Package ‘PopED’

March 20, 2015

Type Package
Title Population (and Individual) Optimal Experimental Design
Version 0.2.0
Depends ggplot2
Imports MASS, mvtnorm, nlm, dplyr, codetools
Suggests testthat, Hmisc
Description Optimal experimental designs for both population and individual studies based on nonlinear mixed-effect models. Often this is based on a computation of the Fisher Information Matrix (FIM). This package was developed for pharmacometric problems, and examples and predefined models are available for these types of systems.
License LGPL (>= 3)
URL http://poped.sourceforge.net
BugReports https://github.com/andrewhooker/PopED/issues
Copyright 2014-2015 Andrew C. Hooker
LazyData true
Author Andrew C. Hooker [aut, cre, trl, cph],
  Sebastian Ueckert [aut] (MATLAB version),
  Marco Foracchia [aut] (O-Matrix version),
  Joakim Nyberg [aut] (MATLAB version),
  Eric Stroemberg [ctb] (MATLAB version)
Maintainer Andrew C. Hooker <andrew.hooker@farmbio.uu.se>
NeedsCompilation no
Repository CRAN
Date/Publication 2015-03-20 18:04:11

R topics documented:

  PopED-package ................................................................. 3
  a_line_search ............................................................... 6
R topics documented:

bfgsb_min ..................................................... 9
blockexp ..................................................... 11
blockfinal ................................................... 13
blockheader .................................................. 15
blockopt ..................................................... 18
calc Autofocus ................................................ 20
calc_ofv_and_fim ............................................. 22
calc_ofv_and_grad ............................................ 26
cell ............................................................ 29
convert_variables ............................................ 30
create.poped.database ..................................... 30
create_design ............................................... 41
create_design_space ......................................... 43
diag_matlab .................................................. 46
Doptim ......................................................... 47
downsizing_general_design ................................. 52
Dtrace ........................................................ 53
ed_laplace_ofv ................................................. 57
ed_mftot ...................................................... 61
evaluate.e.ofv.fim .......................................... 64
evaluate.fim .................................................. 68
feps.add ....................................................... 71
feps.add.prop ............................................... 73
feps.prop ..................................................... 74
feval .......................................................... 76
ff.PK.1.comp.oral.md.CL .................................... 77
ff.PK.1.comp.oral.md.KE ................................... 79
ff.PK.1.comp.oral.sd.CL ................................... 80
ff.PK.1.comp.oral.sd.KE ................................... 82
ff.PKPD.1.comp.oral.md.CL.imax ......................... 84
ff.PKPD.1.comp.sd.CL.emax ................................ 86
fileparts ..................................................... 88
getfulld ...................................................... 89
getTruncatedNormal ......................................... 90
get_all_params .............................................. 91
get_rse ....................................................... 92
gradf_eps .................................................... 95
isempty ...................................................... 97
isfield ....................................................... 98
LEDoptim ..................................................... 98
LinMatrixH .................................................. 102
LinMatrixL .................................................. 104
LinMatrixLH ............................................... 105
LinMatrixL_occ ............................................. 107
log_prior_pdf ............................................... 109
mf ........................................................... 111
mf3 .......................................................... 113
mf5 .......................................................... 115
PopED-package

<table>
<thead>
<tr>
<th>Function</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>mf6</td>
<td>117</td>
</tr>
<tr>
<td>mf7</td>
<td>119</td>
</tr>
<tr>
<td>mf8</td>
<td>121</td>
</tr>
<tr>
<td>mfea</td>
<td>124</td>
</tr>
<tr>
<td>mftot</td>
<td>127</td>
</tr>
<tr>
<td>mftot0</td>
<td>129</td>
</tr>
<tr>
<td>mftot1</td>
<td>131</td>
</tr>
<tr>
<td>mftot2</td>
<td>133</td>
</tr>
<tr>
<td>mftot3</td>
<td>134</td>
</tr>
<tr>
<td>mftot4</td>
<td>135</td>
</tr>
<tr>
<td>mftot5</td>
<td>137</td>
</tr>
<tr>
<td>mftot6</td>
<td>139</td>
</tr>
<tr>
<td>mftot7</td>
<td>141</td>
</tr>
<tr>
<td>model_prediction</td>
<td>143</td>
</tr>
<tr>
<td>ofv_criterion</td>
<td>147</td>
</tr>
<tr>
<td>ofv_fim</td>
<td>150</td>
</tr>
<tr>
<td>ones</td>
<td>152</td>
</tr>
<tr>
<td>pargen</td>
<td>153</td>
</tr>
<tr>
<td>plot_efficiency_of_windows</td>
<td>155</td>
</tr>
<tr>
<td>plot_model_prediction</td>
<td>157</td>
</tr>
<tr>
<td>poped.choose</td>
<td>160</td>
</tr>
<tr>
<td>poped_optimize</td>
<td>160</td>
</tr>
<tr>
<td>rand</td>
<td>166</td>
</tr>
<tr>
<td>randn</td>
<td>167</td>
</tr>
<tr>
<td>randperm</td>
<td>168</td>
</tr>
<tr>
<td>RS_opt</td>
<td>168</td>
</tr>
<tr>
<td>RS_opt_gen</td>
<td>172</td>
</tr>
<tr>
<td>size</td>
<td>176</td>
</tr>
<tr>
<td>test_for_max</td>
<td>177</td>
</tr>
<tr>
<td>test_for_min</td>
<td>177</td>
</tr>
<tr>
<td>test_for_zeros</td>
<td>178</td>
</tr>
<tr>
<td>test_mat_size</td>
<td>179</td>
</tr>
<tr>
<td>tic</td>
<td>180</td>
</tr>
<tr>
<td>toc</td>
<td>181</td>
</tr>
<tr>
<td>zeros</td>
<td>182</td>
</tr>
</tbody>
</table>

Index 183

---

**PopED-package**

**PopED - Population (and individual) optimal Experimental Design.**

---

**Description**

PopED computes optimal experimental designs for both population and individual studies based on nonlinear mixed-effect models. Often this is based on a computation of the Fisher Information Matrix (FIM).
Details

To get started you need to define

1. A model.
2. An initial design (and design space if you want to optimize).
3. The tasks to perform.

There are a number of functions to help you with these tasks. The user-level functions defined below are meant to be run with a minimum of arguments (for beginners to advanced users). Many of the other functions in the package (and not listed here) are called by these user-level functions and are often not as user-friendly (developer level or advanced user functions).


Define a residual unexplained variability model (residual error model): feps.add.prop, feps.add, feps.prop.

Create an initial study design (and design space): create.poped.database.

Evaluate the model and/or design through simulation and graphics: plot_model_prediction, model_prediction, plot_efficiency_of_windows.

Evaluate the design using the FIM: evaluate.fim, evaluate.e.ofv.fim, ofv_fim, get_rse.

Optimize the design (evaluate afterwards using the above functions): poped_optimize, RS_opt, a_line_search.

See the "Examples" section below for a short introduction to using the above functions. There are several other examples, as r-scripts, in the "examples" folder in the PopED installation directory located at (run at the R command line):

system.file("examples", package="PopED").

References

3. poped.sf.net
4. https://github.com/andrewhooker/PopED.git

Examples

library(PopED)

#-- Model: One comp first order absorption
#-- Analytic solution for both multiple and single dosing
ff <- function(model_switch, xt, parameters, poped.db){
with(as.list(parameters),{
  y=x
  N = floor(xt/TAU)+1
  y=(DOSE+Favail/V)*((KA/(KA - CL/V)) *
    (exp(-CL/V * (xt - (N - 1) * TAU)) * (1 - exp(-N * CL/V * TAU)))/(1 - exp(-CL/V * TAU)) -
    exp(-KA * (xt - (N - 1) * TAU)) * (1 - exp(-N * KA * TAU)))/(1 - exp(-KA * TAU)))
  return(list( y=y,poped.db=poped.db))
})

## -- parameter definition function
## -- names match parameters in function ff
sfg <- function(x,a,bpop,b,bocc){
  parameters=c( V=bpop[1]*exp(b[1]),
    KA=bpop[2]*exp(b[2]),
    CL=bpop[3]*exp(b[3]),
    Favail=bpop[4],
    DOSE=a[1],
    TAU=a[2])
  return( parameters )
}

## -- Residual unexplained variability (RUV) function
## -- additive + Proportional
feps <- function(model_switch,xt,parameters,epsi,poped.db){
  returnArgs <- do.call(poped.db$model$ff_pointer,list(model_switch,xt,parameters,poped.db))
  y <- returnArgs[[1]]
  poped.db <- returnArgs[[2]]

  y = y*(1+epsi[,1]+epsi[,2])
  return(list( y = y,poped.db =poped.db ))
}

## -- Define design and design space
poped.db <- create.poped.database(ff_file="ff",
  fg_file="sfg",
  fError_file="feps",
  groupsize=20,
  m=2,
  sigma=diag(c(0.04,5e-6)),
  bpop=c(V=72.8,KA=0.25,CL=3.75,Favail=0.9),
  d=c(V=0.09,KA=0.09,CL=0.25*2),
  notfixed_bpop=c(1,1,1,0),
  notfixed_sigma=c(0,0),
  xt=c( 1,2,8,240,245),
  minxt=c(0,0,0,240,240),
  maxxt=c(10,10,10,248,248),
  a=cbind(c(20,40),c(24,24)),
  bUseGrouped_xt=1,
  maxa=c(200,24),
  mina=c(0,24))
a_line_search

## Create plot of model without variability

```r
plot_model_prediction(poped.db)
```

## Not run:

## Create plot of model with variability

```r
plot_model_prediction(poped.db,IPRED=T,DV=T,separate.groups=T)
```

## End(Not run)

## Evaluate initial design

```r
FIM <- evaluate.fim(poped.db)
FIM
det(FIM)
get_rse(FIM,poped.db)
```

## Not run:

### RS+SG+LS optimization of sample times

```r
output <- poped_optimize(poped.db,opt_xt=T)
get_rse(output$fmf,output$poped.db)
plot_model_prediction(output$poped.db,IPRED=F,DV=F)
```

### RS+SG+LS optimization of sample times and doses

```r
output <- poped_optimize(poped.db,opt_xt=T,opt_a=T)
get_rse(output$fmf,output$poped.db)
plot_model_prediction(output$poped.db,IPRED=F,DV=F)
```

### MFEA optimization with only integer times allowed

```r
mfeaN.output <- poped_optimize(poped.db,opt_xt=1,
bUseExchangeAlgorithm=1, EAStepSize=1)
get_rse(mfeaN.output$fmf,mfeaN.output$poped.db)
plot_model_prediction(mfeaN.output$poped.db)
```

### Efficiency of sampling windows

```r
plot_efficiency_of_windows(mfeaN.output$poped.db,xt_windows=0.5)
plot_efficiency_of_windows(mfeaN.output$poped.db,xt_windows=1)
```

## End(Not run)

### a_line_search

**Optimize using line search**

---

**Description**

The function performs a grid search sequentially along design variables. The grid is defined by `ls_step_size`.
Usage

a_line_search(poped.db, out_file = "", bED = FALSE, diff = 0, fmf_initial = 0, dmf_initial = 0, opt_xt = poped.db$settings$optsw[2], opt_a = poped.db$settings$optsw[4], opt_x = poped.db$settings$optsw[3], opt_samps = poped.db$settings$optsw[1], opt_ind = poped.db$settings$optsw[5], ls_step_size = poped.db$settings$ls_step_size)

Arguments

- **poped.db**: A PopED database.
- **out_file**: The output file to write to.
- **bED**: If the algorithm should use E-family methods. Logical.
- **diff**: The OFV difference that is deemed significant for changing a design. If, by changing a design variable the difference between the new and old OFV is less than **diff** the change is not made.
- **fmf_initial**: The initial value of the FIM. If 0 then the FIM is calculated from poped.db.
- **dmf_initial**: The initial value of the objective function value (OFV). If 0 then the OFV is calculated from poped.db.
- **opt_xt**: Should the sample times be optimized?
- **opt_a**: Should the continuous design variables be optimized?
- **opt_x**: Should the discrete design variables be optimized?
- **opt_samps**: Are the number of sample times per group being optimized?
- **opt_ind**: Are the number of individuals per group being optimized?
- **ls_step_size**: Number of grid points in the line search

Value

A list containing:

- **fmf**: The FIM.
- **dmf**: The final value of the objective function value.
- **best_changed**: If the algorithm has found a better design than the starting design.
- **xt**: A matrix of sample times. Each row is a vector of sample times for a group.
- **x**: A matrix for the discrete design variables. Each row is a group.
- **a**: A matrix of covariates. Each row is a group.
- **poped.db**: A PopED database.

See Also

Other Optimize: Doptim; LEDoptim; RS_opt_gen; RS_opt; bfgsb_min; calc autofocus; calc_ofv_and_grad; mfea; poped_optimize
Examples

```r
## Warfarin example from software comparison in:
## Nyberg et al., "Methods and software tools for design evaluation
## for population pharmacokinetics-pharmacodynamics studies",

## Optimization using an additive + proportional residual error to
## avoid sample times at very low concentrations (time 0 or very late samples).
library(PopED)

## find the parameters that are needed to define from the structural model
ff.PK.1.comp.oral.sd.CL

## -- parameter definition function
## -- names match parameters in function ff
sfg <- function(x,a,bpop,bocc){
  parameters=c(CL=bpop[1]*exp(b[1]),
  V=bpop[2]*exp(b[2]),
  KA=bpop[3]*exp(b[3]),
  Favail=bpop[4],
  DOSE=a[1])
  return(parameters)
}

## -- Define initial design and design space
poped.db <- create.poped.database(ff_file="ff.PK.1.comp.oral.sd.CL",
  fg_file="sfg",
  fError_file="feps.add.prop",
  bpop=c(CL=0.15, V=8, KA=1, Favail=1),
  notfixed_bpop=c(1,1,1,0),
  d=c(CL=0.07, V=0.02, KA=0.6),
  sigma=c(0.01,0.25),
  groupsize=32,
  xt=c( 0.5,1,2,6,24,36,72,120),
  minxt=0,
  maxxt=120,
  a=70,
  mina=0,
  maxa=100)

# warfarin optimization model

# should give a warning
output <- a_line_search(poped.db)

# very sparse grid to evaluate (4 points for each design variable)
output <- a_line_search(poped.db,opt_xt=TRUE,ls_step_size=4)

## Not run:

# longer run time
output <- a_line_search(poped.db,opt_xt=TRUE)
```
bfgsb_min

Nonlinear minimization using BFGS with box constraints

Description

This is the implementation of a Broyden Fletcher Goldfarb Shanno (BFGS) method for nonlinear minimization with box constraints.

Usage

bfgsb_min(f_name, f_options, x0, l, u, options = list())

Arguments

- **f_name**: A function name (as a text string) that returns an objective function and the gradient of that objective function, in that order. See `calc_ofv_and_grad` as used in Doptim.
- **f_options**: Options for the f_name argument.
- **x0**: the initial values to optimize
- **l**: the lower bounds
- **u**: the upper bounds
- **options**: a list of additional settings arguments

Value

A list containing:

- **x_k**: The objective function.
- **f_k**: The gradient.
- **B_k**: The hessian.

See Also

Other Optimize: Doptim; LEDoptim; RS_opt_gen; RS_opt; a_line_search; calc Autofocus; calc_ofv_and_grad; mfea; poped_optimize
Examples

## Warfarin example from software comparison in:
## Nyberg et al., "Methods and software tools for design evaluation
## for population pharmacokinetics-pharmacodynamics studies",

## Optimization using an additive + proportional residual error to
## avoid sample times at very low concentrations (time 0 or very late samples).
library(PopED)

## find the parameters that are needed to define from the structural model
ff.PK.1.comp.oral.sd.CL

## -- parameter definition function
## -- names match parameters in function ff
sfg <- function(x,a,bpop,b,bocc){
  parameters=c(CL=bpop[1]*exp(b[1]),
               V=bpop[2]*exp(b[2]),
               KA=bpop[3]*exp(b[3]),
               Favail=bpop[4],
               DOSE=a[1])
  return(parameters)
}

## -- Define initial design and design space
poped.db <- create.poped.database(ff_file="ff.PK.1.comp.oral.sd.CL",
                                   fg_file="sfg",
                                   fError_file="feps.add.prop",
                                   bpop=c(CL=0.15, V=8, KA=1.0, Favail=1),
                                   notfixed_bpop=c(1,1,1,0),
                                   d=c(CL=0.07, V=0.02, KA=0.6),
                                   sigma=c(0.01,0.25),
                                   groupsize=32,
                                   xt=c(0.5,1,2,6,24,36,72,120),
                                   minxt=0,
                                   maxxt=120,
                                   a=70,
                                   mina=0,
                                   maxa=100)

# warfarin optimization model

## Not run:

# BFGS search, DOSE and sample time optimization
bfgs.output <- poped_optimize(poped.db,opt_xt=1,opt_a=0,
                               bUseRandomSearch=0,
                               bUseStochasticGradient=0,
                               bUseBFGSMinimizer=1,
                               bUseLineSearch=0)

f_name <- 'calc_ofv_and_grad'
blockexp <- downsizing_general_design(poped.db)

aa <- @poped.db@settings$cfaa$matrix(1,poped.db$design$m.size(poped.db$design$a,2))
axt=@poped.db@settings$cfaxt$matrix(1,poped.db$design$m,max(poped.db$design_space$maxni))

f_options_1 <- list(gen_des$x,1, 0, gen_des$model_switch,
   aa,axt,gen_des$groupsize,
   gen_des$nii,
   gen_des$xt,gen_des$x,gen_des$a,gen_des$bpop[2,drop=F],
   getfullld(gen_des$d[,2,drop=F],poped.db$parameters$covd),
   poped.db$parameters$sigma,
   getfullld(poped.db$parameters$docc[2,drop=F],
   poped.db$parameters$covdocc),poped.db)

options=list('factr'=poped.db$settings$BFGSConvergenceCriteriaMinStep,
   'ftol'=poped.db$settings$BFGSTolerancef,
   'gtol'=poped.db$settings$BFGSToleranceg,
   'xtol'=poped.db$settings$BFGSTolerancex)

x_k=t(gen_des$xt)
lb=t(gen_des$minxt)
ub=t(gen_des$maxxt)

output <- bfgsb_min(f_name,f_options, x_k,lb,ub,options)

## End(Not run)

---

**blockexp**

*Summarize your experiment for optimization routines*

**Description**

Create some output to the screen and a text file that summarizes the initial design and the design space you will use to optimize.

**Usage**

```r
blockexp(fn, poped.db, e_flag = FALSE, opt_xt = poped.db$settings$optsw[2],
   opt_a = poped.db$settings$optsw[4], opt_x = poped.db$settings$optsw[4],
   opt_samps = poped.db$settings$optsw[1],
   opt_inds = poped.db$settings$optsw[5])
```

**Arguments**

- `fn` The file handle to write to.
A PopED database.

Shoule output be with uncertainty around parameters?

Should the sample times be optimized?

Should the continuous design variables be optimized?

Should the discrete design variables be optimized?

Are the number of sample times per group being optimized?

Are the number of individuals per group being optimized?

See Also

Other Helper: blockfinal; blockheader; blockopt

Examples

```r
## Warfarin example from software comparison in:
## Nyberg et al., "Methods and software tools for design evaluation
## for population pharmacokinetics-pharmacodynamics studies",

## Optimization using an additive + proportional reidual error to
## avoid sample times at very low concentrations (time 0 or very late samples).
library(PopED)

## find the parameters that are needed to define from the structural model
ff.PK.1.comp.orat.so.CL

## -- parameter definition function
## -- names match parameters in function ff
sfg <- function(x,a,bpop,b,bocc){
  parameters=c(CL=bpop[1]*exp(b[1]),
              V=bpop[2]*exp(b[2]),
              KA=bpop[3]*exp(b[3]),
              Favail=bpop[4],
              DOSE=a[1])
  return(parameters)
}

## -- Define initial design and design space
poped.db <- create.poped.database(ff_file="ff.PK.1.comp.orat.so.CL",
                                    ffg_file="sfg",
                                    fError_file="feps.add.prop",
                                    bpop=c(CL=0.15, V=8, KA=1.0, Favail=1),
                                    notfixed_bpop=c(1,1,1,0),
                                    d=c(CL=0.07, V=0.02, KA=0.6),
                                    sigma=c(0.01,0.25),
                                    groupsize=32,
                                    xt=c(0.5,1,2,6,24,36,72,120),
                                    minxt=0,
                                    maxxt=120,
                                    a=70,
```
# warfarin optimization model

blockexp("", poped.db, opt_xt=TRUE)

---

Result function for optimization routines

## Description

Create some output to the screen and a text file that summarizes the problem you solved.

