Package 'TesiproV'

March 25, 2022

Type Package

Title Calculation of Reliability and Failure Probability in Civil Engineering

Version 0.9.2

Maintainer Konstantin Nille-Hauf <konstantin.nillehauf@googlemail.com>

Description Calculate the failure probability of civil engineering problems with Level I up to Level III Methods. Have fun and enjoy. References: Spaethe (1991, ISBN:3-211-82348-4) ``Die Sicherheit tragender Baukonstruktionen'', AU,BECK (2001) ``Estimation of small failure probabilities in high dimensions by subset simulation.'' <doi:10.1016/S0266-8920(01)00019-4>, Breitung (1989) ``Asymptotic approximations for probability integrals.'' <doi:10.1016/0266-8920(89)90024-6>.

URL https://www.hochschule-biberach.de/transfer/forschung/institut-fuer-konstruktiven-ingenieurbau

Encoding UTF-8

License MIT + file LICENSE

RoxygenNote 7.1.2

Suggests testthat, knitr, rmarkdown, evd

VignetteBuilder knitr

Imports pracma, nloptr, methods, edfun, ggplot2, gridExtra

NeedsCompilation no

Author Konstantin Nille-Hauf [aut, cre],
Tania Feiri [aut],
Marcus Ricker [aut]

Repository CRAN

Date/Publication 2022-03-25 15:50:02 UTC

R topics documented:

`debug.print` *internal Helper function to debug more easy*

Description

internal Helper function to debug more easy

Usage

```
debug.print(infoLevel, flag = "", values, msg = "", type = "INFO")
```

Arguments

`infoLevel`	If 0, no Output (just Errors), if 1 little output, if 2 bigger output
`flag`	Parse additonal info
`values`	If you check variables then post this into values
`msg`	here add some extra msg
`type`	Type can be "INFO" or "ERROR"

Author(s)

(C) 2021 - M. Ricker, K. Nille-Hauf, T. Feiri - Hochschule Biberach, Institut fuer Konstruktiven Ingenieurbau

`dlt` *Density Function for logarithmic student T distritbution*

Description

Density Function for logarithmic student T distritbution

Usage

```
dlt(x, m, s, n, nue)
```

Arguments

`x`	quantiles
`m`	mean (1. parameter)
`s`	standard deviation (2. parameter)
`n`	3. paramter
`nue`	degrees of freedom

Value

density

Author(s)

(C) 2021 - K. Nille-Hauf, T. Feiri, M. Ricker - Hochschule Biberach, Institut fuer Konstruktiven Ingenieurbau

Examples

```
dlt(0.5,3,6,2,5)
```

`FORM` *First Order Reliablity Method*

Description

Method to calculate failure probability for structural engineering using approximation of limit state function with linear part.

Usage

```
FORM(
  lsf,
  lDistr,
  n_optim = 10,
  loctol = 0.01,
  optim_type = "rackfies",
  debug.level = 0
)
```

Arguments

`lsf`	objective function with limit state function in form of `function(R,E) {R-E}`. Supplied by a SYS_ object, do not supply yourself.
`lDistr`	list ob distribiutions regarding the distribution object of TesiproV. Supplied by a SYS_ object, do not supply yourself.
`n_optim`	number of opimaziationcycles (not recommended/need for lagrangian algorithms).
`loctol`	Tolerance of the local solver algorithm
`optim_type`	Optimaziationtypes. Available: Augmented Lagrangian Algorithm (use: "auglag"), Rackwitz-Fissler Algorithm (use: "rackfies").
`debug.level`	If 0 no additional info if 2 high output during calculation

Value

The results will be provided within a list with the following objects.

beta HasoferLind Beta Index

pf probablity of failure

u_points solution points

dy gradients

Author(s)

(C) 2021 - K. Nille-Hauf, T. Feiri, M. Ricker - Hochschule Biberach, Institut fuer Konstruktiven Ingenieurbau

References

HASOFER AM, LIND NC. An exact and invarient first order reliability format. J Eng Mech Div Proc ASCE 1974;100(1):111–21.

