testthat, rmarkdown, knitr

Remotes nhejazi/biotmleData

VignetteBuilder knitr

RoxygenNote 6.0.1

biocViews GeneExpression, DifferentialExpression, Sequencing, Microarray, RNASeq

NeedsCompilation no

R topics documented:

biomarkerTmle ................................................................. 2
biomarkerTMLE_exposure .................................................. 3
biomarkertmle

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, type = c("exposure", "outcome"), parallel = TRUE, family = "gaussian", 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 "se" argument above).
type (character) - choice of the type of TMLE to perform: "exposure" to identify biomarkers related to an exposure (input as A), or "outcome" to identify biomarkers related to an outcome (input as Y).
parallel (logical, numeric) - whether to use or the number of cores to be used when the TMLE-based estimation procedure is parallelized.
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
S4 object of class biomarkertmle, 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

```r
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,,],
   varInt = varInt_index,
   type = "exposure",
   parallel = 1,
   family = "gaussian",
   g_lib = c("SL.mean"),
   Q_lib = c("SL.mean")
)
```

biomarkerTMLE_exposure

*TMLE procedure for Biomarker Identification 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

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

Arguments

- **Y** (numeric vector) - a vector of expression values for a single biomarker.
- **W** (numeric matrix) - a matrix of baseline covariates to be controlled in the estimation process.
- **A** (numeric vector) - a discretized exposure vector (e.g., from a design matrix whose effect on expression values is of interest.
- **a** (numeric vector) - the levels of A against which comparisons are to be made.
- **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.

---

**biomarkerTMLE_outcome**  
*TMLE procedure for Biomarker Identification from Outcome*

**Description**

This function performs influence curve-based estimation of the effect of expression changes of a biomarker on an outcome while controlling for a set of user-specified baseline covariates.

**Usage**

```
biomarkerTMLE_outcome(Y, W, A, a = 1, family = "binomial", g_lib, Q_lib)
```

**Arguments**

- `Y`  
  (numeric vector) - a vector of binarized outcome values, thought to be impacted by changes in biomarker expression values.

- `W`  
  (numeric matrix) - a matrix of baseline covariates to be controlled for in the estimation procedure.

- `A`  
  (numeric vector) - a discretized vector of expression values from a given biomarker.

- `a`  
  (numeric vector) - the levels of A against which comparisons are to be made.

- `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 changes in biomarker expression and an outcome 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.
Example

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

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)),
    modtestOut = as.data.frame(matrix(NA, 10, 10)),
    topTable = as.data.frame(matrix(NA, 10, 10))
  )
  return(biotmle)
}

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

**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**

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

**Arguments**

- `x`: object of class `biotmle` as produced by an appropriate call to `biomarkerIC`
- `...`: 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

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

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

limmaTMLEout <- modtest_ic(biotmle = biomarkerTMLEout, design = design)

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

---

**modtest_ic**

*Moderated Statistical Tests for Influence Curves*

**Description**

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

**Usage**

```r
modtest_ic(biotmle, design, ...)
```

**Arguments**

- `biotmle`: biotmle object as generated by biomarkertmle
- `design`: a design matrix providing the contrasts to be used in the linear model fitting procedure of `limma::lmFit`
- `...`: additional arguments to be passed to functions from `limma`

**Value**

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

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

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

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

---

**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

```r
## 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

```r
library(dplyr)
library(biotmleData)
library(SummarizedExperiment)
data(illuminaData)
data(biomarkerTMLEout)

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

limmaTMLEout <- modtest_ic(biotmle = biomarkerTMLEout, design = design)

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

---

**volcano_ic**

**Volcano plot for class biotmle**

**Description**

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

**Usage**

```
volcano_ic(biotmle)
```

**Arguments**

- `biotmle`: object of class `biotmle` as produced by an appropriate call to `biomarkertmle`

**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(illuminaData)
data(biomarkertmleOut)
"%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")
designVar <- as.data.frame(colData(illuminaData))[, varInt_index]
design <- as.numeric(designVar == max(designVar))
```
volcano_ic

```r
limmaTMLEout <- modtest_ic(biomle = biomarkerTMLEout, IDs = NULL,
                         design = design)

volcano_ic(biomle = limmaTMLEout)
```
Index

.bioutmle (bioTMLE-class), 4
biomarkertmle, 2
biomarkerTMLE_exposure, 3
biomarkerTMLE_outcome, 4
bioTMLE-class, 4

heatmap_ic, 5
modtest_ic, 6
plot.bioTMLE, 7
volcano_ic, 8