## Usage

```r
blockfinal(fn, fmf, dmf, groupsize, ni, xt, x, a, model_switch, bpop, d, docc,
          sigma, poped.db, opt_xt = poped.db$settings$optsw[2],
          opt_a = poped.db$settings$optsw[4], opt_x = poped.db$settings$optsw[4],
          fmf_init = NULL, dmf_init = NULL, param_cvs_init = NULL,
          compute_inv = TRUE, out_file = NULL, trflag = TRUE,
          footer_flag = TRUE, ...)
```

## Arguments

- **fn**
  - The file handle to write to.
- **fmf**
  - The initial value of the FIM. If set to zero then it is computed.
- **dmf**
  - The initial OFV. If set to zero then it is computed.
- **groupsize**
  - A vector of the number of individuals in each group.
- **ni**
  - A vector of the number of samples in each group.
- **xt**
  - A matrix of sample times. Each row is a vector of sample times for a group.
- **x**
  - A matrix for the discrete design variables. Each row is a group.
- **a**
  - A matrix of covariates. Each row is a group.
- **model_switch**
  - A matrix that is the same size as xt, specifying which model each sample belongs to.
- **bpop**
  - Matrix defining the fixed effects, per row (row number = parameter_number) we should have:
    - column 1 the type of the distribution for E-family designs (0 = Fixed, 1 = Normal, 2 = Uniform, 3 = User Defined Distribution, 4 = lognormal and 5 = truncated normal)
    - column 2 defines the mean.
    - column 3 defines the variance of the distribution (or length of uniform distribution).

Can also just supply the parameter values as a vector c()
d Matrix defining the diagonals of the IIV (same logic as for the fixed effects). can also just supply the parameter values as a c().

docc Matrix defining the IOV, the IOV variances and the IOV distribution

sigma Matrix defining the variances can covariances of the residual variability terms of the model. can also just supply the diagonal parameter values (variances) as a c().

poped.db A PopED database.

opt_xt Should the sample times be optimized?

opt_a Should the continuous design variables be optimized?

opt_x Should the discrete design variables be optimized?

fim_init Initial FIM.

dmf_init Initial OFV.

param_cvs_init The initial design parameter RSE values.

compute_inv should the inverse of the FIM be used to compute expected RSE values? Often not needed except for diagnostic purposes.

out_file Which file should the output be directed to? A string, a file handle using file or "" will output to the screen.

trflag Should the optimization be output to the screen and to a file?

footer_flag Should the footer text be printed out?

... arguments passed to evaluate.fim and ofv.fim.

See Also

Other Helper: blockexp; blockheader; blockopt

Examples

## Warfarin example from software comparison in:
## Nyberg et al., "Methods and software tools for design evaluation
## for population pharmacokinetics-pharmacodynamics studies",

## Optimization using an additive + proportional residual error to
## avoid sample times at very low concentrations (time 0 or very late samples).
library(PopED)

## find the parameters that are needed to define from the structural model
ff.PK.1.comp.oral.sd.CL

## -- parameter definition function
## -- names match parameters in function ff
sfg <- function(x,a,bpop,b,boc){
  parameters=c(CL=bpop[1]*exp(x[1]),
             V=bpop[2]*exp(x[2]),
             KA=bpop[3]*exp(x[3]),
             Favail=bpop[4],
             ...)
```r
dose=a[1])
return(parameters)
}

## -- Define initial design and design space
poped.db <- create.poped.database(ff_file="FF.PK.1.comp.oral.sd.CL",
fg_file="sfg",
fError_file="feps.add.prop",
bpop=c(CL=0.15, V=8, KA=1.0, Favail=1),
notfixed_bpop=c(1,1,1,0),
d=c(CL=0.07, V=0.02, KA=0.6),
sigma=c(0.01,0.25),
groupsize=32,
x=c(0.5,1,2,6,24,36,72,120),
minx=0,
maxx=120,
a=70,
mina=0,
maxa=100)

# warfarin optimization model

FIM <- evaluate.fim(poped.db)
dmf <- det(FIM)

blockfinal(fn="",fmf=FIM,
dmf=dmf,
groupsize=poped.db$design$groupsize,
ni=poped.db$design$ni,
x=poped.db$design$x,a=poped.db$design$a,
model_switch=poped.db$design$model_switch,
poped.db$parameters$param.pt.val$bpop,
poped.db$parameters$param.pt.val$d,
poped.db$parameters$docc,
poped.db$parameters$param.pt.val$sigma,
poped.db,
opt_xt=TRUE,
fmf_init=FIM,
dmf_init=dmf,
param_cvs_init=cbind(get_rse(FIM,poped.db,use_percent=FALSE)))
```

---

**Header function for optimization routines**

**Description**

Create some output to the screen and a text file that summarizes the problem you are trying to solve.
Usage

```r
blockheader(poped.db, name = "Default", iter = NULL, 
  e_flag = !(poped.db$settings$d_switch), 
  opt_xt = poped.db$settings$optsw[2], opt_a = poped.db$settings$optsw[4], 
  opt_x = poped.db$settings$optsw[4], 
  opt_samps = poped.db$settings$optsw[1], 
  opt_inds = poped.db$settings$optsw[5], fmf = 0, dmf = 0, bpop = NULL, 
  d = NULL, docc = NULL, sigma = NULL, 
  name_header = poped.db$settings$strOutputFileName, 
  file_path = poped.db$settings$strOutputFilePath, out_file = NULL, 
  compute_inv = TRUE, trflag = TRUE, header_flag = TRUE, ...)```

Arguments

- **poped.db**: A PopED database.
- **name**: The name used for the output file. Combined with `name_header` and `iter`. If "" then output is to the screen.
- **iter**: The last number in the name printed to the output file, combined with name.
- **e_flag**: Should output be with uncertainty around parameters?
- **opt_xt**: Should the sample times be optimized?
- **opt_a**: Should the continuous design variables be optimized?
- **opt_x**: Should the discrete design variables be optimized?
- **opt_samps**: Are the number of sample times per group being optimized?
- **opt_inds**: Are the number of individuals per group being optimized?
- **fmf**: The initial value of the FIM. If set to zero then it is computed.
- **dmf**: The initial OFV. If set to zero then it is computed.
- **bpop**: Matrix defining the fixed effects, per row (row number = parameter_number) we should have:
  - column 1 the type of the distribution for E-family designs (0 = Fixed, 1 = Normal, 2 = Uniform, 3 = User Defined Distribution, 4 = lognormal and 5 = truncated normal)
  - column 2 defines the mean.
  - column 3 defines the variance of the distribution (or length of uniform distribution).
  Can also just supply the parameter values as a vector c().
- **d**: Matrix defining the diagonals of the IIV (same logic as for the fixed effects). can also just supply the parameter values as a c().
- **docc**: Matrix defining the IOV, the IOV variances and the IOV distribution
- **sigma**: Matrix defining the variances can covariances of the residual variability terms of the model. can also just supply the diagonal parameter values (variances) as a c().
- **name_header**: The initial portion of the file name.
file_path  The path to where the file should be created.
out_file   Which file should the output be directed to? A string, a file handle using file or "" will output to the screen.
compute_inv should the inverse of the FIM be used to compute expected RSE values? Often not needed except for diagnostic purposes.
trflag     Should the optimization be output to the screen and to a file?
header_flag Should the header text be printed out?
...    Additional arguments passed to further functions.

Value

fn A file handle (or ' ' if name=' ')

See Also

Other Helper: blockexp; blockfinal; blockopt

Examples

## Warfarin example from software comparison in:
## Nyberg et al., "Methods and software tools for design evaluation
## for population pharmacokinetics-pharmacodynamics studies",

## Optimization using an additive + proportional residual error to
## avoid sample times at very low concentrations (time 0 or very late samples).
library(PopED)

## find the parameters that are needed to define from the structural model
ff.PK.1.comp.oral.sd.CL

## -- parameter definition function
## -- names match parameters in function ff
sfg <- function(x,a,bpop,bocc){
  parameters=c(CL=bpop[1]*exp(b[1]),
              V=bpop[2]*exp(b[2]),
              KA=bpop[3]*exp(b[3]),
              Favail=bpop[4],
              DOSE=a[1])
  return(parameters)
}

## -- Define initial design and design space
poped.db <- create.poped.database(ff_file="ff.PK.1.comp.oral.sd.CL",
                                   fg_file="sfg",
                                   fError_file="feps.add.prop",
                                   bpop=c(CL=0.15, V=8, KA=1.0, Favail=1),
                                   notfixed_bpop=c(1,1,1,0),
                                   d=c(CL=0.07, V=0.02, KA=0.6),
                                   sigma=c(0.81,0.25),
# warfarin optimization model

groupsize=32,
    xt=c(0.5, 1, 2, 6, 24, 36, 72, 120),
    minxt=0,
    maxxt=120,
    a=70,
    mina=0,
    maxa=100)

FIM <- evaluate.fim(poped.db)
dmf <- det(FIM)

blockheader(poped.db, name="")

blockheader(name="", iter=1, poped.db)

blockheader(name='',
    iter=1,
    poped.db,
    e_flag=FALSE,
    opt_xt=TRUE,
    opt_a=TRUE, opt_x=poped.db$settings$optsw[4],
    opt_samps=poped.db$settings$optsw[1], opt_inds=poped.db$settings$optsw[5],
    fmf=FIM, dmf=dmf,
    bpop=poped.db$parameters$param.pt.val$bpop,
    d=poped.db$parameters$param.pt.val$d,
    docc=poped.db$parameters$docc, sigma=poped.db$parameters$param.pt.val$sigma)

blockheader(name='',
    iter=1,
    poped.db,
    e_flag=TRUE,
    opt_xt=TRUE,
    opt_a=TRUE, opt_x=poped.db$settings$optsw[4],
    opt_samps=poped.db$settings$optsw[1], opt_inds=poped.db$settings$optsw[5],
    fmf=FIM, dmf=dmf,
    bpop=poped.db$parameters$param.pt.val$bpop,
    d=poped.db$parameters$param.pt.val$d,
    docc=poped.db$parameters$docc, sigma=poped.db$parameters$param.pt.val$sigma)

blockopt

---

Summarize your optimization settings for optimization routines
Description

Create some output to the screen and a text file that summarizes the optimization settings you will use to optimize.

Usage

blockopt(fn, poped.db, opt_method = "")

Arguments

fn The file handle to write to.
poped.db A PopED database.
opt_method If "RS" (random search), "SG" (stochastic gradient) or "DO" (discrete optimization) then specific output is produced.

See Also

Other Helper: blockexp; blockfinal; blockheader

Examples

## Warfarin example from software comparison in:
## Nyberg et al., "Methods and software tools for design evaluation
## for population pharmacokinetics-pharmacodynamics studies",

## Optimization using an additive + proportional residual error to
## avoid sample times at very low concentrations (time 0 or very late samples).
library(PopED)

## find the parameters that are needed to define from the structural model
ff.PK.1.comp.oral.sd.CL

## -- parameter definition function
## -- names match parameters in function ff
sfg <- function(x,a,bpop,bocc){
  parameters=c(CL=bpop[1]*exp(b[1]),
             V=bpop[2]*exp(b[2]),
             KA=bpop[3]*exp(b[3]),
             Favail=bpop[4],
             DOSE=a[1])
  return(parameters)
}

## -- Define initial design and design space
poped.db <- create.poped.database(ff_file="ff.PK.1.comp.oral.sd.CL",
                                   fg_file="sfg",
                                   fError_file="feps.add.prop",
                                   bpop=c(CL=0.15, V=8, KA=1.0, Favail=1),
                                   notfixed_bpop=c(1,1,1,0),
                                   d=c(CL=0.07, V=0.02, KA=0.6),
                                   ...
description
Compute the autofocus portion of the stochastic gradient routine

usage

calc_autofocus(m, ni_var, dmf, varopt, varopto, maxvar, minvar, gradvar, normgvar, avar, model_switch, groupsize, xtopt, xopt, aopt, ni, bpop, d, sigma, docc, poped.db)

arguments

m Number of groups in the study. Each individual in a group will have the same design.

ni_var The ni_var.

dmf The initial OFV. If set to zero then it is computed.

varopt The varopt.

varopto The varopto.

maxvar The maxvar.

minvar The minvar.

gradvar The gradvar.

normgvar The normgvar.

avar The avar.

model_switch A matrix that is the same size as xt, specifying which model each sample belongs to.

groupsize A vector of the number of individuals in each group.

xtopt The optimal sampling times matrix.
The optimal discrete design variables matrix.

The optimal continuous design variables matrix.

A vector of the number of samples in each group.

Matrix defining the fixed effects, per row (row number = parameter number) we should have:

- column 1 the type of the distribution for E-family designs (0 = Fixed, 1 = Normal, 2 = Uniform, 3 = User Defined Distribution, 4 = lognormal and 5 = truncated normal)
- column 2 defines the mean.
- column 3 defines the variance of the distribution (or length of uniform distribution).

Can also just supply the parameter values as a vector c().

Matrix defining the diagonals of the IIV (same logic as for the fixed effects). Can also just supply the parameter values as a c().

Matrix defining the variances can covariances of the residual variability terms of the model. Can also just supply the diagonal parameter values (variances) as a c().

Matrix defining the IOV, the IOV variances and the IOV distribution.

A PopED database.

A list containing:

- `navar` The autofocus parameter.
- `poped.db` PopED database.

Other Optimize: `Doptim; LEDoptim; RS_opt_gen; RS_opt; a_line_search; bfgsb_min; calc_ofv_and_grad; mfea; poped_optimize`

```
## Warfarin example from software comparison in:
## Nyberg et al., "Methods and software tools for design evaluation
## for population pharmacokinetics-pharmacodynamics studies",

## Optimization using an additive + proportional residual error to
## avoid sample times at very low concentrations (time 0 or very late samoples).
library(PopED)

## find the parameters that are needed to define from the structural model
ff.PK.1.comp.oral.sd.CL

## -- parameter definition function
```
### Description

This function computes the expectation of the FIM and OFV(FIM) for either point values of parameter estimates or parameter distributions given the model, parameters, distributions of parameter uncertainty, design and methods defined in the PopED database.
calc_ofv_and_fim

Usage

calc_ofv_and_fim(poped.db, ofv = 0, fim = 0, 
d_switch = poped.db$settings$d_switch, 
bpopdescr = poped.db$parameters$bpop, ddescr = poped.db$parameters$d, 
bpop = bpopdescr[, 2, drop = F], d = getfulld(ddescr[, 2, drop = F], 
poped.db$parameters$covd), docc_full = getfulld(poped.db$parameters$docc[, 
2, drop = F], poped.db$parameters$covdocc), 
model_switch = poped.db$design$model_switch, ni = poped.db$design$ni, 
xt = poped.db$design$xt, x = poped.db$design$x, a = poped.db$design$a, 
fim.calc.type = poped.db$settings$stFIMCalculationType, 
use_laplace = poped.db$settings$slEDCalculationType, laplace.fim = FALSE, 
...
)

Arguments

poped.db
A PopED database.
ofv
The current ofv. If other than zero then this values is simply returned unchanged.
fim
The current FIM. If other than zero then this values is simply returned unchanged.
d_switch
• **START OF CRITERION SPECIFICATION OPTIONS**********
D-family design (1) or ED-familty design (0) (with or without parameter uncertainty)
bpopdescr
Matrix defining the fixed effects, per row (row number = parameter_number) we should have:
• column 1 the type of the distribution for E-family designs (0 = Fixed, 1 = Normal, 2 = Uniform, 3 = User Defined Distribution, 4 = lognormal and 5 = truncated normal)
• column 2 defines the mean.
• column 3 defines the variance of the distribution (or length of uniform distribution).
ddescr
Matrix defining the diagonals of the IIV (same logic as for the bpopdescr).
bpop
Matrix defining the fixed effects, per row (row number = parameter_number) we should have:
• column 1 the type of the distribution for E-family designs (0 = Fixed, 1 = Normal, 2 = Uniform, 3 = User Defined Distribution, 4 = lognormal and 5 = truncated normal)
• column 2 defines the mean.
• column 3 defines the variance of the distribution (or length of uniform distribution).
Can also just supply the parameter values as a vector c()
d
Matrix defining the diagonals of the IIV (same logic as for the fixed effects). 
can also just supply the parameter values as a c().
docc_full
A between occasion variability matrix.
model_switch A matrix that is the same size as xt, specifying which model each sample belongs to.
ni A vector of the number of samples in each group.
xt A matrix of sample times. Each row is a vector of sample times for a group.
x A matrix for the discrete design variables. Each row is a group.
a A matrix of covariates. Each row is a group.
fim.calc.type The method used for calculating the FIM. Potential values:
• 0 = Full FIM. No assumption that fixed and random effects are uncorrelated. See mftot0.
• 1 = Reduced FIM. Assume that there is no correlation in the FIM between the fixed and random effects, and set these elements in the FIM to zero. See mftot1.
• 2 = weighted models (placeholder).
• 3 = Not currently used.
• 4 = Reduced FIM and computing all derivatives with respect to the standard deviation of the residual unexplained variation (sqrt(SIGMA) in NONMEM). This matches what is done in PFIM, and assumes that the standard deviation of the residual unexplained variation is the estimated parameter (NOTE: NONMEM estimates the variance of the residual unexplained variation by default). See mftot4.
• 5 = Full FIM parameterized with A,B,C matrices & derivative of variance. See mftot5.
• 6 = Calculate one model switch at a time, good for large matrices. See mftot6.
• 7 = Reduced FIM parameterized with A,B,C matrices & derivative of variance. See mftot7.

use_laplace Should the Laplace method be used in calculating the expectation of the OFV?
laplace.fim Should an E(FIM) be calculated when computing the Laplace approximated E(OFV)? Typically the FIM does not need to be computed and, if desired, this calculation is done using the standard MC integration technique, so can be slow.

Value
A list containing the FIM and OFV(FIM) or the E(FIM) and E(OFV(FIM)) according to the function arguments.

See Also
Other E-family: ed_laplace_ofv; ed_mftot; evaluate.e.ofv.fim
Other FIM: LinMatrixH; LinMatrixLH; LinMatrixL_occ; ed_laplace_ofv; ed_mftot; evaluate.e.ofv.fim; evaluate.fim; gradf_eps; mf3; mf5; mf6; mf7; mf8; mftot0; mftot1; mftot2; mftot3; mftot4; mftot5; mftot6; mftot7; mftot; mf; ofvCriterion; ofv_fim
Other evaluate_FIM: evaluate.e.ofv.fim; evaluate.fim; ofv_fim
Examples

```r
## Warfarin example from software comparison in:
## Nyberg et al., "Methods and software tools for design evaluation
## for population pharmacokinetics-pharmacodynamics studies",

## Optimization using an additive + proportional residual error to
## avoid sample times at very low concentrations (time 0 or very late samples).
library(PopED)

## find the parameters that are needed to define from the structural model
ff.PK.1.comp.oral.sd.CL

## -- parameter definition function
## -- names match parameters in function ff
sfg <- function(x,a,bpop,b,bocc){
  parameters=c(CL=bpop[1]*exp(b[1]),
               V=bpop[2]*exp(b[2]),
               KA=bpop[3]*exp(b[3]),
               Favail=bpop[4],
               DOSE=a[1])
  return(parameters)
}

# Adding 10% log-normal Uncertainty to fixed effects (not Favail)
bpop_vals <- c(CL=0.15, V=8, KA=1.0, Favail=1)
bpop_vals_ed_ln <- cbind(ones(length(bpop_vals),1)*4, # log-normal distribution
                         bpop_vals,
                         ones(length(bpop_vals),1)*(bpop_vals*0.1)^2) # 10% of bpop value
bpop_vals_ed_ln["Favail",] <- c(0,1,0)
bpop_vals_ed_ln

## -- Define initial design and design space
poped.db <- create.poped.database(ff_file="ff.PK.1.comp.oral.sd.CL",
                                   fg_file="sfg",
                                   fError_file="feps.add.prop",
                                   bpop=bpop_vals_ed_ln,
                                   notfixed_bpop=c(1,1,1,0),
                                   d=c(CL=0.07, V=0.02, KA=0.6),
                                   sigma=c(0.01,0.25),
                                   groupsize=32,
                                   xt=c( 0.5,1,2,6,24,36,72,120),
                                   minxt=0,
                                   maxxt=120,
                                   a=70,
                                   mina=0,
                                   maxa=100)

# warfarin ed model
calc_ofv_and_fim(poped.db)

## Not run:
```
calc_ofv_and_grad(poped.db,d_switch=0)
calc_ofv_and_grad(poped.db,d_switch=0,use_laplace=TRUE)
calc_ofv_and_grad(poped.db,d_switch=0,use_laplace=TRUE,laplace.fim=TRUE)

## End(Not run)

---

**calc_ofv_and_grad**  
*Compute an objective function and gradient*

---

**Description**

Compute an objective function and gradient with respect to the optimization parameters. This function can be passed to the Broyden Fletcher Goldfarb Shanno (BFGS) method for nonlinear minimization with box constraints implemented in `bfgsb_min`.

**Usage**