Rackwitz-Fiessler: RACKWITZ R., FIESSLER B. Structural reliability under combined random load sequences. Comput Struct 1978;9(5), S. 489–94.

Optimised algorithm: YPMA, J., JOHNSON, S.G., BORCHERS, H.W., EDDELBUETTEL, D., RIPLEY, B., HORNIK K., CHIQUET, J., ADLER, A., nloptr: R Interface to NLopt. R package. 2020. Version 1.2.2.

Spaethe, G.: Die Sicherheit tragender Baukonstruktionen, 2. Aufl. Wien: Springer, 1991. – ISBN 3-211-82348-4

MC_CRUDE *Crude MonteCarlo Simulation*

Description

Method to calculate failure probability for structural engineering

Usage

```
MC_CRUDE(
  lsf,
  lDistr,
  cov_user = 0.05,
  n_batch = 400,
  n_max = 1e+07,
  use_threads = 6,
  dataRecord = TRUE,
  debug.level = 0
)
```

Arguments

`lsf`	objective function with limit state function in form of function(x) x[1]+x[2]...
`lDistr`	list ob distribiutions regarding the distribution object of TesiproV
`cov_user`	The Coefficent of variation the simulation should reach
`n_batch`	Size per batch for parallel computing
`n_max`	maximum of iteration the MC should do - its like a stop criterion
`use_threads`	Number of threads for parallel computing, use_threds=1 for single core. Doesnt work on windows!
`dataRecord`	If True all single steps are recorded and available in the results file after on
`debug.level`	If 0 no additional info, if 2 high output during calculation

Value

The results will be provided within a list with the following objects. Acess them with "$"-accessor

pf probablity of failure

pf_FORM probablity of failure of the FORM Algorithm

var variation

cov_mc coefficent of the monteCarlo

n_mc number of iterations done

Author(s)

(C) 2021 - M. Ricker, K. Nille-Hauf, T. Feiri - Hochschule Biberach, Institut fuer Konstruktiven Ingenieurbau

References

Spaethe, G.: Die Sicherheit tragender Baukonstruktionen, 2. Aufl. Wien: Springer, 1991. – ISBN 3-211-82348-4

MC_IS *MonteCarlo Simulation with importance sampling*

Description

Method to calculate failure probability for structural engineering using a simulation method with importance sampling (a method to reduce the amount of needed samples)

Usage

```
MC_IS(
  lsf,
  lDistr,
  cov_user = 0.05,
  n_batch = 16,
  n_max = 1e+06,
  use_threads = 6,
  sys_type = "parallel",
  dataRecord = TRUE,
  beta_l = 100,
  densityType = "norm",
  dps = NULL,
  debug.level = 0
)
```

Arguments

`lsf`	objective function with limit state function in form of function(x) x[1]+x[2]...
`lDistr`	Distributions in input space
`cov_user`	The Coefficent of variation the simulation should reach
`n_batch`	Size per batch for parallel computing
`n_max`	maximum of iteration the MC should do - its like a stop criterion
`use_threads`	determine how many threads to split the work (1=singlecore, 2^n = multicore)
`sys_type`	Determine if parallel or serial system (in case MCIS calculates a system)
`dataRecord`	If True all single steps are recorded and available in the results file afteron
`beta_l`	In Systemcalculation: LSF´s with beta higher than beta_l wont be considered
`densityType`	determines what distributiontype should be taken for the h() density
`dps`	Vector of design points that sould be taken instead of the result of a FORM analysis
`debug.level`	If 0 no additional info if 2 high output during calculation

Value

The results will be provided within a list with the following objects. Acess them with "$"-accessor

pf probablity of failure

pf_FORM probablity of failure of the FORM Algorithm

var variation

cov_mc coefficent of the monteCarlo

n_mc number of iterations done

Author(s)

(C) 2021 - K. Nille-Hauf, T. Feiri, M. Ricker - Hochschule Biberach, Institut fuer Konstruktiven Ingenieurbau

References

DITLEVSEN O, MADSEN H. Structural reliability methods, vol. 178. New York: Wiley; 1996.

