

Package ‘MCPModPack’

January 20, 2025

Type Package

Title Simulation-Based Design and Analysis of Dose-Finding Trials

Version 0.5

Date 2023-05-23

Depends R (>= 3.4.1), mvtnorm

Imports shiny, shinydashboard, devEMF, officer, flextable, Rcpp,
RcppNumerical

Suggests testthat, devtools, DoseFinding, covr

LinkingTo Rcpp, RcppEigen, RcppNumerical

Maintainer Alex Dmitrienko <admitrienko@mediana.us>

Description An efficient implementation of the MCPMod (Multiple Comparisons and Modeling) method to support a simulation-based design and analysis of dose-finding trials with normally distributed, binary and count endpoints (Bretz et al. (2005) <[doi:10.1111/j.1541-0420.2005.00344.x](https://doi.org/10.1111/j.1541-0420.2005.00344.x)>).

License GPL-3

LazyLoad yes

LazyData true

NeedsCompilation yes

Author Alex Dmitrienko [aut, cre],
Alun Bedding [ctb],
John Cook [ctb]

Repository CRAN

URL <https://github.com/medianasoft/MCPModPack>

BugReports <https://github.com/medianasoft/MCPModPack/issues>

Date/Publication 2023-05-30 12:20:04 UTC

Contents

MCPModPack-package	2
AnalysisApp	4
AnalysisReport	5
binary	6
count	7
MCPModAnalysis	7
MCPModSimulation	10
normal	13
SimulationApp	14
SimulationReport	14

Index	17
--------------	-----------

MCPModPack-package	<i>Design and analysis of dose-finding trials</i>
--------------------	---

Description

The MCPModPack package facilitates the design and analysis of dose-finding clinical trials with normally distributed, binary and count endpoints using the MCPMod methodology.

Details

Package:	MCPModPack
Type:	Package
Version:	0.3
Date:	2020-08-01
License:	GPL-3

Key functions included in the package:

- [MCPModAnalysis](#): Analyze data from a dose-finding trial using MCPMod.
- [AnalysisReport](#): Generate a detailed summary of MCPMod analysis results in a Microsoft Word format.
- [AnalysisApp](#): Launch a Shiny-based graphical user interface to analyze data from a dose-finding trial.
- [MCPModSimulation](#): Perform a simulation-based evaluation of dose-finding trial designs using MCPMod.
- [SimulationReport](#): Generate a detailed summary of MCPMod simulation results in a Microsoft Word format.
- [SimulationApp](#): Launch a Shiny-based graphical user interface to perform a simulation-based evaluation of dose-finding trial designs.

The package comes with three example data sets:

- **normal**: Data set based on a dose-finding trial with a normally distributed endpoint.
- **binary**: Data set based on a dose-finding trial with a binary endpoint.
- **count**: Data set based on a dose-finding trial with a count endpoint.

Author(s)

Alex Dmitrienko <admitrienko@mediana.us>

References

- Bornkamp, B., Bezlyak, V., Bretz, F. (2015). Implementing the MCP-Mod procedure for dose-response testing and estimation. *Modern Approaches to Clinical Trials Using SAS*. Menon, S., Zink, R. (editors). SAS Press: Cary, NC.
- Bretz, F., Pinheiro, J.C., Branson, M. (2005). Combining multiple comparisons and modeling techniques in dose response studies. *Biometrics*. 61, 738-748.
- Bretz, F., Tamhane, A.C., Pinheiro, J. (2009). Multiple testing in dose response problems. *Multiple Testing Problems in Pharmaceutical Statistics*. Dmitrienko, A., Tamhane, A.C., Bretz, F. (editors). New York: Chapman and Hall/CRC Press.
- Nandakumar, S., Dmitrienko, A., Lipkovich, I. (2017). Dose-finding methods. *Analysis of Clinical Trials Using SAS: A Practical Guide* (Second Edition). Dmitrienko, A., Koch, G.G. (editors). SAS Press: Cary, NC.
- Pinheiro, J. C., Bornkamp, B., Bretz, F. (2006). Design and analysis of dose finding studies combining multiple comparisons and modeling procedures. *Journal of Biopharmaceutical Statistics*. 16, 639-656.
- Pinheiro J., Bornkamp B., Glimm E., Bretz F. (2013). Model-based dose finding under model uncertainty using general parametric models. *Statistics in Medicine*. 33, 1646-1661.