```r
calc_ofv_and_grad(x, optxt, opta, model_switch, aa, axt, groupsize, ni, xtopto, xopto, aopto, bpop, d, sigma, docc_full, poped.db, only_fim = FALSE)
```

**Arguments**

- `x`: A matrix for the discrete design variables. Each row is a group.
- `optxt`: If sampling times are optimized
- `opta`: If continuous design variables are optimized
- `model_switch`: A matrix that is the same size as `xt`, specifying which model each sample belongs to.
- `aa`: The `aa` value
- `axt`: the `axt` value
- `groupsize`: A vector of the number of individuals in each group.
- `ni`: A vector of the number of samples in each group.
- `xtopto`: the `xtopto` value
- `xopto`: the `xopto` value
- `aopto`: the `aopto` value
- `bpop`: Matrix defining the fixed effects, per row (row number = parameter_number) we should have:
  - column 1 the type of the distribution for E-family designs (0 = Fixed, 1 = Normal, 2 = Uniform, 3 = User Defined Distribution, 4 = lognormal and 5 = truncated normal)
  - column 2 defines the mean.
  - column 3 defines the variance of the distribution (or length of uniform distribution).
Can also just supply the parameter values as a vector c().

d Matrix defining the diagonals of the IIV (same logic as for the fixed effects). can also just supply the parameter values as a c().
sigma Matrix defining the variances can covariances of the residual variability terms of the model. can also just supply the diagonal parameter values (variances) as a c().
docc_full A between occasion variability matrix.
poped.db A PopED database.
only_fim Should the gradient be calculated?

Value

A list containing:

f The objective function.
g The gradient.

See Also

Other Optimize: Doptim; LEDoptim; RS_opt_gen; RS_opt; a_line_search; bfgsb_min; calc_autofocus; mfea; poped_optimize

Examples

## Warfarin example from software comparison in:
## Nyberg et al., "Methods and software tools for design evaluation
## for population pharmacokinetics-pharmacodynamics studies",

## Optimization using an additive + proportional residual error to
## avoid sample times at very low concentrations (time 0 or very late samples).
library(PopED)

## find the parameters that are needed to define from the structural model
ff.PK.1.comp.oral.sd.CL

## -- parameter definition function
## -- names match parameters in function ff
sfg <- function(x,a,bpop,bocc){
  parameters=c(CL=bpop[1]*exp(b[1]),
    V=bpop[2]*exp(b[2]),
    KA=bpop[3]*exp(b[3]),
    Favail=bpop[4],
    DOSE=a[1])
  return(parameters)
}

## -- Define initial design and design space
poped.db <- create.poped.database(ff_file="ff.PK.1.comp.oral.sd.CL",
                                   fg_file="sfg",
fError_file="feps.add_prop",
bpop=c(CL=0.15, V=8, KA=1.0, Favail=1),
notfixed_bpop=c(1.1,1,0),
d=c(CL=0.07, V=0.02, KA=0.6),
sigma=c(0.01,0.25),
groupsize=32,
xt=c( 0.5,1,2,6,24,36,72,120),
minxt=0,
maxxt=120,
a=70,
mina=0,
maxa=100)

# warfarin optimization model

opta=TRUE
aa=opta*poped.db$settings$cfaa*matrix(1,poped.db$design$m,size(poped.db$design$a,2))

optxt=TRUE
axt=optxt*poped.db$settings$cfaxt*matrix(1,poped.db$design$m,max(poped.db$design_space$maxni))

calc_ofv_and_grad(x=c(poped.db$design$xt,poped.db$design$a),
optxt=optxt, opta=opta,
model_switch=poped.db$design$model_switch,
aa=aa,
axt=axt,
groupsize=poped.db$design$groupsize,
ni=poped.db$design$ni,
xtopo=poped.db$design$xt,
xopto=poped.db$design$x,
aopto=poped.db$design$a,
bpop=poped.db$parameters$param.pt.val$bpop,
d=poped.db$parameters$param.pt.val$d,
sigma=poped.db$parameters$param.pt.val$sigma,
docc_full=poped.db$parameters$param.pt.val$docc,
poped.db,
only_fim=FALSE)

## Not run:

# BFGS search, DOSE and sample time optimization
bfgs.output <- poped_optimize(poped.db, opt_xt=1, opt_a=1,
bUseRandomSearch= 0,
bUseStochasticGradient = 0,
bUseBFSGMinimizer = 1,
bUseLineSearch = 0)

## End(Not run)
cell  MATLAB cell function

Description

Create a cell array as in MATLAB.

Usage

cell(…)

Arguments

… Dimensions for the cell array.

Value

A list of empty lists.

Note

This is a modified version of the same function in cell

See Also

Other MATLAB: diag_matlab; feval; fileparts; isempty; isfield; ones; randn; randperm; rand; size; tic; toc; zeros

Examples

    cell(3)
    cell(2,3)

    # define possible values of 2 categorical design variable
    x.space <- cell(1,2)
    x.space[1,1] <- list(seq(10,100,10))
    x.space[1,2] <- list(seq(10,300,10))
    x.space
    x.space[1,1]
    x.space[1,2]
convert_variables 

Create global variables in the PopED database

Description

Function takes design variables from input files and converts them to the global variables needed in PopED. Typically not used by the user. Instead use the function create.poped.database.

Usage

convert_variables(poped.db)

Arguments

poped.db A PopED database

Value

A PopED database

See Also

Other poped_input: create.poped.database; create_design_space; create_design; downsizing_general_design; poped.choose

create.poped.database 

Create a PopED database

Description

This function takes the input file supplied by the user, or function arguments, and creates a database that can then be used to run all other PopED functions. The function supplies default values to elements of the database that are not specified in the input file or as function arguments. Default arguments are supplied in the Usage section (easiest to use a text search to find values you are interested in).

Usage

create.poped.database(popedInput = list(),
    ff_file = poped.choose(popedInput[["ff_file"]], "ff"), ff_fun = NULL,
    fg_file = poped.choose(popedInput$fg_file, "sfg"), fg_fun = NULL,
    fError_file = poped.choose(popedInput$fError_file, "feps"),
    fError_fun = NULL, optsw = poped.choose(popedInput$optsw, cbind(0, 0, 0, 0), xt = poped.choose(popedInput$design[["xt"]],
    stop("xt' needs to be defined")), m = poped.choose(popedInput[["m"]],
    NULL), x = poped.choose(popedInput$design[["x"]], NULL),
    ... )
nx = poped.choose(popedInput$nx, NULL),
a = poped.choose(popedInput$design[['a']], NULL),
na = poped.choose(popedInput$na, NULL),
groupsize = poped.choose(popedInput$design$groupsize,
  stop("groupsize' needs to be defined")),
i = poped.choose(popedInput$design$i, NULL),
model_switch = poped.choose(popedInput$design$model_switch, NULL),
maxni = poped.choose(popedInput$maxni, NULL),
mmini = poped.choose(popedInput$mmini, NULL),
maxtotni = poped.choose(popedInput$maxtотni, NULL),
mintotni = poped.choose(popedInput$mintotni, NULL),
maxgroupsize = poped.choose(popedInput$design$maxgroupsize, NULL),
mingroupsize = poped.choose(popedInput$design$mingroupsize, NULL),
maxtotgroupsize = poped.choose(popedInput$design$maxtотgroupsize, NULL),
mintotgroupsize = poped.choose(popedInput$design$mintotgroupsize, NULL),
maxxt = poped.choose(popedInput$design$maxxt, NULL),
mintxt = poped.choose(popedInput$design$mintxt, NULL),
discrete_x = poped.choose(popedInput$design$discrete_x, NULL),
maxa = poped.choose(popedInput$design$maxa, NULL),
mina = poped.choose(popedInput$design$mina, NULL),
bUseGrouped_xt = poped.choose(popedInput$bUseGrouped_xt, FALSE),
G_xt = poped.choose(popedInput$design$G_xt, NULL),
bUseGrouped_a = poped.choose(popedInput$bUseGrouped_a, FALSE),
G_a = poped.choose(popedInput$design$G_a, NULL),
bUseGrouped_x = poped.choose(popedInput$bUseGrouped_x, FALSE),
G_x = poped.choose(popedInput$design$G_x, NULL),
iFICalculationType = poped.choose(popedInput$iFICalculationType, 1),
iApproximationMethod = poped.choose(popedInput$iApproximationMethod, 0),
iFOCENumInd = poped.choose(popedInput$iFOCENumInd, 1000),
prior_fim = poped.choose(popedInput$prior_fim, matrix(0, 0, 1)),
strAutoCorrelationFile = poped.choose(popedInput$strAutoCorrelationFile,
  ""),
d_switch = poped.choose(popedInput$d_switch, 1),
ofv_calc_type = poped.choose(popedInput$ofv_calc_type, 1),
ds_index = popedInput$CriterionOptions$ds_index,
strEDPenaltyFile = poped.choose(popedInput$strEDPenaltyfile, ""),
iEDCalculationType = poped.choose(popedInput$iEDCalculationType, 0),
ED_samp_size = poped.choose(popedInput$ED_samp_size, 45),
bLHS = poped.choose(popedInput$bLHS, 1),
strUserDistributionFile = poped.choose(popedInput$strUserDistributionFile,
  ""),
nbpop = popedInput$nbpop, NumRanEff = popedInput$n,
NumDocc = popedInput$ndocc, NumOcc = popedInput$NumOcc,
ng = popedInput$ng, bpop = poped.choose(popedInput$design$bpop,
  stop("bpop must be defined")),
d = poped.choose(popedInput$design$d,
  stop("d must be defined")),
covd = popedInput$design$covd,
sigma = popedInput$design$sigma,
docc = poped.choose(popedInput$design$docc, matrix(0, 0, 3)),
covdocc = poped.choose(popedInput$design$covdocc, zeros(1, length(docc[, 2,
drop = F]) * (length(docc[, 2, drop = F]) - 1)/2)),

create.poped.database

notfixed_bpop = popedInput$notfixed_bpop,
notfixed_d = popedInput$notfixed_d,
notfixed_cvd = popedInput$notfixed_cvd,
notfixed_docc = popedInput$notfixed_docc,
notfixed_covdocc = poped.choose(popedInput$notfixed_covdocc, zeros(1, length(covdocc))),
notfixed_sigma = poped.choose(popedInput$notfixed_sigma, t(rep(1, size(sigma, 2))))
notfixed_covsigma = poped.choose(popedInput$notfixed_covsigma, zeros(1, length(notfixed_sigma) * (length(notfixed_sigma) - 1)/2)),
bUseRandomSearch = poped.choose(popedInput$bUseRandomSearch, TRUE),
bUseStochasticGradient = poped.choose(popedInput$bUseStochasticGradient, TRUE),
bUseLineSearch = poped.choose(popedInput$bUseLineSearch, TRUE),
bUseExchangeAlgorithm = poped.choose(popedInput$bUseExchangeAlgorithm, FALSE),
bUseBFGSMinimizer = poped.choose(popedInput$bUseBFGSMinimizer, FALSE),
EACriteria = poped.choose(popedInput$EAxCriteria, 1),
strRunFile = poped.choose(popedInput$strRunFile, ""),
poped_version = poped.choose(popedInput$strPopEDVersion, packageVersion("PopED")),
modtit = poped.choose(popedInput$modtit, "PopED model"),
output_file = poped.choose(popedInput$output_file, paste("PopED_output","_summary", sep = "")),
output_function_file = poped.choose(popedInput$output_function_file, paste("PopED", "_out_file", sep = "")),
strIterationFileName = poped.choose(popedInput$strIterationFileName, paste("PopED","\_current.R", sep = "")),
user_data = poped.choose(popedInput$user_data, cell(0, 0)),
ourzero = poped.choose(popedInput$ourzero, 1e-05),
dSeed = poped.choose(popedInput$dSeed, -1),
line_opta = poped.choose(popedInput$line_opta, NULL),
line_optx = poped.choose(popedInput$line_optx, NULL),
bShowGraphs = poped.choose(popedInput$bShowGraphs, FALSE),
use_logfile = poped.choose(popedInput$use_logfile, FALSE),
m1_switch = poped.choose(popedInput$m1_switch, 1),
m2_switch = poped.choose(popedInput$m2_switch, 1),
hle_switch = poped.choose(popedInput$hle_switch, 1),
gradff_switch = poped.choose(popedInput$gradff_switch, 1),
gradfg_switch = poped.choose(popedInput$gradfg_switch, 1),
rsit_output = poped.choose(popedInput$rsit_output, 5),
sgit_output = poped.choose(popedInput$sgit_output, 1),
hm1 = poped.choose(popedInput$hm1, 1e-05),
hlf = poped.choose(popedInput$hlf, 1e-05),
hlg = poped.choose(popedInput$hlg, 1e-05),
hm2 = poped.choose(popedInput$hm2, 1e-05),
hgd = poped.choose(popedInput$hgd, 1e-05),
hle = poped.choose(popedInput$hle, 1e-05),
AbsTol = poped.choose(popedInput$AbsTol, 1e-05),
RelTol = poped.choose(popedInput$RelTol, 1e-05),
iDiffSolverMethod = poped.choose(popedInput$iDiffSolverMethod, 0),
bUseMemorySolver = poped.choose(popedInput$bUseMemorySolver, FALSE),
rsit = poped.choose(popedInput$rsit, 300),
sgit = poped.choose(popedInput$sgit, 150),
intrsit = poped.choose(popedInput$intrsit, 250),
intrsgit = poped.choose(popedInput$intrsgit, 50),
maxrsnullit = poped.choose(popedInput$maxrsnullit, 50),
convergence_eps = poped.choose(popedInput$convergence_eps, 1e-08),
rs1xt = poped.choose(popedInput$rs1xt, 10),
rs1la = poped.choose(popedInput$rs1la, 10),
cfaxt = poped.choose(popedInput$cfaxt, 0.001),
cfaa = poped.choose(popedInput$cfaa, 0.001),
bGreedyGroupOpt = poped.choose(popedInput$bGreedyGroupOpt, FALSE),
EAStepSize = poped.choose(popedInput$EAStepSize, 0.01),
EANumPoints = poped.choose(popedInput$EANumPoints, FALSE),
EAConvergenceCriteria = poped.choose(popedInput$EAConvergenceCriteria,
  1e-20), bEANoReplicates = poped.choose(popedInput$bEANoReplicates, FALSE),
BFGSConvergenceCriteriaMinStep = poped.choose(popedInput$BFGSConvergenceCriteriaMinStep,
  1e-08),
BFGSProjectedGradient Tol = poped.choose(popedInput$BFGSProjectedGradient Tol,
  1e-04), BFGSTolerancef = poped.choose(popedInput$BFGSTolerancef, 0.001),
BFGSTolerancecg = poped.choose(popedInput$BFGSTolerancecg, 0.9),
BFGSTolerancecx = poped.choose(popedInput$BFGSTolerancecx, 0.1),
ED_diff_it = poped.choose(popedInput$ED_diff_it, 30),
ED_diff_percent = poped.choose(popedInput$ED_diff_percent, 10),
line_search_it = poped.choose(popedInput$line_search_it, 50),
Doptim_iter = poped.choose(popedInput$iNumSearchItera tionsIfNotLineSearch,
  1),
icompileoption = poped.choose(popedInput$parallelSettings$icompileoption,
-1),
iUseParallelMethod = poped.choose(popedInput$parallelSettings$iUseParallelMethod,
  1),
MCC_Dep = poped.choose(popedInput$parallelSettings$MCC_Dep, FALSE),
strExecuteName = poped.choose(popedInput$parallelSettings$strExecuteName,
  "calc_fim.exe"),
inNumProcesses = poped.choose(popedInput$parallelSettings$inNumProcesses, 2),
inNumChunkDesignEvals = poped.choose(popedInput$parallelSettings$inNumChunkDesignEvals,
-2),
strMatFileInputPrefix = poped.choose(popedInput$parallelSettings$strMatFileInputPrefix,
  "parallel_input"),
Mat_Out_Pre = poped.choose(popedInput$parallelSettings$Mat_Out_Pre,
  "parallel_output"),
strExtraRunOptions = poped.choose(popedInput$parallelSettings$strExtraRunOptions,
  ""),
dPollResultTime = poped.choose(popedInput$parallelSettings$dPollResultTime,
  0.1),
strFunctionInputName = poped.choose(popedInput$parallelSettings$strFunctionInputName,
  "function_input"),
bParallelRS = poped.choose(popedInput$parallelSettings$bParallelRS, FALSE),
```r
bParallelSG = poped.choose(popedInput$parallelSettings$bParallelSG, FALSE),
bParallelMFEA = poped.choose(popedInput$parallelSettings$bParallelMFEA, FALSE), bParallelLS = poped.choose(popedInput$parallelSettings$bParallelLS, FALSE))
```

**Arguments**

- `popedInput` An input file to PopED. List elements should match the values seen in the Usage section (the defaults to function arguments). Can also be an empty list `list()`.  
- `ff_file` **• START OF MODEL DEFINITION OPTIONS*******
  A string giving the function name or filename and path of the structural model. The filename and the function name must be the same if giving a filename. e.g. "ff.PK.1.comp.oral.md.KE"
- `ff_fun` Function describing the structural model. e.g. `ff.PK.1.comp.oral.md.KE`.
- `fg_file` A string giving the function name or filename and path of the parameter model. The filename and the function name must be the same if giving a filename. e.g. "parameter.model"
- `fg_fun` Function describing the parameter model. e.g. `parameter.model`
- `fError_file` A string giving the function name or filename and path of the residual error model. The filename and the function name must be the same if giving a filename. e.g. "feps.prop".
- `fError_fun` Function describing the residual error model. e.g. `feps.prop`.
- `optsw` **• WHAT TO OPTIMIZE*******
  Row vector of optimization tasks (1=TRUE,0=FALSE) in the following order: (Samples per subject, Sampling schedule, Discrete design variable, Continuous design variable, Number of id per group). All elements set to zero => only calculate the FIM with current design.
- `xt` **• START OF INITIAL DESIGN OPTIONS*******
  Matrix defining the initial sampling schedule. Each row is a group/individual. If only one vector is supplied, e.g. `c(1,2,3,4)`, then all groups will have the same initial design.
- `m` Number of groups in the study. Each individual in a group will have the same design.
- `x` A matrix defining the initial discrete values for the model. Each row is a group/individual.
- `nx` Number of discrete design variables.
- `a` Matrix defining the initial continuous covariate values. n_rows=number of groups, n_cols=number of covariates. If the number of rows is one and the number of groups > 1 then all groups are assigned the same values.
- `na` The number of covariates in the model.
- `groupsize` Vector defining the size of the different groups (num individuals in each group). If only one numer then the number will be the same in every group.
- `ni` Vector defining the number of samples for each group.
- `model_switch` Matrix defining which response a certain sampling time belongs to.
create.poped.database

• ******START OF DESIGN SPACE OPTIONS**********

maxni
Max number of samples per group/individual

minni
Min number of samples per group/individual

maxtotni
Number defining the maximum number of samples allowed in the experiment.

mintotni
Number defining the minimum number of samples allowed in the experiment.

maxgroupsize
Vector defining the max size of the different groups (max number of individuals in each group)

mingroupsize
Vector defining the min size of the different groups (min num individuals in each group) –

maxtotgroupsize
The total maximal groupsize over all groups

mintotgroupsize
The total minimal groupsize over all groups

maxxt
Matrix or single value defining the maximum value for each xt sample. If a single value is supplied then all xt values are given the same maximum value.

minxt
Matrix or single value defining the minimum value for each xt sample. If a single value is supplied then all xt values are given the same minimum value

discrete_x
Cell array defining the discrete variables for each x value.

maxa
Vector defining the max value for each covariate. IF a single value is supplied then all a values are given the same max value

mina
Vector defining the min value for each covariate. IF a single value is supplied then all a values are given the same min value

bUseGrouped_xt
Use grouped time points (1=TRUE, 0=FALSE).

G_xt
Matrix defining the grouping of sample points. Matching integers mean that the points are matched.

bUseGrouped_a
Use grouped covariates (1=TRUE, 0=FALSE)

G_a
Matrix defining the grouping of covariates. Matching integers mean that the points are matched.

bUseGrouped_x
Use grouped discrete design variables (1=TRUE, 0=FALSE).

G_x
Matrix defining the grouping of discrete design variables. Matching integers mean that the points are matched.

fIMCalculationType
• ******START OF FIM CALCULATION OPTIONS**********

Fisher Information Matrix type
• 0=Full FIM
• 1=Reduced FIM
• 2=weighted models
• 3=Loc models
• 4=reduced FIM with derivative of SD of sigma as in PFIM
• 5=FULL FIM parameterized with A,B,C matrices & derivative of variance
• 6=Calculate one model switch at a time, good for large matrices
- 7=Reduced FIM parameterized with A,B,C matrices & derivative of variance

**iApproximationMethod**
Approximation method for model, 0=FO, 1=FOCE, 2=FOCEI, 3=FOI

**iFOCENumInd**
Num individuals in each step of FOCE

**prior_fim**
The prior FIM (added to calculated FIM)

**strAutoCorrelationFile**
Filename and path, or function name, for the Autocorrelation function, empty string means no autocorrelation.

**d_switch**
- ******START OF CRITERION SPECIFICATION OPTIONS************

D-family design (1) or ED-familty design (0) (with or without parameter uncertainty)

**ofv_calc_type**
- 1 = "D-optimality". Determinant of the FIM: det(FIM)
- 2 = "A-optimality". Inverse of the sum of the expected parameter variances: 1/trace_matrix(inv(FIM))
- 4 = "lnD-optimality". Natural logarithm of the determinant of the FIM: log(det(FIM))
- 6 = "Ds-optimality". Ratio of the Determinant of the FIM and the Determinant of the uninteresting rows and columns of the FIM: det(FIM)/det(FIM_u)
- 7 = Inverse of the sum of the expected parameter RSE: 1/sum(get_rse(FIM,poped.db,use_percent=FALSE))

**ds_index**
Ds_index is a vector set to 1 if a parameter is uninteresting, otherwise 0. size=(1,num unfixed parameters). First unfixed bpop, then unfixed d, then unfixed docc and last unfixed sigma. Default is the fixed effects being important, everything else not important. Used in conjunction with ofv_calc_type=6.