Spaethe, G.: Die Sicherheit tragender Baukonstruktionen, 2. Aufl. Wien: Springer, 1991. – ISBN 3-211-82348-4

MC_SubSam *MonteCarlo with Subset-Sampling*

Description

MonteCarlo with Subset-Sampling

Usage

```
MC_SubSam(
  lsf,
  lDistr,
  Nsubset = 1e+05,
  p0 = 0.1,
  MaxSubsets = 10,
  Alpha = 0.05,
  variance = "uniform",
  debug.level = 0
)
```

Arguments

`lsf`	limit-state function
`lDistr`	list of basevariables in input space
`Nsubset`	number of samples in each simulation level
`p0`	level probability or conditional probability
`MaxSubsets`	maximum number of simulation levels that are used to terminate the simulation procedure to avoid infinite loop when the target domain cannot be reached
`Alpha`	confidence level
`variance`	gaussian, uniform
`debug.level`	If 0 no additional info if 2 high output during calculation

Value

The results are provided within a list() of the following elements:

beta

pf

betaCI and pfCI are the corresponding confidence intervals

CoV COV of the result

NumOfSubsets Amount of Markov-Chains

NumOfEvalLSF_nom Markov-Chains times Iterations

NumOfEvalLSF_eff Internal counter that shows the real evaluations of the lsf

runtime Duration since start to finish of the function

Author(s)

(C) 2021 - K. Nille-Hauf, T. Feiri, M. Ricker - Hochschule Biberach, Institut fuer Konstruktiven Ingenieurbau

References

AU, S. K. & BECK, J. L. Estimation of small failure probabilities in high dimensions by subset simulation. Probabilistic Engineering Mechanics, 2001, 16.4: 263-277.

MVFOSM *MVFOSM*

Description

MVFOSM

Usage

```
MVFOSM(lsf, lDistr, h = 1e-04, isExpression = FALSE, debug.level)
```

Arguments

`lsf`	LSF Definition, can be Expression or Function. Defined by the FLAG isExpression (see below)
`lDistr`	List of Distributions
`h`	If isExpression is False, than Finite Difference Method is used for partial deviation. h is the Windowsize
`isExpression`	Boolean, If TRUE lsf has to be typeof expression, otherwise lsf has to be type of function()
`debug.level`	If 0 no additional info if 2 high output during calculation

Value

beta, pf, design.point in x space, alphas, runtime

Author(s)

(C) 2021 - K. Nille-Hauf, T. Feiri, M. Ricker - Hochschule Biberach, Institut fuer Konstruktiven Ingenieurbau#'

References

FREUDENTHAL, A.M. Safety and the probability of structural failure. Am Soc Civil Eng Trans 1956; 121(2843):1337–97.

PARAM_BASEVAR-class *Object for parametric variable*

Description

Object to create parametric basic variables

Fields

- `ParamValues` A vector of values of the parametric studie (e.g. c(1,3,5,7) or seq(1,10,2))
- `ParamType` A field to determine what should be parametric. Possible is: "Mean", "Sd", "DistributionType"

Author(s)

(C) 2021 - K. Nille-Hauf, T. Feiri, M. Ricker - Hochschule Biberach, Institut fuer Konstruktiven Ingenieurbau

PARAM_DETVAR-class *Object for parametric deterministic variable*

Description

Object to create parametric deterministic variables

Fields

- `ParamValues` A vector of values. The first element goes with the first run, second element with second run and so on.

Author(s)

(C) 2021 - K. Nille-Hauf, T. Feiri, M. Ricker - Hochschule Biberach, Institut fuer Konstruktiven Ingenieurbau

PARAM_LSF-class *System Limit State Functions*

Description

Interface for LSF through PROB_LSF. No changes.