Examples

```
# MCPMod-based analysis of a dose-finding trial with a normally distributed endpoint

# Select the candidate dose-response models and initial values
# of the non-linear model parameters (linear, quadratic, exponential,
# emax, logistic and sigemax)
models = list(linear = NA,
              quadratic = -0.5,
              exponential = 0.3,
              emax = 0.3,
              logistic = c(0.5, 0.1),
              sigemax = c(0.5, 5))

# One-sided Type I error rate
alpha = 0.025

# Direction of the dose-response relationship
direction = "increasing"

# Model selection criterion
model_selection = "AIC"
```

```
# The treatment effect for identifying the target dose
# (this effect is defined relative to the placebo effect)
Delta = 0.5

# Perform an MCPMod-based analysis of the trial's data
# The data set normal is included in the package
results = MCPModAnalysis(endpoint_type = "Normal",
                          models = models,
                          dose = normal$dose,
                          resp = normal$resp,
                          alpha = alpha,
                          direction = direction,
                          model_selection = model_selection,
                          Delta = Delta)

# Simple summary of the MCPMod analysis results
results

# Detailed summary of the MCPMod analysis results (remove tempfile)
AnalysisReport(results,
               "MCPMod analysis summary (Normally distributed endpoint)",
               tempfile("MCPMod analysis summary (Normally distributed endpoint).docx", fileext=".docx"))
```

AnalysisApp

*Creation of a Shiny-based interface to perform an MCPMod analysis
of a dose-finding trial*

Description

This function creates a Shiny-based graphical user interface to perform MCPMod-based analysis of a dose-finding trial. The application requires data sets with the dose and response information (comma-separated value file). The data set is required to include the dose and resp variables with a single record per patient. The first row of the file should contain the required variable names as follows: "dose","resp" (including the quotation marks and the comma). Subsequent rows should contain comma-separated values corresponding to the dose and response values for each patient (quotation marks are not required for the rows of data). See for example the [normal](#) data set.

Usage

```
AnalysisApp()
```

Author(s)

Alex Dmitrienko <admitrienko@mediana.us>

See Also

[MCPModAnalysis](#), [AnalysisReport](#)

AnalysisReport	<i>Generation of a Word-based summary of MCPMod analysis results</i>
----------------	--

Description

This function creates a detailed summary of MCPMod analysis results in a Microsoft Word format.

Usage

```
AnalysisReport(results, report_title, report_filename)
```

Arguments

results	Object of class 'MCPModAnalysisResults' created by the MCPModAnalysis function.
report_title	Character value defining the report's title.
report_filename	Character value defining the report's filename. The report is saved in the current working directory.

Author(s)

Alex Dmitrienko <admitrienko@mediana.us>

See Also

[MCPModAnalysis](#), [SimulationReport](#)

Examples

```
# MCPMod-based analysis of a dose-finding trial with a
# binary endpoint

# Endpoint type
endpoint_type = "Binary"

# Select the candidate dose-response models and initial values
# of the non-linear model parameters (linear, quadratic, exponential,
# emax, logistic and sigemax)
models = list(linear = NA,
              quadratic = -0.5,
              exponential = 0.3,
              emax = 0.3,
              logistic = c(0.5, 0.1),
              sigemax = c(0.5, 5))

# One-sided Type I error rate
alpha = 0.025
```

```

# Direction of the dose-response relationship
direction = "increasing"

# Model selection criterion
model_selection = "AIC"

# The treatment effect for identifying the target dose
# (this effect is defined relative to the placebo effect)
Delta = 0.3

# Perform an MCPMod-based analysis of the trial's data
# The data set binary is included in the package
results = MCPModAnalysis(endpoint_type = endpoint_type,
                          models = models,
                          dose = binary$dose,
                          resp = binary$resp,
                          alpha = alpha,
                          direction = direction,
                          model_selection = model_selection,
                          Delta = Delta)

# Simple summary of the MCPMod analysis results
results

# Detailed summary of the MCPMod analysis results (remove tempfile)
AnalysisReport(results,
               "MCPMod analysis summary (Binary endpoint)",
               tempfile("MCPMod analysis summary (Binary endpoint).docx", fileext=".docx"))

```

binary

Example data set with a binary endpoint

Description

Example data set based on a trial with a binary primary endpoint.

Usage

```
data(binary)
```

Format

A data set with 100 observations and 2 variables:

dose Dose level (0 g, 0.05 g, 0.2 g, 0.6 g and 1 g).

resp Binary outcome variable (0 or 1). The value of 1 indicates a beneficial outcome.

count	<i>Example data set with a count endpoint</i>
-------	---

Description

Example data set based on a trial with a count primary endpoint.