**strEDPenaltyFile**
Penalty function name or path and filename, empty string means no penalty. User defined criterion can be defined this way.

**iEDCalculationType**
- ******START OF E-FAMILY CRITERION SPECIFICATION OPTIONS************

ED Integral Calculation, 0=Monte-Carlo-Integration, 1=Laplace Approximation, 2=BFGS Laplace Approximation – –

**ED_samp_size**
Sample size for E-family sampling

**bLHS**
How to sample from distributions in E-family calculations. 0=Random Sampling, 1=LatinHyperCube –

**strUserDistributionFile**
Filename and path, or function name, for user defined distributions for E-family designs

**nbpop**
- ******START OF Model parameters SPECIFICATION OPTIONS************

Number of typical values

**NumRanEff**
Number of IIV parameters. Typically can be computed from other values and not supplied.
<table>
<thead>
<tr>
<th>Variable</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>NumDocc</td>
<td>Number of IOV variance parameters. Typically can be computed from other</td>
</tr>
<tr>
<td></td>
<td>values and not supplied.</td>
</tr>
<tr>
<td>NumOcc</td>
<td>Number of occasions. Typically can be computed from other values and not</td>
</tr>
<tr>
<td></td>
<td>supplied.</td>
</tr>
<tr>
<td>ng</td>
<td>The length of the g parameter vector. Typically can be computed from other</td>
</tr>
<tr>
<td></td>
<td>values and not supplied.</td>
</tr>
<tr>
<td>bpop</td>
<td>Matrix defining the fixed effects, per row (row number = parameter_number)</td>
</tr>
<tr>
<td></td>
<td>we should have:</td>
</tr>
<tr>
<td></td>
<td>• column 1 the type of the distribution for E-family designs (0 = Fixed, 1</td>
</tr>
<tr>
<td></td>
<td>= Normal, 2 = Uniform, 3 = User Defined Distribution, 4 = lognormal and 5</td>
</tr>
<tr>
<td></td>
<td>= truncated normal)</td>
</tr>
<tr>
<td></td>
<td>• column 2 defines the mean.</td>
</tr>
<tr>
<td></td>
<td>• column 3 defines the variance of the distribution (or length of uniform</td>
</tr>
<tr>
<td></td>
<td>distribution).</td>
</tr>
<tr>
<td></td>
<td>Can also just supply the parameter values as a vector c().</td>
</tr>
<tr>
<td>d</td>
<td>Matrix defining the diagonals of the IIV (same logic as for the fixed effects).</td>
</tr>
<tr>
<td></td>
<td>can also just supply the parameter values as a c().</td>
</tr>
<tr>
<td>covd</td>
<td>Matrix defining the covariances of the IIV variances. Set to zero if not</td>
</tr>
<tr>
<td></td>
<td>defined.</td>
</tr>
<tr>
<td>sigma</td>
<td>Matrix defining the variances can covariances of the residual variability</td>
</tr>
<tr>
<td></td>
<td>terms of the model. can also just supply the diagonal parameter values</td>
</tr>
<tr>
<td></td>
<td>(variances) as a c().</td>
</tr>
<tr>
<td>docc</td>
<td>Matrix defining the IOV, the IOV variances and the IOV distribution</td>
</tr>
<tr>
<td>covdocc</td>
<td>Matrix defining the covariance of the IOV.</td>
</tr>
<tr>
<td>notfixed_bpop</td>
<td>• START OF Model parameters fixed or not SPECIFICATION OPTIONS**********</td>
</tr>
<tr>
<td></td>
<td>Vector defining if a typical value is fixed or not (1=not fixed, 0=fixed)</td>
</tr>
<tr>
<td>notfixed_d</td>
<td>Vector defining if a IIV is fixed or not (1=not fixed, 0=fixed)</td>
</tr>
<tr>
<td>notfixed_covd</td>
<td>Vector defining if a covariance IIV is fixed or not (1=not fixed, 0=fixed)</td>
</tr>
<tr>
<td>notfixed_docc</td>
<td>Vector defining if an IOV variance is fixed or not (1=not fixed, 0=fixed)</td>
</tr>
<tr>
<td>notfixed_covdocc</td>
<td>Vector row major order for lower triangular matrix defining if a covariance</td>
</tr>
<tr>
<td></td>
<td>IOV is fixed or not (1=not fixed, 0=fixed).</td>
</tr>
<tr>
<td>notfixed_sigma</td>
<td>Vector defining if a residual error parameter is fixed or not (1=not fixed,</td>
</tr>
<tr>
<td></td>
<td>0=fixed)</td>
</tr>
<tr>
<td>notfixed_covsigma</td>
<td>Vector defining if a covariance residual error parameter is fixed or not</td>
</tr>
<tr>
<td></td>
<td>(1=not fixed, 0=fixed). Default is fixed.</td>
</tr>
<tr>
<td>bUseRandomSearch</td>
<td>• START OF Optimization algorithm SPECIFICATION OPTIONS**********</td>
</tr>
<tr>
<td></td>
<td>Use random search (1=TRUE, 0=FALSE)</td>
</tr>
<tr>
<td>bUseStochasticGradient</td>
<td>Use Stochastic Gradient search (1=TRUE, 0=FALSE)</td>
</tr>
<tr>
<td>bUseLineSearch</td>
<td>Use Line search (1=TRUE, 0=FALSE)</td>
</tr>
</tbody>
</table>
bUseExchangeAlgorithm
   Use Exchange algorithm (1=TRUE, 0=FALSE)
bUseBFGSMinimizer
   Use BFGS Minimizer (1=TRUE, 0=FALSE)
EACriteria
   Exchange Algorithm Criteria, 1 = Modified, 2 = Fedorov
strRunFile
   Filename and path, or function name, for a run file that is used instead of the
   regular PopED call.
poped_version
   • ********START OF Labeling and file names SPECIFICATION OPTIONS**********
      The current PopED version
modtit
   The model title
output_file
   Filename and path of the output file during search
output_function_file
   Filename suffix of the result function file
strIterationFileName
   Filename and path for storage of current optimal design
user_data
   • ********START OF Miscellaneous SPECIFICATION OPTIONS**********
      User defined data structure that, for example could be used to send in data to the
      model
ourzero
   Value to interpret as zero in design
dSeed
   The seed number used for optimization and sampling – integer or -1 which cre-
   ates a random seed
line_opta
   Vector for line search on continuous design variables (1=TRUE,0=FALSE)
line_optx
   Vector for line search on discrete design variables (1=TRUE,0=FALSE)
bShowGraphs
   Use graph output during search
use_logfile
   If a log file should be used (0=FALSE, 1=TRUE)
m1_switch
   Method used to calculate M1 (0=Complex difference, 1=Central difference,
   20=Analytic derivative, 30=Automatic differentiation)
m2_switch
   Method used to calculate M2 (0=Central difference, 1=Central difference, 20=An-
   alytic derivative, 30=Automatic differentiation)
hle_switch
   Method used to calculate linearization of residual error (0=Complex difference,
   1=Central difference, 30=Automatic differentiation)
g3f3_switch
   Method used to calculate the gradient of the model (0=Complex difference,
   1=Central difference, 20=Analytic derivative, 30=Automatic differentiation)
g3fg_switch
   Method used to calculate the gradient of the parameter vector g (0=Complex
differe licence, 1=Central difference, 20=Analytic derivative, 30=Automatic differ-
entiation)
rsit_output
   Number of iterations in random search between screen output
sgit_output
   Number of iterations in stochastic gradient search between screen output
hm1
   Step length of derivative of linearized model w.r.t. typical values
hlf
   Step length of derivative of model w.r.t. g
hlg
   Step length of derivative of g w.r.t. b
hm2 Step length of derivative of variance w.r.t. typical values
hgd Step length of derivative of OFV w.r.t. time
hle Step length of derivative of model w.r.t. sigma
AbsTol The absolute tolerance for the diff equation solver
RelTol The relative tolerance for the diff equation solver
iDiffSolverMethod The diff equation solver method, 0, no other option
bUseMemorySolver If the differential equation results should be stored in memory (1) or not (0)
rsit Number of Random search iterations
sgit Number of stochastic gradient iterations
intrsit Number of Random search iterations with discrete optimization.
intsgit Number of Stochastic Gradient search iterations with discrete optimization
maxrsnullit Iterations until adaptive narrowing in random search
convergence_eps Stochastic Gradient convergence value, (difference in OFV for D-optimal, difference in gradient for ED-optimal)
rslxt Random search locality factor for sample times
rsla Random search locality factor for covariates
cfxt Stochastic Gradient search first step factor for sample times
cfaa Stochastic Gradient search first step factor for covariates
bGreedyGroupOpt Use greedy algorithm for group assignment optimization
EAStepSize Exchange Algorithm StepSize
EANumPoints Exchange Algorithm NumPoints
EAConvergenceCriteria Exchange Algorithm Convergence Limit/Criteria
bEANoReplicates Avoid replicate samples when using Exchange Algorithm
BFGSConvergenceCriteriaMinStep BFGS Minimizer Convergence Criteria Minimum Step
BFGSProjectedGradientTol BFGS Minimizer Convergence Criteria Normalized Projected Gradient Tolerance
BFGSTolerancef BFGS Minimizer Line Search Tolerance f
BFGSTolerancecg BFGS Minimizer Line Search Tolerance g
BFGSTolerancecx BFGS Minimizer Line Search Tolerance x
ED_diff_it Number of iterations in ED-optimal design to calculate convergence criteria
ED_diff_percent ED-optimal design convergence criteria in percent
line_search_it Number of grid points in the line search
Doptim_iter  Number of iterations of full Random search and full Stochastic Gradient if line search is not used

iCompileOption  *****START OF PARALLEL OPTIONS******* Compile options for PopED
  • -1 = No compilation,
  • 0 or 3 = Full compilation,
  • 1 or 4 = Only using MCC (shared lib),
  • 2 or 5 = Only MPI,
  • Option 0,1,2 runs PopED and option 3,4,5 stops after compilation

iUseParallelMethod  Parallel method to use (0 = Matlab PCT, 1 = MPI)

MCC_Dep  Additional dependencies used in MCC compilation (mat-files), if several space separated

strExecuteName  Compilation output executable name

iNumProcesses  Number of processes to use when running in parallel (e.g. 3 = 2 workers, 1 job manager)

iNumChunkDesignEvals  Number of design evaluations that should be evaluated in each process before getting new work from job manager

strMatFileInputPrefix  The prefix of the input mat file to communicate with the executable

Mat_Out_Prefix  The prefix of the output mat file to communicate with the executable

strExtraRunOptions  Extra options send to e$g. the MPI executable or a batch script, see execute_parallel$m for more information and options

dPollResultTime  Polling time to check if the parallel execution is finished

strFunctionInputName  The file containing the popedInput structure that should be used to evaluate the designs

bParallelRS  If the random search is going to be executed in parallel

bParallelSG  If the stochastic gradient search is going to be executed in parallel

bParallelMFEA  If the modified exchange algorithm is going to be executed in parallel

bParallelLS  If the line search is going to be executed in parallel

Value
  A PopED database

See Also
  Other poped_input: convert_variables; create_design_space; create_design; downsizing_general_design; poped.choose
create_design

Examples

## Warfarin example from software comparison in:
## Nyberg et al., "Methods and software tools for design evaluation
## for population pharmacokinetics-pharmacodynamics studies",

library(PopED)

## find the parameters that are needed to define from the structural model
ff.PK.1.comp.oral.md.CL

## -- parameter definition function
## -- names match parameters in function ff
sfg <- function(x,a,bpop,b,bocc){
  parameters=c(CL=bpop[[1]]*exp(b[1]),
              V=bpop[[2]]*exp(b[2]),
              KA=bpop[[3]]*exp(b[3]),
              Favail=bpop[4],
              DOSE=a[1])
  return(parameters)
}

## -- Define initial design and design space
poped.db <- create.poped.database(ff_file="ff.PK.1.comp.oral.sd.CL",
                                  fg_file="sfg",
                                  fError_file="feps.prop",
                                  bpop=c(CL=0.15, V=8, KA=1.0, Favail=1),
                                  notfixed_bpop=c(1,1,1,0),
                                  d=c(CL=0.07, V=0.02, KA=0.6),
                                  sigma=0.01,
                                  groupsize=32,
                                  xt=c(0.5,1,2,6,24,36,72,120),
                                  minxt=0,
                                  maxxt=120,
                                  a=70)

## evaluate initial design
FIM <- evaluate.fim(poped.db)
FIM
  det(FIM)
  get_rse(FIM,poped.db)
create_design

Description

Create design variables to fully describe a design. If variables are supplied then these variables are checked for consistency and, if possible, changed to sizes that make sense if there are inconsistencies. Returns a list of matrices compatible with PopED.

Usage

create_design(xt, groupsize, m = NULL, x = NULL, a = NULL, ni = NULL,
model_switch = NULL)

Arguments

- `xt`: Matrix defining the sampling schedule. Each row is a group.
- `groupsize`: Vector defining the size of the different groups (number of individuals in each group).
- `m`: A number defining the number of groups. Computed from xt if not defined.
- `x`: A matrix defining the discrete design variables for the model. Each row is a group.
- `a`: Matrix defining the continuous design variables. Each row is a group.
- `ni`: Vector defining the number of samples for each group, computed as all elements of xt for each group by default.
- `model_switch`: Matrix defining which response a certain sampling time belongs to. Defaults to one for all elements of xt.

Details

If a value (or a vector/list of values) is supplied that corresponds to only one group and the design has multiple groups then all groups will have the same value(s). If a matrix is expected then a list of lists can be supplied instead, each list corresponding to a group.

See Also

Other poped_input: convert_variables; create.poped.database; create_design_space; downsizing_general_design; poped.choose

Examples

library(PopED)

xt1 <- list(c(1,2,3),c(1,2,3,4))
x4 <- list(c(1,2,3,4,5),c(1,2,3,4))
x2 <- rbind(c(1,2,3,4),c(1,2,3,4))
x3 <- c(1,2,3,4)

design_1 <- create_design(xt=xt1,groupsize=28)
design_2 <- create_design(xt=xt4,groupsize=28)
design_3 <- create_design(xt=xt2,groupsize=28)
design_4 <- create_design(xt=xt3,groupsize=28)
create_design_space

Create design variables and a design space for a full description of an optimization problem.

Description

create_design_space takes an initial design and arguments for a design space and creates a design and design space for design optimization. Checks the sizes of supplied design space variables and changes them to sizes that make sense if there are inconsistencies. Function arguments can use shorthand notation (single values, vectors, lists of vectors and list of list) or matricies. Returns a list of matrices compatible with PopED.

Usage

create_design_space(design, maxni = NULL, minni = NULL, maxtotni = NULL, mintotni = NULL, maxgroupsize = NULL, mingroupsize = NULL, maxtotgroupsize = NULL, mintotgroupsize = NULL, maxxt = NULL, maxt = NULL,
create_design_space

\[
\text{minxt} = \text{NULL}, \quad \text{maxa} = \text{NULL}, \quad \text{mina} = \text{NULL}, \quad \text{x_space} = \text{NULL}, \\
\text{use_grouped_xt} = \text{FALSE}, \quad \text{grouped_xt} = \text{NULL}, \quad \text{use_grouped_a} = \text{FALSE}, \\
\text{grouped_a} = \text{NULL}, \quad \text{use_grouped_x} = \text{FALSE}, \quad \text{grouped_x} = \text{NULL}, \\
\text{our_zero} = \text{NULL}
\]

**Arguments**

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>design</strong></td>
<td>The output from a call to <code>create_design</code>.</td>
</tr>
<tr>
<td><strong>maxni</strong></td>
<td>Vector defining the maximum number of samples per group.</td>
</tr>
<tr>
<td><strong>minni</strong></td>
<td>Vector defining the minimum number of samples per group.</td>
</tr>
<tr>
<td><strong>maxtotni</strong></td>
<td>Number defining the maximum number of samples allowed in the experiment.</td>
</tr>
<tr>
<td><strong>mintotni</strong></td>
<td>Number defining the minimum number of samples allowed in the experiment.</td>
</tr>
<tr>
<td><strong>maxgroupsize</strong></td>
<td>Vector defining the maximum size of the different groups (maximum number of</td>
</tr>
<tr>
<td></td>
<td>individuals in each group)</td>
</tr>
<tr>
<td><strong>mingroupsize</strong></td>
<td>Vector defining the minimum size of the different groups (minimum num indi-</td>
</tr>
<tr>
<td></td>
<td>viduals in each group)</td>
</tr>
<tr>
<td><strong>maxxt</strong></td>
<td>Matrix or single value defining the maximum value for each xt sample. If a</td>
</tr>
<tr>
<td></td>
<td>single value is supplied then all xt values are given the same maximum value.</td>
</tr>
<tr>
<td><strong>minxt</strong></td>
<td>Matrix or single value defining the minimum value for each xt sample. If a</td>
</tr>
<tr>
<td></td>
<td>single value is supplied then all xt values are given the same minimum value</td>
</tr>
<tr>
<td><strong>maxa</strong></td>
<td>Vector defining the maximum value for each covariate. IF a single value is</td>
</tr>
<tr>
<td></td>
<td>supplied then all a values are given the same maximum value</td>
</tr>
<tr>
<td><strong>mina</strong></td>
<td>Vector defining the minimum value for each covariate. IF a single value is</td>
</tr>
<tr>
<td></td>
<td>supplied then all a values are given the same minimum value</td>
</tr>
<tr>
<td><strong>x_space</strong></td>
<td>Cell array <code>cell</code> defining the discrete variables for each x value.</td>
</tr>
<tr>
<td><strong>use_grouped_xt</strong></td>
<td>Group sampling times between groups so that each group has the same values</td>
</tr>
<tr>
<td></td>
<td>(TRUE or FALSE).</td>
</tr>
<tr>
<td><strong>grouped_xt</strong></td>
<td>Matrix defining the grouping of sample points. Matching integers mean that</td>
</tr>
<tr>
<td></td>
<td>the points are matched. Allows for finer control than use_grouped_xt</td>
</tr>
<tr>
<td><strong>use_grouped_a</strong></td>
<td>Group continuous design variables between groups so that each group has</td>
</tr>
<tr>
<td></td>
<td>the same values (TRUE or FALSE).</td>
</tr>
<tr>
<td><strong>grouped_a</strong></td>
<td>Matrix defining the grouping of continuous design variables. Matching intei-</td>
</tr>
<tr>
<td></td>
<td>gers mean that the values are matched. Allows for finer control than use_</td>
</tr>
<tr>
<td></td>
<td>grouped_a.</td>
</tr>
<tr>
<td><strong>use_grouped_x</strong></td>
<td>Group discrete design variables between groups so that each group has the</td>
</tr>
<tr>
<td></td>
<td>same values (TRUE or FALSE).</td>
</tr>
<tr>
<td><strong>grouped_x</strong></td>
<td>Matrix defining the grouping of discrete design variables. Matching integers</td>
</tr>
<tr>
<td></td>
<td>mean that the values are matched. Allows for finer control than use_grouped_x</td>
</tr>
<tr>
<td><strong>our_zero</strong></td>
<td>Value to interpret as zero in design.</td>
</tr>
</tbody>
</table>
Details

If a value (or a vector or a list of values) is supplied that corresponds to only one group and the
design has multiple groups then all groups will have the same value(s). If a matrix is expected then
a list of lists can be supplied instead, each list corresponding to a group.

See Also

Other poped_input: convert_variables; create.poped.database; create_design; downsizing_general_design;
poped.choose

Examples

library(PopED)

design_1 <- create_design(xt=list(c(1,2,3,4,5),
  c(1,2,3,4)),
  groupsize=c(50,20),
  a=list(c(WT=70,DOSE=1000),
  c(DOSE=1000,WT=35))

ds_1 <- create_design_space(design_1)

ds_2 <- create_design_space(design_1,maxni=10,maxxt=10,minxt=0)

ds_3 <- create_design_space(design_1,maxni=10,mingroupsize=20,maxxt=10,minxt=0)

ds_4 <- create_design_space(design_1,maxa=c(100,2000))

ds_5 <- create_design_space(design_1,mina=c(10,20))

design_2 <- create_design(xt=list(c(1,2,3,4,5),
  c(1,2,3,4)),
  groupsize=c(50,20),
  a=list(c(WT=70,DOSE=1000),
  c(WT=35,DOSE=1000)),
  x=list(c(SEX=1,DOSE_discrete=100),
  c(SEX=2,DOSE_discrete=200)))

ds_6 <- create_design_space(design_2)

ds_7 <- create_design_space(design_2,
  x_space=list(SEX=c(1,2),
  DOSE_discrete=seq(100,400,by=20)))

ds_8 <- create_design_space(design_2,
  x_space=list(SEX=c(1,2),
  DOSE_discrete=seq(100,400,by=20)),
  grouped_xt=c(1,2,3,4,5))

ds_9 <- create_design_space(design_2,
  x_space=list(SEX=c(1,2),
  DOSE_discrete=seq(100,400,by=20)),
  grouped_xt=c(1,2,3,4,5))
```r
design_3 <- create_design(xt=list(c(1,2,3,4,5),
                          c(1,2,3,4)),
groupsize=c(50,20),
a=list(c(WT=35,DOSE=1000)),
x=list(c(SEX=1,DOSE_discrete=100)))

