Package 'DDModeling'

February 1, 2020

Type Package

Title A package for the easy simulation of drift diffusion processes in cognitive psychology

Version 0.1.0

Author Thomas Pelzer

Maintainer <thomas.pelzer.arbeit@gmail.com>

Description A package for the easy simulation of drift diffusion processes in cognitive psychology. It grands access to easy modelling and fitting.

License What license is it under?

Encoding UTF-8

LazyData true

Imports Rcpp (>= 1.0.2), parallel, RSAGA, ParallelLogger, ArgumentCheck, methods

LinkingTo Rcpp

RoxygenNote 6.1.1

Collate 'DDFitPar.R'
'DDModel.R'
'DDRep.R'
'DDFit.R'
'Fit_DDModel.R'
'Get_Grid.R'
'RcppExports.R'
'Sim_DDModel.R'
'package-DDModeling.R'

R topics documented:

DDFit-class *An S4 class to represent a Fit of a given DDRep to a model*

Description

An S4 class to represent a Fit of a given DDRep to a model

Slots

`INP_REP` `DDRep` object containing the to be fitted data

`FIT_REP` `DDRep` object containing the fitted data

`MODEL` `DDModel` object containing the model that was used in the fit

`FIT` `DDFitPar` object containing information regarding the fit

DDFitPar-class *An S4 class to represent a drift diffusion model*

Description

An S4 class to represent a drift diffusion model

Slots

`INP_P` `data.frame` containing the to be fitted parameter

`FIT_P` `data.frame` containing the fitted parameter

`FIT_V` `numeric` representing the value of the Fit

`FIT_N` `numeric` representing the number of evaluation points used in the fit

DDModel *Constructor for DDModel-class*

Description

Userfriendly function for the construction of a DDModel-class.

Usage

```
DDModel(model = NULL, task = NULL, conditions = NULL,
  parameter = NULL, dt = NULL, sigma = NULL, CDF_perc = NULL,
  CAF_perc = NULL)
```

Arguments

model	character of the name of the Model to be used (choices are "DSTP","DMC","SSP")
task	character specifying a specific predefined modelstructure ("flanker","custom)
conditions	character vector of the names of conditions
parameter	character vector of the names of custom parameters
dt	numeric representing the integration constant of the diffusion process
sigma	numeric representing the diffusion constant of the diffusion process
CDF_perc	Numeric vector specifying the CDF percentiles (note: numbers equal to absolut percentiles!)
CAF_perc	Numeric vector specifying the CAF percentiles (note: numbers equal to boarders of segments!)

Value

DDModel-class.

Examples

```
M_DSTP <- DDModel(model="DSTP",task = "flanker",CDF_perc = c(0.1,0.3,0.5,0.7,0.9),CAF_perc = c(0.0,0.2,0.4,0.
M_DMC  <- DDModel(model="DMC",task = "flanker",CDF_perc = c(0.1,0.3,0.5,0.7,0.9),CAF_perc = c(0.0,0.2,0.4,0.6
M_Custom <- DDModel(model="DSTP",task="custom",CDF_perc = c(0.1,0.3,0.5,0.7,0.9),CAF_perc = c(0.0,0.2,0.4,0.
```

DDModel-class DDModel *class definition*

Description

DDModel class definition

Slots

ID character that represents the name of the model to be used (i.e. "DSTP","DMC","SSP")

MM list of matrix that contain values which map custom parameters to correspondend modelparameters

DM matrix that contains the domain of all custom parameters (and grid size steps)

SP matrix that contains a set of simulation-parameters important for simulation

RF list of numeric vectors that contain the percentiles of the representation.

DDModeling *DDModeling Pakage*

Description

DDModeling pakage

Details

A pakage for the easy integration of dirft diffusion models in cognitive psychology

Author(s)

Thomas Pelzer

DDRep-class *An S4 class to represent a representation of a drift diffusion simulation*

Description

An S4 class to represent a representation of a drift diffusion simulation

Slots

`RAW` `list` of data.frames that contain the RAW data (i.e. 3 coloumns: $cond $Resp $time)

`REP` `list` of data.frames that contain data representations (CDF and CAF)

`RF` `list` of numeric vectors that contain the percentiles of the representation.

Fit_DDModel *Function to fit a given DDRep to a given DDModel*

Description

Function to fit a given DDRep to a given DDModel

Usage

```
Fit_DDModel(model = NULL, rep = NULL, grid_path = NULL,
  s_sampling = FALSE, trials = 10000)
```

Arguments

`model`	`DDModel` object
`rep`	`DDRep` object or list of `DDRep` objects
`grid_path`	`path` to a direcotey containing a .GRID fileset. If NULL the model will be fitted using 20 randomly drawn startparametersets from the model-DOMAIN.
`s_sampling`	`bool` indicating super sampling while fitting
`trials`	`integer` indicating the number of trials used while fitting (s_sampling = FALSE) or the maximum number of trials used while super sampling (s_sampling = TRUE)

Value

DDFit object

Get_Grid *Generates a Grid from a given DDModel*

Description

After calling the function the user will be instructed to enter the step sizes corresponding to the parameters listet in the used model. Step size should allways be of an integer value, as they represent the number of evaluation points per parameter that are used in the grid. Note that in the given function the evaluation points are allways equally spaced concerning the corresponding parameter domain in the used model. Therefor, If one would like to specify the used evaluation points it is advised to specify the domain in the model.

Usage

```
Get_Grid(model = NULL, path = NULL, name = NULL)
```

Arguments

model	DDModel object
path	character that specifies the full path to the directory in which the Grid should be saved
name	character that represents the name (and subdirectory in path) of the Grid

Value

No direct return value inside of the R-session. The calculated Grid will be saved in the specified path!

Sim_DDModel *Function to simulate a given DDModel*

Description

Function to simulate a given DDModel

Usage

```
Sim_DDModel(model = NULL, trials = NULL)
```

Arguments

model	DDModel object to be used in the simulation.
trials	Numeric specifying the number of trials per condition.

Details

For now this function will only initialize a given model with randomly drawn parameters from the domains specified in the DDModel object.

Value

DDRep object.

Examples

```
M1 <- DDModel(model="DSTP",task = "flanker",CDF_perc = c(0.1,0.3,0.5,0.7,0.9),CAF_perc = c(0.0,0.2,0.4,0.6,0.
```

Index