Usage

```
data(count)
```

Format

A data set with 100 observations and 2 variables:

dose Dose level (0 g, 0.05 g, 0.2 g, 0.6 g and 1 g).

resp Count-type outcome variable (number of events for each patient). A higher value of this variable indicates a beneficial outcome.

MCPModAnalysis	<i>MCPMod-based analysis of dose-finding clinical trials with normally distributed, binary and count endpoints</i>
----------------	--

Description

This function implements the MCPMod-based analysis of dose-finding clinical trials with normally distributed, binary and count endpoints, including derivation of the optimal contrasts for the candidate dose-response models, evaluation of dose-response tests based on the optimal contrasts, selection of the significant dose-response models and estimation of the target dose. For more information, see the technical manual in the package's doc folder.

Usage

```
MCPModAnalysis(endpoint_type, models, dose, resp, alpha, direction,
               model_selection, Delta, theta)
```

Arguments

endpoint_type Character value defining the primary endpoint's type. Possible values:

- "Normal": Normally distributed primary endpoint.
- "Binary": Binary primary endpoint.
- "Count": Count-type primary endpoint.

models	List of candidate dose-response models with initial values of the non-linear model parameters. The package supports the following dose-response models: linear, quadratic, exponential, Emax, logistic and sigEmax. No initial value is required for the linear model, a single initial value is required for the quadratic, exponential and Emax models, and two initial values are required for the logistic and sigEmax models.
dose, resp	Numeric vectors of equal length specifying the dose and response values.
alpha	Numeric value defining the one-sided significance level (default value is 0.025).
direction	Character value defining the direction of the dose-response relationship. Possible values: <ul style="list-style-type: none"> • "Increasing": A larger value of the treatment difference corresponds to a beneficial treatment effect. • "Decreasing": A smaller value of the treatment difference indicates a beneficial treatment effect.
model_selection	Character value defining the criterion for selecting the best dose-response model. Possible values: <ul style="list-style-type: none"> • "AIC": Akaike information criterion (AIC). • "maxT": Most significant test statistic. • "aveAIC": Weighted AIC-based criterion.
Delta	Numeric value specifying the treatment effect for identifying the target dose. The treatment effect is defined relative to the placebo effect. A positive value is required if direction = "Increasing" and a negative value is required otherwise.
theta	Numeric vector defining the overdispersion parameter in each trial arm (required only with count-type primary endpoints).

Value

The function returns an object of class 'MCPModAnalysisResults'. This object is a list with the following components:

input_parameters	a list containing the user-specified parameters, e.g, endpoint type, model selection criteria, etc.
selected_models	a logical vector defining the candidate dose-response models.
descriptive_statistics	a list containing the descriptive statistics computed from the trial's data set.
contrast_results	a list containing the contrast evaluation results for the candidate dose-response models, e.g., the model-specific optimal dose-response contrasts and contrast correlation matrix.
mcp_results	a list containing the multiplicity adjustment results for the candidate dose-response models, e.g., the model-specific test statistics and adjusted p-values.


```

Delta = Delta)

# Simple summary of the MCPMod analysis results
results

# Detailed summary of the MCPMod analysis results (remove tempfile)
AnalysisReport(results,
  "MCPMod analysis summary (Binary endpoint)",
  tempfile("MCPMod analysis summary (Binary endpoint).docx", fileext=".docx"))

```

MCPModSimulation

MCPMod-based simulation of dose-finding trial designs

Description

This function implements the simulation-based analysis of dose-finding clinical trials with normally distributed, binary and count endpoints using the MCPMod methodology. For more information, see the technical manual in the package's doc folder.

Usage

```

MCPModSimulation(endpoint_type, models, alpha, direction,
  model_selection, Delta, theta, sim_models, sim_parameters)

```

Arguments

endpoint_type	Character value defining the primary endpoint's type. Possible values: <ul style="list-style-type: none"> • "Normal": Normally distributed primary endpoint. • "Binary": Binary primary endpoint. • "Count": Count-type primary endpoint.
models	List of candidate dose-response models with initial values of the non-linear model parameters. The package supports the following dose-response models: linear, quadratic, exponential, emax, logistic and sigemax. No initial value is required for the linear model, a single initial value is required for the quadratic, exponential and Emax models, and two initial values are required for the logistic and sigEmax models.
alpha	Numeric value defining the one-sided significance level (default value is 0.025).
direction	Character value defining the direction of the dose-response relationship. Possible values: <ul style="list-style-type: none"> • "Increasing": A larger value of the treatment difference corresponds to a beneficial treatment effect. • "Decreasing": A smaller value of the treatment difference indicates a beneficial treatment effect.
model_selection	Character value defining the criterion for selecting the best dose-response model. Possible values:

- "AIC": Akaike information criterion (AIC).
- "maxT": Most significant test statistic.
- "aveAIC": Weighted AIC-based criterion.