ds_10 <- create_design_space(design_3,
                          x_space=list(SEX=c(1,2),DOSE_discrete=seq(100,400,by=20)),
                          use_grouped_xt=TRUE)

ds_11 <- create_design_space(design_2,
                          x_space=list(SEX=c(1,2),DOSE_discrete=seq(100,400,by=20)),
                          grouped_a=list(c(1,2),c(3,2)))

ds_12 <- create_design_space(design_3,
                          x_space=list(SEX=c(1,2),DOSE_discrete=seq(100,400,by=20)),
                          use_grouped_x=TRUE)

ds_13 <- create_design_space(design_3,
                          x_space=list(SEX=c(1,2),DOSE_discrete=seq(100,400,by=20)),
                          grouped_x=list(c(1,2),c(3,2)))
```

---

**diag_matlab**  
*Function written to match MATLAB’s diag function*

**Description**

There are some differences between the MATLAB and the R version of diag. Specifically, if a 1xN or a Nx1 matrix is supplied to the R `diag` function then just the first element of this vector is returned. This function tries to match the MATLAB version in handling vectors (matrices with one dimension equal to one), and will return a diagonal matrix in these situations.

**Usage**

```r
diag_matlab(mat)
```

**Arguments**

- **mat**  
  Either a vector to make into a diagonal matrix or a matrix you want to extract the diagonal from.

**Value**

Either a diagonal matrix or the diagonal of a matrix.
Doptim

See Also

Other MATLAB: cell; feval; fileparts; isempty; isfield; ones; randn; randperm; rand; size; tic; toc; zeros

Other matrix_manipulation: test_for_max; test_for_min; test_for_zeros

Examples

diag_matlab(3)
diag_matlab(c(1,2,3))
diag_matlab(cbind(1,2,3))
diag_matlab(rbind(1,2,3))

diag_matlab(matrix(c(1, 2, 3),6,6))

# here is where the R default does something different
diag(cbind(1,2,3))
diag(rbind(1,2,3))

Doptim  \hspace{1cm} D-family optimization function

Description

Optimize the objective function. There are 4 different optimization algorithms used in this function

1. Adaptive random search. See RS_opt.
2. Stochastic gradient.
3. A Broyden Fletcher Goldfarb Shanno (BFGS) method for nonlinear minimization with box constraints.
4. A line search. See a_line_search.

The optimization algorithms run in series, taking as input the output from the previous method. The stopping rule used is to test if the line search algorithm finds a better optimum than its initial value. If so, then the chain of algorithms is run again. If line search is not used then the argument iter_tot defines the number of times the chain of algorithms is run. This function takes information from the PopED database supplied as an argument. The PopED database supplies information about the model, parameters, design and methods to use. Some of the arguments coming from the PopED database can be overwritten; if they are supplied then they are used instead of the arguments from the PopED database.
Usage

Doptim(poped.db, ni, xt, model_switch, x, a, bpopdescr, ddescr, maxxt, minxt, maxa, mina, fmf = 0, dmf = 0, trflag = TRUE, bUseRandomSearch = poped.db$settings$bUseRandomSearch, bUseStochasticGradient = poped.db$settings$bUseStochasticGradient, bUseBFGSMinimizer = poped.db$settings$bUseBFGSMinimizer, bUseLineSearch = poped.db$settings$bUseLineSearch, sgit = poped.db$settings$sgit, ls_step_size = poped.db$settings$ls_step_size, BFGSConvergenceCriteriaMinStep = poped.db$settings$BFGSConvergenceCriteriaMinStep, BFGSProjectedGradientTol = poped.db$settings$BFGSProjectedGradientTol, BFGSTolerancef = poped.db$settings$BFGSTolerancef, BFGSToleranceg = poped.db$settings$BFGSToleranceg, BFGSTolerancex = poped.db$settings$BFGSTolerancex, iter_tot = poped.db$settings$iNumSearchIterationsIfNotLineSearch, iter_max = 10, ...)

Arguments

poped.db  A PopED database.
ni         A vector of the number of samples in each group.
xt         A matrix of sample times. Each row is a vector of sample times for a group.
model_switch A matrix that is the same size as xt, specifying which model each sample belongs to.
x          A matrix for the discrete design variables. Each row is a group.
a          A matrix of covariates. Each row is a group.
bpopdescr  Matrix defining the fixed effects, per row (row number = parameter_number) we should have:
    • column 1 the type of the distribution for E-family designs (0 = Fixed, 1 = Normal, 2 = Uniform, 3 = User Defined Distribution, 4 = lognormal and 5 = truncated normal)
    • column 2 defines the mean.
    • column 3 defines the variance of the distribution (or length of uniform distribution).
ddescr    Matrix defining the diagonals of the IIV (same logic as for the bpopdescr).
maxxt      Matrix or single value defining the maximum value for each xt sample. If a single value is supplied then all xt values are given the same maximum value.
minxt      Matrix or single value defining the minimum value for each xt sample. If a single value is supplied then all xt values are given the same minimum value.
maxa       Vector defining the max value for each covariate. IF a single value is supplied then all a values are given the same max value
mina       Vector defining the min value for each covariate. IF a single value is supplied then all a values are given the same min value
fmf        The initial value of the FIM. If set to zero then it is computed.
dmf The initial OFV. If set to zero then it is computed.
trflag Should the optimization be output to the screen and to a file?

- *****START OF Optimization algorithm SPECIFICATION OPTIONS***********
Use random search (1=TRUE, 0=FALSE)

bUseStochasticGradient Use Stochastic Gradient search (1=TRUE, 0=FALSE)

bUseBFGSMinimizer Use BFGS Minimizer (1=TRUE, 0=FALSE)

bUseLineSearch Use Line search (1=TRUE, 0=FALSE)

sgit Number of stochastic gradient iterations

ls_step_size Number of grid points in the line search

BFGSConvergenceCriteriaMinStep BFGS Minimizer Convergence Criteria Minimum Step

BFGSProjectedGradientTol BFGS Minimizer Convergence Criteria Normalized Projected Gradient Tolerance

BFGSTolerancef BFGS Minimizer Line Search Tolerance f

BFGSToleranceg BFGS Minimizer Line Search Tolerance g

BFGSTolerancex BFGS Minimizer Line Search Tolerance x

iter_tot Number of iterations to use if line search is not used. Must be less than iter_max to be used.

iter_max If line search is used then the algorithm tests if line search (always run at the end of the optimization iteration) changes the design in any way. If not, the algorithm stops. If yes, then a new iteration is run unless iter_max iterations have already been run.

arguments passed to evaluate.fim and ofv.fim.

References


See Also

Other Optimize: LEDoptim; RS_opt_gen; RS_opt; a_line_search; bfgsb_min; calc_autofocus; calc_ofv_and_grad; mfea; poped_optimize
Examples

```r
## Warfarin example from software comparison in:
## Nyberg et al., "Methods and software tools for design evaluation
## for population pharmacokinetics-pharmacodynamics studies",

## Optimization using an additive + proportional residual error to
## avoid sample times at very low concentrations (time 0 or very late samples).
library(PopED)

## find the parameters that are needed to define from the structural model
ff.PK.1.comp.oral.sd.CL

## -- parameter definition function
## -- names match parameters in function ff
sfg <- function(x,a,bpop,b,bocc){
  parameters=c(CL=bpop[1]*exp(b[1]),
              V=bpop[2]*exp(b[2]),
              KA=bpop[3]*exp(b[3]),
              Favail=bpop[4],
              DOSE=a[1])
  return(parameters)
}

## -- Define initial design and design space
popped.db <- create.poped.database(ff_file="ff.PK.1.comp.oral.sd.CL",
                                    fg_file="sfg",
                                    fError_file="feps.add.prop",
                                    bpop=c(CL=0.15, V=8, KA=1.0, Favail=1),
                                    notfixed_bpop=c(1,1,1,0),
                                    d=c(CL=0.07, V=0.02, KA=0.6),
                                    sigma=c(0.01,0.25),
                                    groupsize=32,
                                    xt=c(0.5,1,2,6,24,36,72,120),
                                    minxt=0,
                                    maxxt=120,
                                    a=70,
                                    mina=0,
                                    maxa=100)

# warfarin optimize model

## Not run:

###############
# typically one will use popped_optimize
# This then calls Doptim for continuous optimization problems
###############

# RS+SG+LS optimization of sample times
# optimization with just a few iterations
# only to check that things are working
```
output <- poped_optimize(poped.db, opt_xt=T, 
sfit=5, sgit=5, ls_step_size=5)

# RS+SG+LS optimization of sample times
# (longer run time than above but more likely to reach a maximum)
output <- poped_optimize(poped.db, opt_xt=T)
get_rse(output$fmf, output$poped.db)
plot_model_prediction(output$poped.db)

# Random search (just a few samples here)
rs.output <- poped_optimize(poped.db, opt_xt=1, opt_a=1, rsit=20, 
bUseRandomSearch= 1,
bUseStochasticGradient = 0,
bUseBFGSMinimizer = 0,
bUseLineSearch = 0)

# line search, DOSE and sample time optimization
ls.output <- poped_optimize(poped.db, opt_xt=1, opt_a=1, 
ls_step_size=10)

# Stochastic gradient search, DOSE and sample time optimization
sg.output <- poped_optimize(poped.db, opt_xt=1, opt_a=1, 
sgit=20)

# BFGS search, DOSE and sample time optimization
bfgs.output <- poped_optimize(poped.db, opt_xt=1, opt_a=1, 
bUseRandomSearch= 0,
bUseStochasticGradient = 0,
bUseBFGSMinimizer = 1,
bUseLineSearch = 0)

# If you really want to you can use Doptim directly

dsl <- downsizing_general_design(poped.db)
poped.db$settings$optsw[2] <- 1 # sample time optimization
output <- Doptim(poped.db, dsl$ni, dsl$xt, dsl$model_switch, dsl$x, dsl$a, 
dsl$bpop, dsl$d, dsl$maxxt, dsl$minxt, dsl$maxa, dsl$mina)

## End(Not run)
downsizing_general_design

*Downsize a general design to a specific design*

**Description**

Function takes a design with potentially empty design variables and resizes the design so that a FIM can be calculated using `mftot`.

**Usage**

```
downsizing_general_design(poped.db)
```

**Arguments**

- `poped.db` A PopED database

**Value**

A list containing:

- `ni` A vector of the number of samples in each group.
- `xt` A matrix of sample times. Each row is a vector of sample times for a group.
- `model_switch` A matrix that is the same size as `xt`, specifying which model each sample belongs to.
- `x` A matrix for the discrete design variables. Each row is a group.
- `a` A matrix of covariates. Each row is a group.
- `bpop` A matrix of fixed effect parameter values.

**See Also**

Other `poped_input`: `convert_variables`; `create.poped.database`; `create_design_space`; `create_design`; `poped.choose`

**Examples**

```r
## Warfarin example from software comparison in:
## Nyberg et al., "Methods and software tools for design evaluation
## for population pharmacokinetics-pharmacodynamics studies",

## Optimization using an additive + proportional residual error to
## avoid sample times at very low concentrations (time 0 or very late samples).
library(PopED)

## find the parameters that are needed to define from the structural model
ff.PK.1.comp.oral.sd.CL
```
## Dtrace

**Trace optimization routines**

**Description**

A helper function for writing output to the screen and files when optimizing.
Usage

Dtrace(fn, it, ni, xtopt, xopt, aopt, gxt, ga, dmf, diff, ixt, ia, itvector, 
dmfvector, poped.db, opt_xt = poped.db$settings$optsw[2], 
opt_a = poped.db$settings$optsw[4], opt_x = poped.db$settings$optsw[3], 
opt_samps = poped.db$settings$optsw[1], 
opt_inds = poped.db$settings$optsw[5], rsit = poped.db$settings$rsit, 
convergence_eps = poped.db$settings$convergence_eps)

Arguments

fn    A file to output information to. Can also be the screen if ''.

it    the iteration number.

ni    A vector of the number of samples in each group.

xtopt The matrix defining current best sampling schedule.

xopt  The cell structure defining the current best discrete design variables.

aopt  The matrix defining the current best continuous design variables.

gxt   The matrix defining the current gradient of the xt vector.

ga    The matrix defining the current gradient for the continuous design variables.

dmf   The current OFV.

diff  The difference from the previous iteration.

ixt   If xt Gradient Inversion Occured or not.

ia    If a Gradient Inversion Occured or not.

itvector The iteration vector. Not currently used.

dmfvector The dmf vector. Not currently used.

poped.db A PopED database.

opt_xt Should the sample times be optimized?

opt_a Should the continuous design variables be optimized?

opt_x Should the discrete design variables be optimized?

opt_samps Are the nuber of sample times per group being optimized?

opt_inds Are the nuber of individuals per group being optimized?

rsit   Number of Random search iterations

convergence_eps Stochastic Gradient convergence value, (difference in OFV for D-optimal, dif-

Examples

## Warfarin example from software comparison in:
## Nyberg et al., "Methods and software tools for design evaluation
## for population pharmacokinetics-pharmacodynamics studies",

## Optimization using an additive + proportional reidual error to
## avoid sample times at very low concentrations (time 0 or very late samples).
library(PopED)

## find the parameters that are needed to define from the structural model
ff.PK.1.comp.oral.sd.CL

## -- parameter definition function
## -- names match parameters in function ff
sfg <- function(x,a,bpop,b,bocc){
  parameters=c(CL=bpop[1]*exp(b[1]),
    V=bpop[2]*exp(b[2]),
    KA=bpop[3]*exp(b[3]),
    Favail=bpop[4],
    DOSE=a[1])
  return(parameters)
}

## -- Define initial design and design space
poped.db <- create.poped.database(ff_file="ff.PK.1.comp.oral.sd.CL",
  fg_file="sfg",
  fError_file="feps.add.prop",
  bpop=c(CL=0.15, V=8, KA=1.0, Favail=1),
  notfixed_bpop=c(1,1,1,0),
  d=c(CL=0.07, V=0.02, KA=0.6),
  sigma=c(0.01,0.25),
  groupsize=32,
  xt=c( 0.5,1,2,6,24,36,72,120),
  minxt=0,
  maxxt=120,
  a=70,
  mina=0,
  maxa=100)

# warfarin optimization model

FIM <- evaluate.fim(poped.db)
dmf <- det(FIM)

Dtrace(fn="",
  it=1,
  ni=poped.db$design$ni,
  xtopt=poped.db$design$xt,
  xopt=poped.db$design$x,
  aopt=poped.db$design$a,
  gxt=0,ga=0,
  dmf=dmf,diff=3,
  ixt=FALSE,
  ia=FALSE,
  itvector=NULL,
  dmfvector=NULL,
  poped.db,
  opt_xt=poped.db$settings$optsw[2],
  opt_a=poped.db$settings$optsw[4],opt_x=poped.db$settings$optsw[3],
opt_samps=poped.db$settings$optsw[1], opt_inds=poped.db$settings$optsw[5], rsit=200)

Dtrace(fn="", it=1,
    ni=poped.db$design$ni,
    xtopt=poped.db$design$xt, xopt=poped.db$design$x, aopt=poped.db$design$a,
    gxt=0, ga=0, dmf=dmf, diff=3,
    ixt=FALSE, ia=FALSE,
    itvector=NULL, dmfvector=NULL,
    poped.db,
    opt_xt=poped.db$settings$optsw[2], opt_a=poped.db$settings$optsw[4], opt_x=poped.db$settings$optsw[3],
    opt_samps=poped.db$settings$optsw[1], opt_inds=poped.db$settings$optsw[5],
    rsit=0)

Dtrace(fn="", it=1,
    ni=poped.db$design$ni,
    xtopt=poped.db$design$xt, xopt=poped.db$design$x, aopt=poped.db$design$a,
    gxt=0, ga=0, dmf=dmf, diff=3,
    ixt=FALSE, ia=FALSE,
    itvector=NULL, dmfvector=NULL,
    poped.db,
    opt_xt=poped.db$settings$optsw[2], opt_a=poped.db$settings$optsw[4], opt_x=poped.db$settings$optsw[3],
    opt_samps=poped.db$settings$optsw[1], opt_inds=poped.db$settings$optsw[5],
    rsit=1)

Dtrace(fn="", it=1,
    ni=poped.db$design$ni,
    xtopt=poped.db$design$xt, xopt=poped.db$design$x, aopt=poped.db$design$a,
    gxt=0, ga=0, dmf=dmf,
    diff=0,
    ixt=FALSE, ia=FALSE,
    itvector=NULL, dmfvector=NULL,
    poped.db,
Evaluate the expectation of determinant the Fisher Information Matrix (FIM) using the Laplace approximation.

Description

Compute the expectation of the \( \det(\text{FIM}) \) using the Laplace approximation to the expectation. Computations are made based on the model, parameters, distributions of parameter uncertainty, design and methods defined in the PopED database or as arguments to the function.

Usage

\[
ed\_laplace\_ofv(model\_switch, \text{groupsize}, n_i, \text{xtopto}, \text{xopto}, \text{aopto}, \text{bpopdescr}, \text{ddescr}, \text{covd}, \text{sigma}, \text{docc}, \text{poped}\_db, \text{method} = 1, \text{return}\_\text{gradient} = \text{FALSE}, \text{optxt} = \text{poped}\_db\$settings\$optsw[2], \text{opta} = \text{poped}\_db\$settings\$optsw[4], x = c())
\]

Arguments

- **model_switch**: A matrix that is the same size as \( \text{xt} \), specifying which model each sample belongs to.
- **groupsize**: A vector of the number of individuals in each group.
- **ni**: A vector of the number of samples in each group.
• column 1 the type of the distribution for E-family designs (0 = Fixed, 1 = Normal, 2 = Uniform, 3 = User Defined Distribution, 4 = lognormal and 5 = truncated normal)
• column 2 defines the mean.
• column 3 defines the variance of the distribution (or length of uniform distribution).

descr Matrix defining the diagonals of the IIV (same logic as for the bpopdescr).
covd Matrix defining the covariances of the IIV variances. Set to zero if not defined.
sigma Matrix defining the variances can covariances of the residual variability terms of the model. can also just supply the diagonal parameter values (variances) as a c().
docc Matrix defining the IOV, the IOV variances and the IOV distribution
poped.db A PopED database.
method If 0 then use an optimization routine translated from poped code written in MATLAB to optimize the parameters in the Laplace approximation. If 1 then use optim to compute both k and the hessian of k (see Dodds et al, JPP, 2005 for more information). If 2 then use fdhess to compute the hessian.
return_gradient Should the gradient be returned.
op txt If sampling times are optimized
op ta If continuous design variables are optimized
x The design parameters to compute the gradient on.

Details

This computation follows the method outlined in Dodds et al, "Robust Population Pharmacokinetic Experiment Design" JPP, 2005, equation 16.

Typically this function will not be run by the user. Instead use evaluate.e.ofv.fim.

Value

The FIM and the hessian of the FIM.

See Also

Other E-family: calc_ofv_and_fim; ed_mftot; evaluate.e.ofv.fim
Other FIM: LinMatrixH; LinMatrixLH; LinMatrixL_occ; calc_ofv_and_fim; ed_mftot; evaluate.e.ofv.fim; evaluate.fim; gradf_eps; mf3; mf5; mf6; mf7; mf8; mftot0; mftot1; mftot2; mftot3; mftot4; mftot5; mftot6; mftot7; mftot; mf; ofv_criterion; ofv_fim
Examples

## Warfarin example from software comparison in:
## Nyberg et al., "Methods and software tools for design evaluation
## for population pharmacokinetics-pharmacodynamics studies",

## Optimization using an additive + proportional reidual error to
## avoid sample times at very low concentrations (time 0 or very late samples).
library(PopED)

## find the parameters that are needed to define from the structural model
ff.PK.1.comp.oral.sd.CL

## -- parameter definition function
## -- names match parameters in function ff
sfg <- function(x,a,bpop,b,bocc){
  parameters=c(CL=bpop[1]*exp(b[1]),
               V=bpop[2]*exp(b[2]),
               KA=bpop[3]*exp(b[3]),
               Favail=bpop[4],
               DOSE=a[1])
  return(parameters)
}

# Normal distribution
bpop_vals <- c(CL=0.15, V=8, KA=1.0, Favail=1)
bpop_vals_ed_n <- cbind(ones(length(bpop_vals),1)*1, # normal distribution
                      bpop_vals,
                      ones(length(bpop_vals),1)*(bpop_vals*0.1)^2) # 10% of bpop value
bpop_vals_ed_n["Favail",] <- c(0,1,0)
bpop_vals_ed_n

## -- Define initial design and design space
poped.db.n <- create.poped.database(ff_file="ff.PK.1.comp.oral.sd.CL",
                                     fg_file="sfg",
                                     fError_file="feps.add.prop",
                                     bpop=bpop_vals_ed_n,
                                     notfixed_bpop=c(1,1,1,0),
                                     d=c(CL=0.07, V=0.02, KA=0.6),
                                     sigma=c(0.01,0.25),
                                     groupsize=32,
                                     xt=c(0.5,1,2,6,24,36,72,120),
                                     minxt=0,
                                     maxxt=120,
                                     a=70,
                                     mina=0,
                                     maxa=100)

## ED evaluate using LaPlace approximation
tic()
output <- evaluate.e.ofv.fim(poped.db.n,use_laplace=TRUE)
toc()
output$E.ofv

## Not run:

## expected value (roughly)
tic()
e.ofv.mc.n <- evaluate.e.ofv.fim(poped.db.n,ED_samp_size=500)
toc()
e.ofv.mc.n$E.ofv

## Using ed_laplace_ofv directly
ed_laplace_ofv(model_switch=poped.db.n$global_model_switch,
    groupsize=poped.db.n$groupsiz,
    ni=poped.db.n$gni,
    xtopto=poped.db.n$xgt,
    xopto=poped.db.n$gx,
    aopto=poped.db.n$ga,
    bpopdescr=poped.db.n$gbpop,
    ddescr=poped.db.n$gd,
    covd=poped.db.n$covd,
    sigma=poped.db.n$sigma,
    docc=poped.db.n$docc,
    poped.db.n)

########################################################################
# Log-normal distribution
########################################################################

# Adding 10% log-normal Uncertainty to fixed effects (not Favail)
bpop_vals <- c(CL=0.15, V=8, KA=1.0, Favail=1)
bpop_vals_ed_ln <- cbind(ones(length(bpop_vals),1)*4, # log-normal distribution
    bpop_vals,
    ones(length(bpop_vals),1)*(bpop_vals*0.1)*2) # 10% of bpop value
bpop_vals_ed_ln[bavail,] <- c(0,1,0)
bpop_vals_ed_ln

## -- Define initial design and design space
poped.db.ln <- create.poped.database(ff_file="ff.PK.1.comp.oral.sd.CL",
    fg_file="sgf",
    fError_file="feps.add.prop",
    bpop=bpop_vals_ed_ln,
    notfixed_bpop=c(1,1,1,0),
    d=c(CL=0.07, V=0.02, KA=0.6),
    sigma=c(0.01,0.25),
    groupsize=32,
    xt=c(0.5,1,2,6,24,36,72,120),
    minxt=0,
Evaluate the expectation of the Fisher Information Matrix (FIM) and the expectation of the OFV (FIM).

Description

Compute the expectation of the FIM given the model, parameters, distributions of parameter uncertainty, design and methods defined in the PopED database.
Usage

ed_mftot(model_switch, groupsize, ni, xtoptn, xoptn, aoptn, bpopdescr, ddescr, 
covd, sigma, docc, poped.db)

Arguments

model_switch A matrix that is the same size as xt, specifying which model each sample belongs to.
groupsize A vector of the number of individuals in each group.
ni A vector of the number of samples in each group.

xtoptn The xtoptn value

xoptn The xoptn

aoptn The aoptn value

bpopdescr Matrix defining the fixed effects, per row (row number = parameter_number) we should have:

• column 1 the type of the distribution for E-family designs (0 = Fixed, 1 = Normal, 2 = Uniform, 3 = User Defined Distribution, 4 = lognormal and 5 = truncated normal)
• column 2 defines the mean.
• column 3 defines the variance of the distribution (or length of uniform distribution).

ddescr Matrix defining the diagonals of the IIV (same logic as for the bpopdescr).
covd Matrix defining the covariances of the IIV variances. Set to zero if not defined.
sigma Matrix defining the variances can covariances of the residual variability terms of the model. can also just supply the diagonal parameter values (variances) as a c().
docc Matrix defining the IOV, the IOV variances and the IOV distribution

poped.db A PopED database.

Value

A list containing the E(FIM) and E(OFV(FIM)) and the a poped.db.

See Also

Other E-family: calc_ofv_and_fim; ed_laplace_ofv; evaluate.e.ofv.fim

Other FIM: LinMatrixH; LinMatrixLH; LinMatrixL_occ; calc_ofv_and_fim; ed_laplace_ofv;
evaluate.e.ofv.fim; evaluate.fim; gradf_eps; mf3; mf5; mf6; mf7; mf8; mftot0; mftot1; mftot2; mftot3; mftot4; mftot5; mftot6; mftot7; mftot; mf; ofv_criterion; ofv_fim
Examples