Author(s)

(C) 2021 - K. Nille-Hauf, T. Feiri, M. Ricker - Hochschule Biberach, Institut fuer Konstruktiven Ingenieurbau

plt — *Probablity Function for logarithmic student T distritbution*

Description

Probablity Function for logarithmic student T distritbution

Usage

```
plt(q, m, s, n, nue)
```

Arguments

q	quantiles
m	mean (1. parameter)
s	standard deviation (2. parameter)
n	3. paramter
nue	degrees of freedom

Value

density

Author(s)

(C) 2021 - M. Ricker, K. Nille-Hauf, T. Feiri - Hochschule Biberach, Institut fuer Konstruktiven Ingenieurbau

PROB_BASEVAR-class — *Object to store the distribution model for base vars*

Description

Object to store the distribution model for base vars...

Fields

`Id` Place in vector of objective functional expression function(x)x[id]

`Name` name like f_ck, used in the limit state function as input name

`Description` Used for better understanding of vars

`DistributionType` Distributiontypes like "norm", "lnorm", "weibull", "t", "gamma", etc...

`Package` The name of the package the Distribution should be taken from (e.g. "evd")

`Mean` The Mean Value of this Basisvariable

`Sd` The SD Value of this Basisvariable

`Cov` The Cov fitting to Mean and Sd.

`x0` Shiftingparameter

`DistributionParameters` Inputparameters of the distribution, may be calculated internally

Methods

`prepare()` Runs the transformations (from mean, sd -> parameters or the other way round) and checks COV, MEAN and SD fitting together. If distribution is not available an error ll be thrown.

Author(s)

(C) 2021 - K. Nille-Hauf, T. Feiri, M. Ricker - Hochschule Biberach, Institut fuer Konstruktiven Ingenieurbau

Examples

```
var1 <- PROB_BASEVAR(Name="var1", Description="yield strength",
DistributionType="norm", Mean=500, Sd=60)
var1$prepare()

var2 <- PROB_BASEVAR(Name="var2", Description="Load",
DistributionType="gumbel",Package="evd",Mean=40, Sd=3)
var2$prepare()
```

`PROB_DETVAR-class` *Object to store a deterministic model for base vars*

Description

Object to store a deterministic model for base vars

Fields

`Id` Place in vector of objective functional expression function(x)x[id]

`Name` readable name like f_ck, used for transform expression to objective function

`Description` - Used for better understanding of vars

`Value` - The deterministic value that sould be used (as mean for the normal distribution with infinite small sd)

Author(s)

(C) 2021 - K. Nille-Hauf, T. Feiri, M. Ricker - Hochschule Biberach, Institut fuer Konstruktiven Ingenieurbau

Examples

```
form_rf<-PROB_MACHINE(name="FORM RF",fCall="FORM",options=list("n_optim"=20,
"loctol"=0.001, "optim_type"="rackfies"))
sorm <- PROB_MACHINE(name="SORM",fCall="SORM")
mcis<-PROB_MACHINE(name="MC IS",fCall="MC_IS",options=list("cov_user" = 0.05, "n_max"=300000))
mcsus<-PROB_MACHINE(name="MC SuS",fCall="MC_SubSam")
```

`PROB_MACHINE-class` *Object to store prob machines*

Description

Object to store prob machines

Fields

- `name` individual name
- `fCall` Function Call of the method. Possible is: "MVFOSM","FORM", "SORM", "MC_Crude", "MC_IS", "MC_SubSam"
- `options` additional options for the method provided as a list. For form e.g. options=list("optim_type"="rackfies"). To get insight of all available settings of each method open the help with ?FORM, ?SORM, ?MC_IS etc.