Delta	Numeric value specifying the treatment effect for identifying the target dose. The treatment effect is defined relative to the placebo effect. A positive value is required if <code>direction = "Increasing"</code> and a negative value is required otherwise.
theta	Numeric vector defining the overdispersion parameter in each trial arm (required only with count-type primary endpoints).
sim_models	List defining the assumed dose-response model and initial values of the non-linear parameters in the simulated trial. The package supports the following dose-response models: linear, quadratic, exponential, Emax, logistic and sigEmax. No initial value is required for the linear model, a single initial value is required for the quadratic, exponential and Emax models, and two initial values are required for the logistic and sigEmax models. The following components are required: <ul style="list-style-type: none"> • "placebo_effect": Numeric value defining the placebo effect in the assumed dose-response model. • "max_effect": Numeric vector defining the effects at the maximum dose in the assumed dose-response model. These effects are defined relative to the placebo effect. Positive values are required if <code>direction = "Increasing"</code> and negative values are required otherwise. • "sd": Numeric vector defining the standard deviations of the response variable in each trial arm (required for normally distributed endpoints).
sim_parameters	List defining the design and simulation parameters in the simulated trial. The following components are required: <ul style="list-style-type: none"> • "n": Integer vector defining the number of patients in each trial arm. • "doses": Numeric vector defining the dose levels in each trial arm. • "dropout_rate": Numeric value defining the dropout rate in the simulated trial (between 0 and 1). • "nsims": Integer value defining the number of simulation runs. • "go_threshold": Numeric value specifying the threshold for computing go probabilities, i.e., probabilities that the maximum effect for the best dose-response model corresponding to a significant contrast exceeds a pre-defined value. The threshold is defined relative to the placebo effect. A positive value is required if <code>direction = "Increasing"</code> and a negative value is required otherwise.

Value

The function returns an object of class 'MCPModSimulationResults'. This object is a list with the following components:

`input_parameters`

a list containing the user-specified parameters, e.g, endpoint type, model selection criteria, etc.

`selected_models` a logical vector defining the candidate dose-response models.

`sim_results` a list containing the simulation results based on the assumed dose-response model, e.g., power, target dose estimates, etc.

A detailed summary of the simulation results can be generated using the `SimulationReport` function.

Author(s)

Alex Dmitrienko <admitrienko@mediana.us>

See Also

[MCPModAnalysis](#)

Examples

```
# Simulation-based evaluation of dose-finding trials with a count endpoint

# Endpoint type
endpoint_type = "Count"

# Select the candidate dose-response models and initial values
# of the non-linear model parameters (linear, quadratic, exponential,
# emax, logistic and sigemax)
models = list(linear = NA,
              quadratic = -0.5,
              exponential = 0.3,
              emax = 0.3,
              logistic = c(0.5, 0.1),
              sigemax = c(0.5, 5))

# One-sided Type I error rate
alpha = 0.025

# Direction of the dose-response relationship
direction = "increasing"

# Model selection criterion
model_selection = "AIC"

# The treatment effect for identifying the target dose
# (this effect is defined relative to the placebo effect)
Delta = 2

# Vector of overdispersion parameters
theta = c(2, 2, 2, 2, 2)

# Select the assumed dose-response model and values of the non-linear model parameters
sim_models = list(emax = 1,
                  placebo_effect = 3,
                  max_effect = seq(from = 0, to = 3, by = 1))
```

```

# Simulation parameters
# (go threshold is defined relative to the placebo effect)
sim_parameters = list(n = c(50, 50, 50, 50, 50),
                      doses = c(0, 0.05, 0.2, 0.6, 1),
                      dropout_rate = 0.05,
                      nsims = 1000,
                      go_threshold = 2)

# Perform an MCPMod-based simulation
results = MCPModSimulation(endpoint_type = endpoint_type,
                           models = models,
                           alpha = alpha,
                           direction = direction,
                           model_selection = model_selection,
                           Delta = Delta,
                           theta = theta,
                           sim_models = sim_models,
                           sim_parameters = sim_parameters)

# Simple summary of the MCPMod simulation results
results

# Detailed summary of the MCPMod simulation results (remove tempfile)
SimulationReport(results,
                 "MCPMod simulation summary (Count endpoint)",
                 tempfile("MCPMod simulation summary (Count endpoint).docx", fileext=".docx"))

```

normal

Example data set with a continuous endpoint

Description

Example data set based on a trial with a continuous (normally distributed) primary endpoint.