```r
## Warfarin example from software comparison in:
## Nyberg et al., "Methods and software tools for design evaluation
## for population pharmacokinetics-pharmacodynamics studies",

## Optimization using an additive + proportional residual error to
## avoid sample times at very low concentrations (time 0 or very late samples).
library(PopED)

## find the parameters that are needed to define from the structural model
ff.PK.1.comp.oral.sd.CL

## -- parameter definition function
## -- names match parameters in function ff
sfg <- function(x,a,bpop,b,bocc){
  parameters=c(CL=bpop[1]*exp(b[1]),
    V=bpop[2]*exp(b[2]),
    KA=bpop[3]*exp(b[3]),
    Favail=bpop[4],
    DOSE=a[1])
  return(parameters)
}

# Adding 10% log-normal Uncertainty to fixed effects (not Favail)
bpop_vals <- c(CL=0.15, V=8, KA=1.0, Favail=1)
bpop_vals_ed_ln <- cbind(ones(length(bpop_vals)),1)*4, # log-normal distribution
  bpop_vals,
  ones(length(bpop_vals),1)*(bpop_vals*0.1)^2) # 10% of bpop value
bpop_vals_ed_ln$Favail,] <- c(0,1,0)
bpop_vals_ed_ln

## -- Define initial design and design space
poped.db <- create.poped.database(ff_file="ff.PK.1.comp.oral.sd.CL",
    sg_file="sfg",
  fError_file="feps.add.prop",
  bpop=bpop_vals.ed_ln,
  notfixed_bpop=c(1,1,1,0),
  d=c(CL=0.07, V=0.02, KA=0.6),
  sigma=c(0.01,0.25),
  groupsize=32,
  xt=c(0.5,1,2,6,24,36,72,120),
  minxt=0,
  maxxt=120,
  a=70,
  mina=0,
  maxa=100)

# warfarin ed model

# very few samples
poped.db$settings$EDsamp_size=10
ed_mftot(model_switch=poped.db$design$model_switch,
  
```
evaluate.e.ofv.fim

Evaluate the expectation of the Fisher Information Matrix (FIM) and the expectation of the OFV (FIM).

Description

Compute the expectation of the FIM and OFV (FIM) given the model, parameters, distributions of parameter uncertainty, design and methods defined in the PopED database. Some of the arguments coming from the PopED database can be overwritten; by default these arguments are NULL in the function, if they are supplied then they are used instead of the arguments from the PopED database.

Usage

```r
evaluate.e.ofv.fim(poped.db, fim.calc.type = NULL,
                   bpop = poped.db$parameters$bpop, d = poped.db$parameters$d,
                   covd = poped.db$parameters$covd, docc = poped.db$parameters$docc,
                   sigma = poped.db$parameters$sigma, model_switch = NULL, ni = NULL,
                   xt = NULL, x = NULL, a = NULL, groupsize = poped.db$design$groupsize,
                   deriv.type = NULL, blhs = poped.db$settings$blhs,
                   ofv_calc_type = poped.db$settings$ofv_calc_type,
                   ED_samp_size = poped.db$settings$ED_samp_size,
                   use_laplace = poped.db$settings$edcalculationType, laplace.fim = FALSE, ...
)
```

Arguments

- `poped.db` A PopED database.
- `fim.calc.type` The method used for calculating the FIM. Potential values:
  - 0 = Full FIM. No assumption that fixed and random effects are uncorrelated. See `mftot0`.
  - 1 = Reduced FIM. Assume that there is no correlation in the FIM between the fixed and random effects, and set these elements in the FIM to zero. See `mftot1`.
• 2 = weighted models (placeholder).
• 3 = Not currently used.
• 4 = Reduced FIM and computing all derivatives with respect to the standard deviation of the residual unexplained variation (\sqrt{SIGMA} in NONMEM). This matches what is done in PFIM, and assumes that the standard deviation of the residual unexplained variation is the estimated parameter (NOTE: NONMEM estimates the variance of the residual unexplained variation by default). See \texttt{mftot4}.
• 5 = Full FIM parameterized with A,B,C matrices & derivative of variance. See \texttt{mftot5}.
• 6 = Calculate one model switch at a time, good for large matrices. See \texttt{mftot6}.
• 7 = Reduced FIM parameterized with A,B,C matrices & derivative of variance. See \texttt{mftot7}.

\texttt{bpop}  
Matrix defining the fixed effects, per row (row number = parameter number) we should have:
• column 1 the type of the distribution for E-family designs (0 = Fixed, 1 = Normal, 2 = Uniform, 3 = User Defined Distribution, 4 = lognormal and 5 = truncated normal)
• column 2 defines the mean.
• column 3 defines the variance of the distribution (or length of uniform distribution).
Can also just supply the parameter values as a vector \texttt{c()}

\texttt{d}  
Matrix defining the diagonals of the IIV (same logic as for the fixed effects). can also just supply the parameter values as a \texttt{c()}.

\texttt{covd}  
Matrix defining the covariances of the IIV variances. Set to zero if not defined.

\texttt{docc}  
Matrix defining the IOV, the IOV variances and the IOV distribution

\texttt{sigma}  
Matrix defining the variances can covariances of the residual variability terms of the model. can also just supply the diagonal parameter values (variances) as a \texttt{c()}.

\texttt{model\_switch}  
A matrix that is the same size as \texttt{xt}, specifying which model each sample belongs to.

\texttt{ni}  
A vector of the number of samples in each group.

\texttt{xt}  
A matrix of sample times. Each row is a vector of sample times for a group.

\texttt{x}  
A matrix for the discrete design variables. Each row is a group.

\texttt{a}  
A matrix of covariates. Each row is a group.

\texttt{groupsize}  
A vector of the number of individuals in each group.

\texttt{deriv\_type}  
A number indicating the type of derivative to use:
• 0=Complex difference
• 1=Central difference
• 20=Analytic derivative (placeholder)
• 30=Automatic differentiation (placeholder)
evaluate.e.ofv.fim

bLHS

How to sample from distributions in E-family calculations. 0=Random Sampling, 1=LatinHyperCube –

ofv_calc_type

OFV calculation type for FIM

- 1 = "D-optimality". Determinant of the FIM: det(FIM)
- 2 = "A-optimality". Inverse of the sum of the expected parameter variances: 1/trace_matrix(inv(FIM))
- 4 = "lnD-optimality". Natural logarithm of the determinant of the FIM: log(det(FIM))
- 6 = "Ds-optimality". Ratio of the Determinant of the FIM and the Determinant of the uninteresting rows and columns of the FIM: det(FIM)/det(FIM_u)
- 7 = Inverse of the sum of the expected parameter RSE: 1/sum(get_rse(FIM,poped.db,use_percent=FALSE))

ED_samp_size

Sample size for E-family sampling

use_laplace

Should the Laplace method be used in calculating the expectation of the OFV?

laplace.fim

Should an E(FIM) be calculated when computing the Laplace approximated E(OFV). Typically the FIM does not need to be computed and, if desired, this calculation is done using the standard MC integration technique, so can be slow.

... Other arguments passed to the function.

Value

A list containing the E(FIM) and E(OFV(FIM)) and the a poped.db updated according to the function arguments.

See Also

Other E-family: calc_ofv_and_fim; ed_laplace_ofv; ed_mftot

Other FIM: LinMatrixH; LinMatrixLH; LinMatrixL_occ; calc_ofv_and_fim; ed_laplace_ofv; ed_mftot; evaluate.fim; gradf_eps; mf3; mf5; mf6; mf7; mf8; mftot0; mftot1; mftot2; mftot3; mftot4; mftot5; mftot6; mftot7; mftot; mf; ofv_criterion; ofv_fim

Other evaluate_FIM: calc_ofv_and_fim; evaluate.fim; ofv_fim

Examples

## Warfarin example from software comparison in:
## Nyberg et al., "Methods and software tools for design evaluation
## for population pharmacokinetics-pharmacodynamics studies",

## Optimization using an additive + proportional residual error to
## avoid sample times at very low concentrations (time 0 or very late samples).
library(PopED)

## find the parameters that are needed to define from the structural model
ff.PK.1.comp.oral.sd.CL

## -- parameter definition function
## -- names match parameters in function ff
evaluate.e.ofv.fim

```r
sfg <- function(x,a,bpop,b.bocc)
{
  parameters=c(CL=bpop[1]*exp(b[1]),
                V=bpop[2]*exp(b[2]),
                KA=bpop[3]*exp(b[3]),
                Favail=bpop[4],
                DOSE=a[1])
  return(parameters)
}

# Adding 10% log-normal Uncertainty to fixed effects (not Favail)
bpop_vals <- c(CL=0.15, V=8, KA=1.0, Favail=1)
bpop_vals_ed_ln <- cbind(ones(length(bpop_vals),1)*4, # log-normal distribution
                        bpop_vals,
                        ones(length(bpop_vals),1)*c(bpop_vals*0.1)^2) # 10% of bpop value
bpop_vals_ed_ln

## -- Define initial design and design space
poped.db <- create.poped.database(ff_file="ff.PK.1.comp.oral.sd.CL",
                                   fg_file="sfg",
                                   fError_file="feps.add.prop",
                                   bpop=bpop_vals_ed_ln,
                                   notfixed_bpop=c(1,1,1,0),
                                   d=c(CL=0.07, V=0.02, KA=0.6),
                                   sigma=c(0.01,0.25),
                                   groupsize=32,
                                   xt=c(0.5,1,2,6,24,36,72,120),
                                   minxt=0,
                                   maxxt=120,
                                   a=70,
                                   mina=0,
                                   maxa=100)

# warfarin ed model

## ED evaluate (with very few samples)
output <- evaluate.e.ofv.fim(poped.db,ED_samp_size=10)
output$E_ofv

## API evaluate (with very few samples)
output <- evaluate.e.ofv.fim(poped.db,ED_samp_size=10,ofv_calc_type=4)
output$E_ofv

## ED evaluate using Laplace approximation
tic()
output <- evaluate.e.ofv.fim(poped.db,use_laplace=TRUE)
toc()
output$E_ofv

## Not run:

## ED expected value with more precision.
## Compare time and value to Laplace approximation.
## Run a couple of times to see stochasticity of calculation.
```
evaluate.fim

Evaluate the Fisher Information Matrix (FIM)

Description

Compute the FIM given the model, parameters, design and methods defined in the PopED database. Some of the arguments coming from the PopED database can be overwritten; by default these arguments are NULL in the function, if they are supplied then they are used instead of the arguments from the PopED database.

Usage

```r
evaluate.fim(poped.db, fim.calc.type = NULL, bpop.val = NULL, d_full = NULL, docc_full = NULL, sigma_full = NULL, model_switch = NULL, ni = NULL, xt = NULL, x = NULL, a = NULL, groupsize = NULL, deriv.type = NULL, ...)
```

Arguments

- **poped.db** A PopED database.
- **fim.calc.type** The method used for calculating the FIM. Potential values:
  - 0 = Full FIM. No assumption that fixed and random effects are uncorrelated. See `mft0`.
  - 1 = Reduced FIM. Assume that there is no correlation in the FIM between the fixed and random effects, and set these elements in the FIM to zero. See `mft1`.
  - 2 = weighted models (placeholder).
  - 3 = Not currently used.
  - 4 = Reduced FIM and computing all derivatives with respect to the standard deviation of the residual unexplained variation (sqrt(SIGMA) in NONMEM). This matches what is done in PFIM, and assumes that the standard
deviation of the residual unexplained variation is the estimated parameter (NOTE: NONMEM estimates the variance of the residual unexplained variation by default). See \texttt{mftot4}.

- \(5\) = Full FIM parameterized with A,B,C matrices & derivative of variance. See \texttt{mftot5}.
- \(6\) = Calculate one model switch at a time, good for large matrices. See \texttt{mftot6}.
- \(7\) = Reduced FIM parameterized with A,B,C matrices & derivative of variance. See \texttt{mftot7}.

\texttt{bpop.val} The fixed effects parameter values. Supplied as a vector.
\texttt{d_full} A between subject variability matrix (OMEGA in NONMEM).
\texttt{docc_full} A between occasion variability matrix.
\texttt{sigma_full} A residual unexplained variability matrix (SIGMA in NONMEM).
\texttt{model_switch} A matrix that is the same size as \texttt{xt}, specifying which model each sample belongs to.
\texttt{ni} A vector of the number of samples in each group.
\texttt{xt} A matrix of sample times. Each row is a vector of sample times for a group.
\texttt{x} A matrix for the discrete design variables. Each row is a group.
\texttt{a} A matrix of covariates. Each row is a group.
\texttt{groupsize} A vector of the number of individuals in each group.
\texttt{deriv.type} A number indicating the type of derivative to use:
  - \(0\)=Complex difference
  - \(1\)=Central difference
  - \(20\)=Analytic derivative (placeholder)
  - \(30\)=Automatic differentiation (placeholder)

\texttt{...} Other arguments passed to the function.

**Value**

The FIM.

**See Also**

Other FIM: \texttt{LinMatrixH}; \texttt{LinMatrixLH}; \texttt{LinMatrixL_occ}; \texttt{calc_ofv_and_fim}; \texttt{ed_laplace_ofv}; \texttt{ed_mftot}; \texttt{evaluate.e_ofv.fim}; \texttt{gradf_eps}; \texttt{mf3}; \texttt{mf5}; \texttt{mf6}; \texttt{mf7}; \texttt{mf8}; \texttt{mftot0}; \texttt{mftot1}; \texttt{mftot2}; \texttt{mftot3}; \texttt{mftot4}; \texttt{mftot5}; \texttt{mftot6}; \texttt{mftot7}; \texttt{mftot}; \texttt{mf}; \texttt{ofv_criterion}; \texttt{ofv_fim}

Other \texttt{evaluate.FIM}: \texttt{calc_ofv_and_fim}; \texttt{evaluate.e_ofv.fim}; \texttt{ofv_fim}

Other \texttt{evaluate.design}: \texttt{get_rse}; \texttt{model_prediction}; \texttt{plot_efficiency_of_windows}; \texttt{plot_model_prediction}
Examples

## Warfarin example from software comparison in:
## Nyberg et al., "Methods and software tools for design evaluation
## for population pharmacokinetics-pharmacodynamics studies",

library(PopED)

## find the parameters that are needed to define from the structural model
ff.PK.1.comp.oral.md.CL

## -- parameter definition function
## -- names match parameters in function ff
sfg <- function(x,a,bpop,b,bocc){
  parameters=c(CL=bpop[1]*exp(b[1]),
              V=bpop[2]*exp(b[2]),
              KA=bpop[3]*exp(b[3]),
              Favail=bpop[4],
              DOSE=a[1])
  return(parameters)
}

## -- Define initial design and design space
poped.db <- create.poped.database(ff_file="ff.PK.1.comp.oral.sd.CL",
                                   fg_file="sfg",
                                   fError_file="feps.prop",
                                   bpop=c(CL=0.15, V=8, KA=1.0, Favail=1),
                                   notfixed_bpop=c(1,1,1,0),
                                   d=c(CL=0.07, V=0.02, KA=0.6),
                                   sigma=0.01,
                                   groupsize=32,
                                   xt=c( 0.5,1,2,6,24,36,72,120),
                                   minxt=0,
                                   maxxt=120,
                                   a=70)

## evaluate initial design with the reduced FIM
FIM.1 <- evaluate.fim(poped.db)
FIM.1
det(FIM.1)
get_rse(FIM.1,poped.db)

## evaluate initial design with the full FIM
FIM.0 <- evaluate.fim(poped.db,fim.calc.type=0)
FIM.0
det(FIM.0)
get_rse(FIM.0,poped.db,fim.calc.type=0)

## evaluate initial design with the reduced FIM
## computing all derivatives with respect to the
## standard deviation of the residual unexplained variation
RUV model: Additive.

Description

This is a residual unexplained variability (RUV) model function that encodes the model described above. The function is suitable for input to the `create.poped.database` function using the `ferror_file` argument.

Usage

```r
feps.add(model_switch, xt, parameters, epsi, poped.db)
```

Arguments

- `model_switch` a vector of values, the same size as `xt`, identifying which model response should be computed for the corresponding `xt` value. Used for multiple response models.
- `xt` a vector of independent variable values (often time).
- `parameters` A named list of parameter values.
- `epsi` A matrix with the same number of rows as the `xt` vector, columns match the numbers defined in this function.
- `poped.db` a poped database. This can be used to extract information that may be needed in the model file.
Value
A list consisting of:

1. `y` the values of the model at the specified points.
2. `poped.db` A (potentially modified) poped database.

See Also
Other RUV_models: `feps.add.prop`, `feps.prop`

Examples

```r
library(PopED)

## find the parameters that are needed to define from the structural model
ff.PK.1.comp.oral.sd.KE

## -- parameter definition function
## -- names match parameters in function ff
sfg <- function(x,a,bpop,b,bocc){
  parameters=c(KE=bpop[1]*exp(b[1]),
              V=bpop[2]*exp(b[2]),
              KA=bpop[3]*exp(b[3]),
              Favail=bpop[4],
              DOSE=a[1])
  return(parameters)
}