Author(s)

(C) 2021 - M. Ricker, K. Nille-Hauf, T. Feiri - Hochschule Biberach, Institut fuer Konstruktiven Ingenieurbau

---

`qlt` *Quantil Function for logarithmic student T distritbution*

Description

Quantil Function for logarithmic student T distritbution

Usage

```
qlt(p, m, s, n, nue)
```

Arguments

`p`	probablity
`m`	mean (1. parameter)
`s`	standard deviation (2. parameter)
`n`	3. paramter
`nue`	degrees of freedom

Value

quantile

Author(s)

(C) 2021 - M. Ricker, K. Nille-Hauf, T. Feiri - Hochschule Biberach, Institut fuer Konstruktiven Ingenieurbau

---

`rlt` *Random Realisation-Function for logarithmic student T distritbution*

Description

Random Realisation-Function for logarithmic student T distritbution

Usage

```
rlt(n_vals, m, s, n, nue)
```

Arguments

`n_vals`	number of realisations
`m`	mean (1. parameter)
`s`	standard deviation (2. parameter)
`n`	3. paramter
`nue`	degrees of freedom

Value

random number

Author(s)

(C) 2021 - M. Ricker, K. Nille-Hauf, T. Feiri - Hochschule Biberach, Institut fuer Konstruktiven Ingenieurbau

SORM *Reliability Analysis at Biberach University of applied sciences*

Description

S. Marelli, and B. Sudret, UQLab: A framework for uncertainty quantification in Matlab, Proc. 2nd Int. Conf. on Vulnerability, Risk Analysis and Management (ICVRAM2014), Liverpool (United Kingdom), 2014, 2554-2563. S. Lacaze and S. Missoum, CODES: A Toolbox For Computational Design, Version 1.0, 2015, URL: www.codes.arizona.edu/toolbox. X. Z. Wu, Implementing statistical fitting and reliability analysis for geotechnical engineering problems in R. Georisk: Assessment and Management of Risk for Engineered Systems and Geohazards, 2017, 11.2: 173-188.

Usage

```
SORM(lsf, lDistr, debug.level = 0)
```

Arguments

`lsf`	objective function with limit state function in form of function(x) x[1]+x[2]...
`lDistr`	list ob distribiutions regarding the distribution object of TesiproV
`debug.level`	If 0 no additional info if 2 high output during calculation

Value

The results will be provided within a list with the following objects. Acess them with "$"-accessor

beta ... HasoferLind Beta Index

pf ... probablity of failure

u_points ... solution points

dy ... gradients

Author(s)

(C) 2021 - T. Feiri, K. Nille-Hauf, M. Ricker - Hochschule Biberach, Institut fuer Konstruktiven Ingenieurbau

References

Breitung, K. (1989). Asymptotic approximations for probability integrals. Probabilistic Engineering Mechanics 4(4), 187–190. 9, 10

Cai, G. Q. and I. Elishakoff (1994). Refined second-order reliability analysis. Structural Safety 14(4), 267–276. 9, 10

Hohenbichler, M., S. Gollwitzer, W. Kruse, and R. Rackwitz (1987). New light on first- and second order reliability methods. Structural Safety 4, 267–284. 10

Tvedt, L. (1990). Distribution of quadratic forms in normal space – Applications to structural reliability. Journal of Engineering Mechanics 116(6), 1183–1197. 10

SYS_LSF-class *System Limit State Functions*

Description

Object that represents a limit state function

Fields

`expr` prepared for expression like SYS_LSF$expr <- expression(f_ck - d_nom)...

`func` prepared for objective functions like SYS_LSF$func <- function(x)return(x[1] + x[2])

`vars` needs list of PROB_BASEVAR-Object

`name` Can be added for better recognition. Otherwise the problem will be called "Unkown Problem"

Methods

`ExpressionToFunction()` Transforms a valid expression into a objective function. Need the set of Variables with correct spelled names and IDs

`check()` Checks all variables. You dont need to execute this, since the system object will do anyway.

Author(s)

(C) 2021 - K. Nille-Hauf, T. Feiri, M. Ricker - Hochschule Biberach, Institut fuer Konstruktiven Ingenieurbau

Examples

```
list_of_vars <- list(PROB_BASEVAR(),PROB_BASEVAR())
lsf1 <- SYS_LSF(name="my first lsf", vars=list_of_vars)
lsf1$func <- function(var1,var2){var1-var2}
```

SYS_PARAM-class *Object for parametric Studies*

Description

Object to create probabilistic problems in parametric studies context. There are no changes how to use compared with SYS_PROB

Fields

`beta_params` Outputfield: See the beta values of the studie

`res_params` Outputfield: See the the full result output of each run

Methods

`printResults(path = "")` TesiproV can create a report file with all the necessary data for you. If you provide a path (or filename, without ending) it will store the data there, otherwise it will report to the console. Set the path via setwd() or check it via getwd().