Usage

```
data(normal)
```

Format

A data set with 100 observations and 2 variables:

dose Dose level (0 g, 0.05 g, 0.2 g, 0.6 g and 1 g).

resp Continuous outcome variable. A higher value of this variable indicates a beneficial outcome.

SimulationApp	<i>Creation of a Shiny-based interface to perform an MCPMod-based simulation of dose-finding trial designs</i>
---------------	--

Description

This function creates a Shiny-based graphical user interface to perform a simulation-based evaluation of dose-finding trial designs using MCPMod.

Usage

```
SimulationApp()
```

Author(s)

Alex Dmitrienko <admitrienko@mediana.us>

See Also

[MCPModSimulation](#), [SimulationReport](#)

SimulationReport	<i>Generation of a Word-based summary of MCPMod simulation results</i>
------------------	--

Description

This function creates a detailed summary of MCPMod simulation results in a Microsoft Word format.

Usage

```
SimulationReport(results, report_title, report_filename)
```

Arguments

results	Object of class ‘MCPModSimulationResults’ created by the MCPModSimulation function.
report_title	Character value defining the report’s title.
report_filename	Character value defining the report’s filename. The report is saved in the current working directory.

Author(s)

Alex Dmitrienko <admitrienko@mediana.us>

See Also

[MCPModSimulation](#), [AnalysisReport](#)

Examples

```
# Simulation-based evaluation of dose-finding trials with a binary endpoint

# Endpoint type
endpoint_type = "Binary"

# Select the candidate dose-response models and initial values
# of the non-linear model parameters (linear, quadratic, exponential,
# emax, logistic and sigemax)
models = list(linear = NA,
              quadratic = -0.5,
              exponential = 0.3,
              emax = 0.3,
              logistic = c(0.5, 0.1),
              sigemax = c(0.5, 5))

# One-sided Type I error rate
alpha = 0.025

# Direction of the dose-response relationship
direction = "increasing"

# Model selection criterion
model_selection = "AIC"

# The treatment effect for identifying the target dose
# (this effect is defined relative to the placebo effect)
Delta = 0.3

# Select the assumed dose-response model and values of the non-linear model parameters
sim_models = list(linear = NA,
                  placebo_effect = 0.2,
                  max_effect = seq(from = 0, to = 0.5, by = 0.1))

# Simulation parameters
sim_parameters = list(n = rep(40, 5),
                     doses = c(0, 0.05, 0.2, 0.6, 1),
                     dropout_rate = 0.05,
                     nsims = 1000,
                     go_threshold = 0.3)

# Perform an MCPMod-based simulation
results = MCPModSimulation(endpoint_type = endpoint_type,
                           models = models,
                           alpha = alpha,
                           direction = direction,
                           model_selection = model_selection,
                           Delta = Delta,
```

```
sim_models = sim_models,  
sim_parameters = sim_parameters)  
  
# Simple summary of the MCPMod simulation results  
results  
  
# Detailed summary of the MCPMod simulation results (remove tempfile)  
SimulationReport(results,  
  "MCPMod simulation summary (Binary endpoint)",  
  tempfile("MCPMod simulation summary (Binary endpoint).docx", fileext=".docx"))
```


Index

* datasets

- binary, [6](#)
- count, [7](#)
- normal, [13](#)

* package

- MCPModPack-package, [2](#)

AnalysisApp, [2](#), [4](#)

AnalysisReport, [2](#), [4](#), [5](#), [15](#)

binary, [3](#), [6](#)

ComputeDRFunctionParameters
(MCPModSimulation), [10](#)

count, [3](#), [7](#)

EvaluateDRFunction (MCPModSimulation),
[10](#)

GenerateAnalysisReport
(AnalysisReport), [5](#)

GenerateSimulationReport
(SimulationReport), [14](#)

MCPModAnalysis, [2](#), [4](#), [5](#), [7](#), [12](#)

MCPModPack (MCPModPack-package), [2](#)

MCPModPack-package, [2](#)

MCPModSimulation, [2](#), [9](#), [10](#), [14](#), [15](#)

normal, [3](#), [4](#), [13](#)

SimulationApp, [2](#), [14](#)

SimulationReport, [2](#), [5](#), [14](#), [14](#)

TestDoseResponseFunction
(MCPModPack-package), [2](#)

TestFindTargetDose
(MCPModPack-package), [2](#)