## -- Define initial design and design space
poped.db <- create.poped.database(ff_file="ff.PK.1.comp.oral.sd.KE",
                                   fg_file="sfg",
                                   ferror_file="feps.add",
                                   bpop=c(KE=0.15/8, V=8, KA=1.0, Favail=1),
                                   notfixed_bpop=c(1,1,1,0),
                                   d=c(KE=0.07, V=0.02, KA=0.6),
                                   sigma=1,
                                   groupsize=32,
                                   xt=c(0.5,1,2,6,24,36,72,120),
                                   minxt=0,
                                   maxxt=120,
                                   a=70)

## create plot of model without variability
plot_model_prediction(poped.db)

## evaluate initial design
FIM <- evaluate.fim(poped.db)
```
feps.add.prop

FIM
det(FIM)
get_rse(FIM,poped.db)

feps.add.prop  RUV model: Additive and Proportional.

Description

This is a residual unexplained variability (RUV) model function that encodes the model described above. The function is suitable for input to the create.poped.database function using the fError_file argument.

Usage

feps.add.prop(model_switch, xt, parameters, epsi, poped.db)

Arguments

- **model_switch**: a vector of values, the same size as xt, identifying which model response should be computed for the corresponding xt value. Used for multiple response models.
- **xt**: a vector of independent variable values (often time).
- **parameters**: A named list of parameter values.
- **epsi**: A matrix with the same number of rows as the xt vector, columns match the numbers defined in this function.
- **poped.db**: a poped database. This can be used to extract information that may be needed in the model file.

Value

A list consisting of:

1. y the values of the model at the specified points.
2. poped.db A (potentially modified) poped database.

See Also

Other RUV_models: feps.add; feps.prop

Other models: feps.add; feps.prop; ff.PK.1.comp.oral.md.CL; ff.PK.1.comp.oral.md.KE; ff.PK.1.comp.oral.sd.CL; ff.PK.1.comp.oral.sd.KE; ff.PKPD.1.comp.oral.md.CL.imax; ff.PKPD.1.comp.sd.CL.emax
Examples

library(PopED)

## find the parameters that are needed to define in the structural model
ff.PK.1.comp.oral.md.CL

## -- parameter definition function
## -- names match parameters in function ff
sfg <- function(x,a,bpop,b,bocc){
  parameters=c(
    V=bpop[1]*exp(b[1]),
    KA=bpop[2]*exp(b[2]),
    CL=bpop[3]*exp(b[3]),
    Favail=bpop[4],
    DOSE=a[1],
    TAU=a[2])
  return( parameters )
}

## -- Define design and design space
poped.db <- create.poped.database(ff_file="ff.PK.1.comp.oral.md.CL",
  fg_file="sfg",
  fError_file="feps.add.prop",
  groupsize=20,
  m=2,
  sigma=c(0.04,5e-6),
  bpop=c(V=72.8,KA=0.25,CL=3.75,Favail=0.9),
  d=c(V=0.09,KA=0.09,CL=0.25*2),
  notfixed_bpop=c(1,1,1,0),
  notfixed_sigma=c(0,0),
  xt=c(1,2,8,240,245),
  minxt=c(0,0,0,240,240),
  maxxt=c(10,10,10,248,248),
  a=cbind(c(20,40),c(24,24)),
  bUseGrouped_xt=1,
  maxa=c(200,24),
  mina=c(0,24))

## create plot of model without variability
plot_model_prediction(poped.db)

## evaluate initial design
FIM <- evaluate.fim(poped.db)
FIM
det(FIM)
get_rse(FIM,poped.db)
Description

This is a residual unexplained variability (RUV) model function that encodes the model described above. The function is suitable for input to the `create.poped.database` function using the `ferror_file` argument.

Usage

`feps.prop(model_switch, xt, parameters, epsi, poped.db)`

Arguments

- **model_switch**: a vector of values, the same size as `xt`, identifying which model response should be computed for the corresponding `xt` value. Used for multiple response models.
- **xt**: a vector of independent variable values (often time).
- **parameters**: A named list of parameter values.
- **epsi**: A matrix with the same number of rows as the `xt` vector, columns match the numbers defined in this function.
- **poped.db**: a poped database. This can be used to extract information that may be needed in the model file.

Value

A list consisting of:

1. `y`: the values of the model at the specified points.
2. `poped.db`: A (potentially modified) poped database.

See Also

Other RUV_models: `feps.add.prop`; `feps.add`

Other models: `feps.add.prop`; `feps.add`; `ff.PK.1.comp.oral.md.CL`; `ff.PK.1.comp.oral.md.KE`; `ff.PK.1.comp.oral.sd.CL`; `ff.PK.1.comp.oral.sd.KE`; `ff.PKPD.1.comp.oral.md.CL.imax`; `ff.PKPD.1.comp.sd.CL.emax`

Examples

```r
## Warfarin example from software comparison in:
## Nyberg et al., "Methods and software tools for design evaluation
## for population pharmacokinetics-pharmacodynamics studies",

library(PopED)

## find the parameters that are needed to define from the structural model
ff.PK.1.comp.oral.sd.CL

## -- parameter definition function
## -- names match parameters in function ff
sfg <- function(x,a,bpop,b,bocc){
```

```
parameters=\[c(\text{CL}=b\text{pop}[1]\times \exp(b[1])),
V=b\text{pop}[2]\times \exp(b[2]),
KA=b\text{pop}[3]\times \exp(b[3]),
F\text{avail}=b\text{pop}[4],
\text{DOSE}=a[1])

return(parameters)
}

### Define initial design and design space
poped.db <- create.poped.database(ff_file="ff.PK.1.comp.oral.sd.CL",
fg_file="sfg",
fError_file="feps.prop",
bpop=c(\text{CL}=0.15, V=8, KA=1.0, F\text{avail}=1),
notfixed_bpop=c(1,1,1,0),
d=c(\text{CL}=0.07, V=0.02, KA=0.6),
sigma=0.01,
groupsize=32,
x=t(c(0.5,1,2,6,24,36,72,120),
\text{minxt}=0,
\text{maxxt}=120,
a=70)

### create plot of model without variability
plot_model_prediction(poped.db)

### evaluate initial design
FIM <- evaluate.fim(poped.db)
FIM
det(FIM)
get_rse(FIM,poped.db)

---

**feval**

*MATLAB feval function*

**Description**

This is just a wrapper for the *do.call* function to behave like the feval function in MATLAB.

**Usage**

`feval(file.name, ...)`

**Arguments**

- `file.name` A function or a string that is the name of a function.
- `...` Arguments for the function. Multiple arguments separated by a comma.
Value

Output from the defined function.

See Also

Other MATLAB: cell; diag_matlab; fileparts; isempty; isfield; ones; randn; randperm; rand; size; tic; toc; zeros

Examples

feval("sin",pi/2)
See Also

Other models: feps.add.prop; feps.add; feps.prop; ff.PK.1.comp.oral.md.KE; ff.PK.1.comp.oral.sd.CL; ff.PK.1.comp.oral.sd.KE; ff.PKPD.1.comp.oral.md.CL.imax; ff.PKPD.1.comp.oral.sd.CL.emax

Other structural models: ff.PK.1.comp.oral.md.KE; ff.PK.1.comp.oral.sd.CL; ff.PK.1.comp.oral.sd.KE; ff.PKPD.1.comp.oral.md.CL.imax; ff.PKPD.1.comp.oral.sd.CL.emax

Examples

library(PopED)

## find the parameters that are needed to define in the structural model
ff.PK.1.comp.oral.md.CL

## -- parameter definition function
## -- names match parameters in function ff
sfg <- function(x,a,bpop,bocc){
  parameters=c( V=bpop[1]*exp(b[1]),
               KA=bpop[2]*exp(b[2]),
               CL=bpop[3]*exp(b[3]),
               Favail=bpop[4],
               DOSe=a[1],
               TAU=a[2])
  return( parameters )
}

## -- Define design and design space
poped.db <- create.poped.database(ff_file="ff.PK.1.comp.oral.md.CL",
                                   fg_file="sfg",
                                   fError_file="feps.add.prop",
                                   groupsize=20,
                                   m=2,
                                   sigma=c(0.84,5e-6),
                                   bpop=c(V=72.8,KA=0.25,CL=3.75,Favail=0.9),
                                   d=c(V=0.09,KA=0.09,CL=0.25*2),
                                   notfixed_bpop=c(1,1,1,0),
                                   notfixed_sigma=c(0,0),
                                   xt=c( 1,2,8,240,245),
                                   minxt=c(0,0,0,240,240),
                                   maxxt=c(10,10,10,248,248),
                                   a=cbind(c(20,40),c(24,24)),
                                   bUseGrouped_xt=1,
                                   maxa=c(200,24),
                                   mina=c(0,24))

## create plot of model without variability
plot_model_prediction(poped.db)

## evaluate initial design
FIM <- evaluate.fim(poped.db)
FIM
det(FIM)
get_rse(FIM,poped.db)

---

**ff.PK.1.comp.oral.md.KE**

*Structural model: one-compartment, oral absorption, multiple bolus dose, parameterized using KE.*

---

### Description

This is a structural model function that encodes a model that is one-compartment, oral absorption, multiple bolus dose, parameterized using KE. The function is suitable for input to the `create.poped.database` function using the `ff_file` argument.

### Usage

```r
ff.PK.1.comp.oral.md.KE(model_switch, xt, parameters, poped.db)
```

### Arguments

- **model_switch**: a vector of values, the same size as `xt`, identifying which model response should be computed for the corresponding `xt` value. Used for multiple response models.
- **xt**: a vector of independent variable values (often time).
- **parameters**: A named list of parameter values.
- **poped.db**: a poped database. This can be used to extract information that may be needed in the model file.

### Value

A list consisting of:

1. `y` the values of the model at the specified points.
2. `poped.db` A (potentially modified) poped database.

### See Also

Other models: `feps.add.prop`; `feps.add`; `feps.prop`; `ff.PK.1.comp.oral.md.CL`; `ff.PK.1.comp.oral.sd.CL`; `ff.PK.1.comp.oral.sd.KE`; `ff.PKPD.1.comp.oral.md.CL.imax`; `ff.PKPD.1.comp.sd.CL.emax`

Other structural models: `ff.PK.1.comp.oral.md.CL`; `ff.PK.1.comp.oral.sd.CL`; `ff.PK.1.comp.oral.sd.KE`; `ff.PKPD.1.comp.oral.md.CL.imax`; `ff.PKPD.1.comp.sd.CL.emax`
Examples

library(PopED)

## find the parameters that are needed to define in the structural model
ff.PK.1.comp.oral.md.KE

## -- parameter definition function
## -- names match parameters in function ff
sfg <- function(x,a,bpop,bocc){
  ## -- parameter definition function
  parameters=c( V=bpop[1]*exp(b[1]),
               KA=bpop[2]*exp(b[2]),
               KE=bpop[3]*exp(b[3]),
               Favail=bpop[4],
               DOSE=a[1],
               TAU=a[2])
  return( parameters )
}

## -- Define design and design space
poped.db <- create.poped.database(ff_file="ff.PK.1.comp.oral.md.KE",
                                   fg_file="sfg",
                                   fError_file="feps.add.prop",
                                   groupsize=20,
                                   m=2,
                                   sigma=c(0.04,5e-6),
                                   bpop=c(V=72.8,KA=0.25,KE=3.75/72.8,Favail=0.9),
                                   d=c(V=0.09,KA=0.09,KE=0.25*2),
                                   notfixed_bpop=c(1,1,1,0),
                                   notfixed_sigma=c(0,0),
                                   xt=c(1,2,8,240,245),
                                   minxt=c(0,0,0,240,240),
                                   maxxt=c(10,10,10,248,248),
                                   a=cbind(c(20,40),c(24,24)),
                                   bUseGrouped_xt=1,
                                   maxa=c(200,40),
                                   mina=c(0,2))

## create plot of model without variability
plot_model_prediction(poped.db)

## evaluate initial design
FIM <- evaluate.fim(poped.db)
FIM
det(FIM)
get_rse(FIM,poped.db)
ff.PK.1.comp.oral.sd.CL

ff.PK.1.comp.oral.sd.CL

Structural model: one-compartment, oral absorption, single bolus dose, parameterized using CL.

Description

This is a structural model function that encodes a model that is one-compartment, oral absorption, single bolus dose, parameterized using CL. The function is suitable for input to the `create.poped.database` function using the `ff_file` argument.

Usage

```r
ff.PK.1.comp.oral.sd.CL(model_switch, xt, parameters, poped.db)
```

Arguments

- `model_switch`: a vector of values, the same size as `xt`, identifying which model response should be computed for the corresponding `xt` value. Used for multiple response models.
- `xt`: a vector of independent variable values (often time).
- `parameters`: A named list of parameter values.
- `poped.db`: a poped database. This can be used to extract information that may be needed in the model file.

Value

A list consisting of:

1. `y` the values of the model at the specified points.
2. `poped.db` A (potentially modified) poped database.

See Also

Other models: `feps.add.prop`; `feps.add`; `feps.prop`; `ff.PK.1.comp.oral.md.CL`; `ff.PK.1.comp.oral.md.KE`; `ff.PK.1.comp.oral.sd.KE`; `ff.PKP.1.comp.oral.md.CL.imax`; `ff.PKP.1.comp.sd.CL.emax`

Other structural models: `ff.PK.1.comp.oral.md.CL`; `ff.PK.1.comp.oral.md.KE`; `ff.PK.1.comp.oral.sd.KE`; `ff.PKP.1.comp.oral.md.CL.imax`; `ff.PKP.1.comp.sd.CL.emax`

Examples

```r
# Warfarin example from software comparison in:
# Nyberg et al., "Methods and software tools for design evaluation
# for population pharmacokinetics-pharmacodynamics studies",

library(PopED)

# find the parameters that are needed to define from the structural model
```
**ff.PK.1.comp.oral.sd.CL**

```r
## -- parameter definition function
## -- names match parameters in function ff
sfg <- function(x,a,bpop,bocc){
  parameters=c(CL=bpop[1]*exp(b[1]),
    V=bpop[2]*exp(b[2]),
    KA=bpop[3]*exp(b[3]),
    Favail=bpop[4],
    DOSE=a[1])
  return(parameters)
}
```

**ff.PK.1.comp.oral.sd.KE**

*Structural model: one-compartment, oral absorption, single bolus dose, parameterized using KE.*

**Description**

This is a structural model function that encodes a model that is one-compartment, oral absorption, single bolus dose, parameterized using KE. The function is suitable for input to the `create.poped.database` function using the `ff_file` argument.

**Usage**

```r
ff.PK.1.comp.oral.sd.KE(model_switch, xt, parameters, poped.db)
```
Arguments

model_switch a vector of values, the same size as xt, identifying which model response should be computed for the corresponding xt value. Used for multiple response models.

xt a vector of independent variable values (often time).

parameters A named list of parameter values.

poped.db a poped database. This can be used to extract information that may be needed in the model file.

Value

A list consisting of:

1. y the values of the model at the specified points.
2. poped.db A (potentially modified) poped database.

See Also

Other models: feps.add.prop; feps.add; feps.prop; ff.PK.1.comp.oral.md.CL; ff.PK.1.comp.oral.md.KE; ff.PK.1.comp.oral.sd.CL; ff.PKPD.1.comp.oral.md.CL.imax; ff.PKPD.1.comp.sd.CL.emax

Other structural_models: ff.PK.1.comp.oral.md.CL; ff.PK.1.comp.oral.md.KE; ff.PK.1.comp.oral.sd.CL; ff.PKPD.1.comp.oral.md.CL.imax; ff.PKPD.1.comp.sd.CL.emax

Examples

```r
library(PopED)

## find the parameters that are needed to define from the structural model
ff.PK.1.comp.oral.sd.KE

## -- parameter definition function
## -- names match parameters in function ff
sfg <- function(x,a,bpop,bocc){
  parameters=c(KE=bpop[1]*exp(a[1]),
              V=bpop[2]*exp(b[2]),
              KA=bpop[3]*exp(b[3]),
              Favail=bpop[4],
              DOSE=a[1])
  return(parameters)
}

## -- Define initial design and design space
poped.db <- create.poped.database(ff_file="ff.PK.1.comp.oral.sd.KE",
                                   fg_file="sfg",
                                   fError_file="feps.prop",
                                   bpop=c(KE=0.15/8, V=8, KA=1.0, Favail=1),
                                   notfixed_bpop=c(1,1,1,0),
                                   d=c(KE=0.07, V=0.02, KA=0.6),
                                   sigma=0.01,
                                   groupsize=32,
                                   )
```
Structural model: one-compartment, oral absorption, multiple bolus dose, parameterized using CL driving an inhibitory IMAX model with a direct effect.

Description

This is a structural model function that encodes the model described above. The function is suitable for input to the `create.poped.database` function using the `ff_file` argument.

Usage

```r
ff.PKPD.1.comp.oral.md.CL.imax(model_switch, xt, parameters, poped.db)
```

Arguments

- `model_switch`: a vector of values, the same size as `xt`, identifying which model response should be computed for the corresponding `xt` value. Used for multiple response models.
- `xt`: a vector of independent variable values (often time).
- `parameters`: A named list of parameter values.
- `poped.db`: a poped database. This can be used to extract information that may be needed in the model file.

Value

A list consisting of:

1. `y`: the values of the model at the specified points.
2. `poped.db`: A (potentially modified) poped database.
**See Also**

Other models: `feps.add.prop`; `feps.add`; `feps.prop`; `ff.PKPD.1.comp.oral.md.CL.imax`; `ff.PKPD.1.comp.oral.md.CL.imax.KE`; `ff.PKPD.1.comp.oral.sd.CL.imax.KE`; `ff.PKPD.1.comp.oral.sd.CL.imax.KEmax`

Other structural_models: `ff.PKPD.1.comp.oral.md.CL.imax`; `ff.PKPD.1.comp.oral.md.CL.imax.KE`; `ff.PKPD.1.comp.oral.sd.CL.imax.KEmax`

**Examples**

```r
library(PopED)

## find the parameters that are needed to define from the structural model
ff.PKPD.1.comp.oral.md.CL.imax
ff.PKPD.1.comp.oral.md.CL

## -- parameter definition function
## -- names match parameters in function ff
sfg <- function(x,a,bpop,bocc){
  ## -- parameter definition function
  parameters=c(
    V=bpop[1]*exp(b[1]),
    KA=bpop[2]*exp(b[2]),
    CL=bpop[3]*exp(b[3]),
    Favail=bpop[4],
    DOSE=a[1],
    TAU = a[2],
    E0=bpop[5]*exp(b[4]),
    IMAX=bpop[6],
    IC50=bpop[7])
  return( parameters )
}

feps <- function(model_switch,xt,parameters,epsi,poped.db){
  ## -- Residual Error function
  returnArgs <- do.call(poped.db$model$ff_pointer,list(model_switch,xt,parameters,poped.db))
  y <- returnArgs[[1]]
  poped.db <- returnArgs[[2]]

  MS <- model_switch

  pk.dv <- y*(1+epsi[,1])+epsi[,2]
  pd.dv <- y*(1+epsi[,3])+epsi[,4]

  y[MS==1] = pk.dv[MS==1]
  y[MS==2] = pd.dv[MS==2]

  return(list( y= y,poped.db =poped.db ))
}
```
## Structural model: one-compartment, single bolus IV dose, parameterized using CL driving an EMAX model with a direct effect.

**Description**

This is a structural model function that encodes the model described above. The function is suitable for input to the `create.poped.database` function using the `ff_file` argument.

**Usage**

```r
ff.PKPD.1.comp.sd.CL.emax(model_switch, xt, parameters, poped.db)
```
Arguments

- `model_switch` a vector of values, the same size as `xt`, identifying which model response should be computed for the corresponding `xt` value. Used for multiple response models.
- `xt` a vector of independent variable values (often time).
- `parameters` A named list of parameter values.
- `poped.db` a poped database. This can be used to extract information that may be needed in the model file.

Value

A list consisting of:

1. `y` the values of the model at the specified points.
2. `poped.db` A (potentially modified) poped database.