`runMachines()` Starts solving all given problems (sys_input) with all given algorithms (probMachines). After that one can access via $res...1

Author(s)

(C) 2021 - M. Ricker, K. Nille-Hauf, T. Feiri - Hochschule Biberach, Institut fuer Konstruktiven Ingenieurbau

---

`SYS_PROB-class` *System Probabliation Solution Object*

Description

Object to create probabilistic problems. Including Equation, List of Basisvariable, and Solutionmachines

Fields

`sys_input` List of SYS_LSFs

`sys_type` determining serial or parallel system, not implemented yet

`probMachines` list of PROB_MACHINES

`res_single` grab results after .runMachines()

Methods

`calculateSystemProbability(calcType = "simpleBounds", params = list())` Calculates the system probablity if more than one lsf is given and a system_type (serial or parallel) is set. If calcType is empty (or simpleBounds), only simpleBounds are applied to further calculation of single soultions. If calcType is MCIS, than a Monte Carlo Importance Sampling Method is used (only for parallel systems available). If calcType is MCC, than a Crude Monte Carlo Simulation is used. If calcType is MCSUS, than the Subset Sampling Algorithm ll be used. You can pass arguments to methods via the params field, while the argument has to be a named list (for example check the vignette).

`plotGraph(plotType = "sim.performance")` not finally implemented. Do not use.

`printResults(path = "")` TesiproV can create a report file with all the necessary data for you. If you provide a path (or filename, without ending) it will store the data there, otherwise it will report to the console. Set the path via setwd() or check it via getwd().

`runMachines()` Starts solving all given problems (sys_input) with all given algorithms (probMachines). After that one can access via $res...1

`saveProject(level, filename = "tesiprov_project")` You can save your calculation project with saveProject(). There are four different levels of detail to save 1st Level: Only the beta values 2nd Level: The result Objects of single or systemcalculation 3th Level: All The Probablity System Object, including limit state functions, machines and solutions 4th Level: An image of your entire workspace

Author(s)

(C) 2021 - K. Nille-Hauf, T. Feiri, M. Ricker - Hochschule Biberach, Institut fuer Konstruktiven Ingenieurbau

Examples

```
ps <- SYS_PROB(
sys_input=list(SYS_LSF(),SYS_LSF()),
probMachines=list(PROB_MACHINE()),
sys_type="serial")
## Not run:
ps$runMachines()
ps$beta_sys
ps$res_sys
ps$printResults("example_1")
ps$saveProject(4,"example_1")

## End(Not run)
```

`TesiproV` *TesiproV: A package for the calculation of reliability and failure probability in civil engineering*

Description

The Package provides three main types of objects:

1. Objects for modeling base variables
2. Objects for modeling limit state functions and systems of them
3. Objects for modeling solving algorithms

Details

By creating and combining those objects, one is able to model quite complex problems in terms of structural reliablity calculation. For normally distributed variables there might be an workflow to calculate correlated problems (but no systems then). There is also implemented a new distribution (logStudentT, often used for conrete compression strength) to show how one can implement your very own or maybe combined multi modal distribution and use it with TesiproV.

Objects for base variables

PROB_BASEVAR, PROB_DETVAR, PARAM_BASEVAR, PARAM_DETVAR

Limit state functions

SYS_LSF, PROB_SYS, PARAM_SYS

Solving algorithms

PROB_MACHINE

Author(s)

(C) 2021 - K. Nille-Hauf, T. Feiri, M. Ricker - Hochschule Biberach, Institut fuer Konstruktiven Ingenieurbau

Index