See Also

Other models: `feps.add.prop; feps.add; feps.prop; ff.PK.1.comp.oral.md.CL; ff.PK.1.comp.oral.md.KE; ff.PK.1.comp.oral.sd.CL; ff.PK.1.comp.oral.sd.KE; ff.PKPD.1.comp.oral.md.CL.imax`

Other structural models: `ff.PK.1.comp.oral.md.CL; ff.PK.1.comp.oral.md.KE; ff.PK.1.comp.oral.sd.CL; ff.PK.1.comp.oral.sd.KE; ff.PKPD.1.comp.oral.md.CL.imax`

Examples

```r
library(PopED)

## find the parameters that are needed to define from the structural model
ff.PKPD.1.comp.sd.CL.emax

## -- parameter definition function
## -- names match parameters in function ff
sfg <- function(x,a,bpop,bocc){
  ## -- parameter definition function
  parameters=c(
    CL=bpop[1]*exp(b[1]) ,
    V=bpop[2]*exp(b[2]) ,
    E0=bpop[3]*exp(b[3]) ,
    EMAX=bpop[4]*exp(b[4]) ,
    EC50=bpop[5]*exp(b[5]) ,
    DOSE=a[1]
  )
  return( parameters )
}

feps <- function(model_switch,xt,parameters,epsi,poped.db){
  ## -- Residual Error function
  ## -- Proportional PK + additive PD
  returnArgs <- do.call(poped.db$model$ff_pointer,list(model_switch,xt,parameters,poped.db))
  y <- returnArgs[[1]]
```
```r
poped.db <- returnArgs[[2]]

MS <- model_switch

prop.err <- y*(1+epsi[,1])
add.err <- y*epsi[,2]

y[MS==1] = prop.err[MS==1]
y[MS==2] = add.err[MS==2]

return(list(y = y, poped.db = poped.db ))
```

```r
## -- Define initial design and design space
poped.db <- create.poped.database(ff_file="ff.PKPD.1.comp.sd.CL.emax",
        fError_file="feps",
        fg_file="sfg",
        groupsize=20,
        m=3,
        sigma=diag(c(0.15,0.15)),
        bpop=c(CL=0.5,V=0.2,E0=1,EMAX=1,EC50=1),
        d=c(CL=0.01,V=0.01,E0=0.01,EMAX=0.01,EC50=0.01),
        xt=c(0.33,0.66,0.95,0.1,1,1,2,5),
        model_switch=c(1,1,1,1,2,2,2),
        minxt=0,
        maxxt=5,
        a=rbind(2.75,5,10),
        bUseGrouped_xt=1,
        maxa=10,
        mina=0.1)
```

```r
## create plot of model without variability
plot_model_prediction(poped.db, facet_scales="free")
```

```r
## evaluate initial design
FIM <- evaluate.fim(poped.db)
FIM
det(FIM)
get_rse(FIM,poped.db)
```

---

**fileparts**

**MATLAB fileparts function**

**Description**

Get the various parts of a file with path string.
Usage

`getfulld` creates a full D (between subject variability) matrix given a vector of variances and covariances.

**Usage**

```matlab
getfulld(variance_vector, covariance_vector = NULL)
```

**Arguments**

- `variance_vector`:
  - The vector of the variances.
- `covariance_vector`:
  - A vector of the covariances. Written in row major order for the lower triangular matrix.

**Examples**

```matlab
getfulld([1 2 3], [2 3; 1 4; 0 1])
```

**Description**

Create a full D (between subject variability) matrix given a vector of variances and covariances.
getTruncatedNormal

Generate a random sample from a truncated normal distribution.

Description

Generate a random sample from a truncated normal distribution.

Usage

getTruncatedNormal(mean, variance)

Arguments

mean: the mean of the normal distribution
variance: The variance of the normal distribution

Value

A random sample from the specified truncated normal distribution

Examples

getTruncatedNormal(mean=3, variance=100)
**get_all_params**  
*Extract all model parameters from the PopED database.*

**Description**
Extract all model parameters from the PopED database.

**Usage**
```
get_all_params(poped.db)
```

**Arguments**
- `poped.db`  A PopED database.

**Value**
A list containing:
- `bpop`  A vector of fixed effect parameter values.
- `d`  A vector of between subject variability parameters
- `covd`  A vector of the covariances of the between subject variability parameters. Row major format of the lower triangular portion of the D (OMEGA) matrix
- `docc`  A vector of the between occasion variability (BOV) terms in the model
- `covdocc`  A vector of the covariances between the BOV terms. Row major format of the lower triangular portion of the BOV matrix.
- `sigma`  A vector of the residual unexplained variances (RUV)
- `covsigma`  A vector of the covariances between the RUV terms
- `all`  A vector with all of the above, in the order of this list.

**Examples**
```
## Warfarin example from software comparison in:
## Nyberg et al., "Methods and software tools for design evaluation
## for population pharmacokinetics-pharmacodynamics studies",

library(PopED)

## find the parameters that are needed to define from the structural model
ff.PK.1.comp.oral.sd.CL

## -- parameter definition function
## -- names match parameters in function ff
sfg <- function(x,a,bpop,b,bocc){
  parameters=c(CL=bpop[1]*exp(b[1]),
                V=bpop[2]*exp(b[2]),
                ...)
  return(parameters)
}
```
get_rse

KA=bpop[3]*exp(b[3]),
Favail=bpop[4],
DOSE=a[1])
return(parameters)
}

## -- Define initial design and design space
poped.db <- create.poped.database(ff_file="ff.PK.1.comp.oral.sd.CL",
fg_file="sfg",
fError_file="feps.prop",
bpop=c(CL=0.15, V=8, KA=1.0, Favail=1),
notfixed_bpop=c(1,1,1,0),
d=c(CL=0.07, V=0.02, KA=0.6),
sigma=0.01,
groupsize=32,
xt=c(0.5,1,2,6,24,36,72,120),
minxt=0,
maxxt=120,
a=70)

# warfarin basic model
get_all_params(poped.db)

get_rse

**Compute the expected parameter relative standard errors**

**Description**

This function computes the expected relative standard errors of a model given a design and a previously computed FIM.

**Usage**

get_rse(fmf, poped.db, bpop = poped.db$parameters$bpop[, 2, drop = F],
d = poped.db$parameters$d[, 2, drop = F], docc = poped.db$parameters$docc,
sigma = poped.db$parameters$sigma, use_percent = T,
fim.calc.type = poped.db$settings$iFIMCalculationType)

**Arguments**

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>fmf</td>
<td>The initial value of the FIM. If set to zero then it is computed.</td>
</tr>
<tr>
<td>poped.db</td>
<td>A PopED database.</td>
</tr>
</tbody>
</table>
| bpop        | Matrix defining the fixed effects, per row (row number = parameter_number) we should have:

- column 1 the type of the distribution for E-family designs (0 = Fixed, 1 = Normal, 2 = Uniform, 3 = User Defined Distribution, 4 = lognormal and 5 = truncated normal)
get_rse

- column 2 defines the mean.
- column 3 defines the variance of the distribution (or length of uniform distribution).

Can also just supply the parameter values as a vector c() d

Matrix defining the diagonals of the IIV (same logic as for the fixed effects).

can also just supply the parameter values as a c() docc

Matrix defining the IOV, the IOV variances and the IOV distribution

sigma

Matrix defining the variances can covariances of the residual variability terms of the model. can also just supply the diagonal parameter values (variances) as a c() use_percent

Should RSE be reported as percent or not?

fim.calc.type

The method used for calculating the FIM. Potential values:

- 0 = Full FIM. No assumption that fixed and random effects are uncorrelated.
  See mftot0.
- 1 = Reduced FIM. Assume that there is no correlation in the FIM between
  the fixed and random effects, and set these elements in the FIM to zero. See
  mftot1.
- 2 = weighted models (placeholder).
- 3 = Not currently used.
- 4 = Reduced FIM and computing all derivatives with respect to the standard
  deviation of the residual unexplained variation (sqrt(SIGMA) in NONMEM).
  This matches what is done in PFIM, and assumes that the standard
  deviation of the residual unexplained variation is the estimated parameter
  (NOTE: NONMEM estimates the variance of the residual unexplained
  variation by default). See mftot4.
- 5 = Full FIM parameterized with A,B,C matrices & derivative of variance.
  See mftot5.
- 6 = Calculate one model switch at a time, good for large matrices. See
  mftot6.
- 7 = Reduced FIM parameterized with A,B,C matrices & derivative of vari-
  ance See mftot7.

Value

A named list of RSE values.

See Also

Other evaluate_design: evaluate.fim; model_prediction; plot_efficiency_of_windows; plot_model_prediction

Examples

## Warfarin example from software comparison in:
## Nyberg et al., "Methods and software tools for design evaluation
## for population pharmacokinetics-pharmacodynamics studies",
library(PopED)

## find the parameters that are needed to define from the structural model
ff.PK.1.comp.oral.md.CL

## -- parameter definition function
## -- names match parameters in function ff
sfg <- function(x,a,bpop,b,bocc){
  parameters=c(CL=bpop[1]*exp(b[1]),
            V=bpop[2]*exp(b[2]),
            KA=bpop[3]*exp(b[3]),
            Favail=bpop[4],
            DOSE=a[1])
  return(parameters)
}

## -- Define initial design and design space
poped.db <- create.poped.database(ff_file="ff.PK.1.comp.oral.sd.CL",
                                    fg_file="sfg",
                                    ferror_file="feps.prop",
                                    bpop=c(CL=0.15, V=8, KA=1.0, Favail=1),
                                    notfixed_bpop=c(1,1,1,0),
                                    d=c(CL=0.07, V=0.02, KA=0.6),
                                    sigma=0.01,
                                    groupsize=32,
                                    xt=c(0.5,1,2,6,24,36,72,120),
                                    minxt=0,
                                    maxxt=120,
                                    a=70)

## evaluate initial design with the reduced FIM
FIM.1 <- evaluate.fim(poped.db)
FIM.1
det(FIM.1)
get_rse(FIM.1,poped.db)

## evaluate initial design with the full FIM
FIM.0 <- evaluate.fim(poped.db,fim.calc.type=0)
FIM.0
det(FIM.0)
get_rse(FIM.0,poped.db,fim.calc.type=0)

## evaluate initial design with the reduced FIM
## computing all derivatives with respect to the
## standard deviation of the residual unexplained variation
FIM.4 <- evaluate.fim(poped.db,fim.calc.type=4)
FIM.4
det(FIM.4)
get_rse(FIM.4,poped.db,fim.calc.type=4)

## evaluate initial design with the full FIM with A,B,C matrices
gradf_eps

## Description

The function performs a linearization of the model with respect to the residual variability. Derivative of model w.r.t. \( \epsilon \) evaluated at \( \epsilon = 0 \) and \( b = b_{\text{ind}} \).

## Usage

```r
gradf_eps(model_switch, xt_ind, x, a, bpop, b_ind, bocc_ind, num_eps, poped.db)
```

## Arguments

- `model_switch`: A matrix that is the same size as `xt`, specifying which model each sample belongs to.
- `xt_ind`: A vector of the individual/group sample times
- `x`: A matrix for the discrete design variables. Each row is a group.
- `a`: A matrix of covariates. Each row is a group.
- `bpop`: The fixed effects parameter values. Supplied as a vector.
- `b_ind`: Vector of individual realization of the BSV terms \( b \)
- `bocc_ind`: Vector of individual realizations of the BOV terms \( b_{\text{occ}} \)
- `num_eps`: The number of \( \epsilon \) in the model.
- `poped.db`: A PopED database.

## Value

A matrix of size (samples per individual x number of epsilons)
See Also

Other FIM: LinMatrixH; LinMatrixLH; LinMatrixL_occ; calc_ofv_and_fim; ed_laplace_ofv; ed_mftot; evaluate.e.ofv_fim; evaluate.fim; mf3; mf5; mf6; mf7; mf8; mftot0; mftot1; mftot2; mftot3; mftot4; mftot5; mftot6; mftot7; mftot; mf; ofvCriterion; ofv_fim

Examples

## Warfarin example from software comparison in:
## Nyberg et al., "Methods and software tools for design evaluation
## for population pharmacokinetics-pharmacodynamics studies",

## Optimization using an additive + proportional reidual error to
## avoid sample times at very low concentrations (time 0 or very late samples).
library(PopED)

## find the parameters that are needed to define from the structural model
ff.PK.1.comp.oral.sd.CL

## -- parameter definition function
## -- names match parameters in function ff
sfg <- function(x,a,bpop,bocc){
  parameters=c(CL=bpop[1]*exp(b[1]),
              V=bpop[2]*exp(b[2]),
              KA=bpop[3]*exp(b[3]),
              Favail=bpop[4],
              DOSE=a[1])
  return(parameters)
}

## -- Define initial design and design space
poped.db <- create.poped.database(ff_file="ff.PK.1.comp.oral.sd.CL",
                                    fg_file="sfg",
                                    ferror_file="feps.add.prop",
                                    bpop=c(CL=8.15, V=8, KA=1.0, Favail=1),
                                    notfixed_bpop=c(1,1,1,0),
                                    d=c(CL=0.07, V=0.02, KA=0.6),
                                    sigma=c(0.01,0.25),
                                    groupsize=32,
                                    xt=c(.05,1,2,6,24,36,72,120),
                                    minxt=0,
                                    maxxt=120,
                                    a=70,
                                    mina=0,
                                    maxa=100)

# warfarin optimization model

# for the FO approximation
ind=1
gradf_eps(model_switch=t(poped.db$design$model_switch[ind,drop=FALSE]),
          xt=ind=t(poped.db$design$xt[ind,drop=FALSE]),
          x=zeros(0,1),
          library(PopED))
isempty Function written to match MATLAB’s isempty function

Description

Function written to match MATLAB’s isempty function

Usage

isempty(...) 

Arguments

... arguments to pass to the function. Typically a matrix.

Value

Logical. True if the passed object has any dimension that is zero.

See Also

Other MATLAB: cell; diag_matlab; feval; fileparts; isfield; ones; randn; randperm; rand; size; tic; toc; zeros

Examples

isempty(zeros(2,3))
isempty(zeros(2,0))
isempty(c(1,2,3))
isfield  

Function written to match MATLAB's isfield function

Description
Check if a list or dataframe has an element with a specific name.

Usage
isfield(obj, sub.obj.str)

Arguments
- obj: A list or dataframe
- sub.obj.str: A string giving the name of the sub-object you want to check for.

Value
Logical. True if the element exists.

See Also
Other MATLAB: cell; diag_matlab; feval; fileparts; isempty; ones; randn; randperm; rand; size; tic; toc; zeros

Examples
```r
foo <- list("fff"=8,"ggg"=9)

isfield(foo,"fff")
isfield(foo,"lll")
```

LEDoptim  

Optimization function for D-family, E-family and Laplace approximated ED designs

Description
Optimize the objective function for D-family, E-family and Laplace approximated ED designs.

Right now there is only one optimization algorithm used in this function

1. Adaptive random search. See RS_opt_gen.

This function takes information from the PopED database supplied as an argument. The PopED database supplies information about the model, parameters, design and methods to use. Some of the arguments coming from the PopED database can be overwritten; if they are supplied then they are used instead of the arguments from the PopED database.
Usage

LEDoptim(poped.db, model_switch = NULL, ni = NULL, xt = NULL, x = NULL,   
a = NULL, bpopdescr = NULL, ddescr = NULL, maxxt = NULL,   
minxt = NULL, maxa = NULL, mina = NULL, ofv_init = 0, fim_init = 0,   
trflag = TRUE, header_flag = TRUE, footer_flag = TRUE,   
opt_xt = poped.db$settings$optsw[2], opt_a = poped.db$settings$optsw[4],   
opt_x = poped.db$settings$optsw[3], out_file = NULL, d_switch = FALSE,   
use_laplace = T, laplace.fim = FALSE,   
use_RS = poped.db$settings$bUseRandomSearch, ...)

Arguments

poped.db A PopED database.
model_switch Matrix defining which response a certain sampling time belongs to.
ni Vector defining the number of samples for each group.
xt Matrix defining the initial sampling schedule. Each row is a group/individual. If only one vector is supplied, e.g. c(1,2,3,4), then all groups will have the same initial design..
x A matrix defining the initial discrete values for the model. Each row is a group/individual.
a Matrix defining the initial continuous covariate values. n_rows=number of groups, n_cols=number of covariates. If the number of rows is one and the number of groups > 1 then all groups are assigned the same values.
bpopdescr Matrix defining the fixed effects, per row (row number = parameter_number) we should have:

- column 1 the type of the distribution for E-family designs (0 = Fixed, 1 = Normal, 2 = Uniform, 3 = User Defined Distribution, 4 = lognormal and 5 = truncated normal)
- column 2 defines the mean.
- column 3 defines the variance of the distribution (or length of uniform distribution).
ddescr Matrix defining the diagonals of the IIV (same logic as for the bpopdescr).
maxxt Matrix or single value defining the maximum value for each xt sample. If a single value is supplied then all xt values are given the same maximum value.
minxt Matrix or single value defining the minimum value for each xt sample. If a single value is supplied then all xt values are given the same minimum value.
maxa Vector defining the max value for each covariate. IF a single value is supplied then all a values are given the same max value
mina Vector defining the min value for each covariate. IF a single value is supplied then all a values are given the same max value
ofv_init The initial OFV. If set to zero then it is computed.
fim_init The initial value of the FIM. If set to zero then it is computed.
trflag Should the optimization be output to the screen and to a file?
**Definition Options**

- **header_flag**: Should the header text be printed out?
- **footer_flag**: Should the footer text be printed out?
- **opt_xt**: Should the sample times be optimized?
- **opt_a**: Should the continuous design variables be optimized?
- **opt_x**: Should the discrete design variables be optimized?
- **out_file**: Which file should the output be directed to? A string, a file handle using `file` or "" will output to the screen.

**Criterion Specification Options**

- **d_switch**: D-family design (1) or ED-family design (0) (with or without parameter uncertainty)
- **use_laplace**: Should the Laplace method be used in calculating the expectation of the OFV?
- **laplace.fim**: Should a FIM be calculated when computing the Laplace approximated E(OFV). Typically the FIM does not need to be computed and, if desired, this calculation is done using the standard MC integration technique, so can be slow.
- **use_RS**: Should the function use a random search algorithm?

**Examples**

### Warfarin example from software comparison in:

```r
# find the parameters that are needed to define from the structural model
ff.PK.1.comp.oral.sd.CL
```

### -- parameter definition function

```r
# -- names match parameters in function ff
sfg <- function(x,a,bpop,b,bocc){
  parameters=c(CL=bpop[1]*exp(b[1]),
              V=bpop[2]*exp(b[2]),
              KA=bpop[3]*exp(b[3]),
              Favail=bpop[4],
              DOSE=a[1])
  return(parameters)
}
```

# Adding 10% log-normal Uncertainty to fixed effects (not Favail)
bpop_vals <- c(CL=0.15, V=8, KA=1.0, Favail=1)
bpop_vals_ed_ln <- cbind(ones(length(bpop_vals),1), cbind(ones(length(bpop_vals),1), bpop_vals))

bpop_vals_ed_ln["Favail",] <- c(0,1,0)

### -- Define initial design and design space

poped.db <- create.poped.database(ff_file="ff.PK.1.comporal.sd.CL",
                                fg_file="sfg",
                                fError_file="feps.add.prop",
                                bpop=bpop_vals_ed_ln,
                                notfixed_bpop=c(1,1,1,0),
                                d=c(CL=0.07, V=0.02, KA=0.6),
                                sigma=c(0.01,0.25),
                                groupsize=32,
                                xt=c(0.5,1,2,6,24,36,72,120),
                                minxt=0,
                                maxxt=120,
                                a=70,
                                mina=0,
                                maxa=100)

# warfain ed model

## Not run:

LEDoptim(poped.db)

LEDoptim(poped.db, opt_xt=T, rsit=10)

LEDoptim(poped.db, opt_xt=T, rsit=10, d_switch=TRUE)

LEDoptim(poped.db, opt_xt=T, rsit=10, laplace.fim=TRUE)

LEDoptim(poped.db, opt_xt=T, rsit=10, use_laplace=FALSE)

## testing header and footer

LEDoptim(poped.db, opt_xt=T, rsit=10, d_switch=TRUE,
             out_file="foobar.txt")

ff <- LEDoptim(poped.db, opt_xt=T, rsit=10, d_switch=TRUE,
               trflag=FALSE)

LEDoptim(poped.db, opt_xt=T, rsit=10, d_switch=TRUE,
             header_flag=FALSE)

LEDoptim(poped.db, opt_xt=T, rsit=10, d_switch=TRUE,
             out_file="")

LEDoptim(poped.db, opt_xt=T, rsit=10, d_switch=TRUE,
             footer_flag=FALSE)

LEDoptim(poped.db, opt_xt=T, rsit=10, d_switch=TRUE,
Model linearization with respect to epsilon.

Description
The function performs a linearization of the model with respect to the residual variability. Derivative of model w.r.t. eps evaluated at eps=0

Usage
LinMatrixH(model_switch, xt_ind, x, a, bpop, b_ind, bocc_ind, poped.db)

Arguments
- model_switch: A matrix that is the same size as xt, specifying which model each sample belongs to.
- xt_ind: A vector of the individual/group sample times
- x: A matrix for the discrete design variables. Each row is a group.
- a: A matrix of covariates. Each row is a group.
- bpop: The fixed effects parameter values. Supplied as a vector.
- b_ind: Vector of individual realization of the BSV terms b
- bocc_ind: Vector of individual realizations of the BOV terms bocc
- poped.db: A PopED database.

Value
A matrix of size (samples per individual x number of epsilons)

See Also
Other FIM: LinMatrixLH; LinMatrixL_occ; calc_ofv_and_fim; ed_laplace_ofv; ed_mftot; evaluate.e.ofv; fim; gradf_eps; mf; mf5; mf6; mf7; mf8; mftot0; mftot1; mftot2; mftot3; mftot4; mftot5; mftot6; mftot7; mftot8; mf; ofv_criterion; ofv_fim
Examples

## Warfarin example from software comparison in:
## Nyberg et al., "Methods and software tools for design evaluation
## for population pharmacokinetics-pharmacodynamics studies",

## Optimization using an additive + proportional residual error to
## avoid sample times at very low concentrations (time 0 or very late samples).
library(PopED)

## find the parameters that are needed to define from the structural model
ff.PK.1.comp.oral.sd.CL

## -- parameter definition function
## -- names match parameters in function ff
sfg <- function(x,a,bpop,b,bocc){
  parameters=c(CL=bpop[1]*exp(b[1]),
               V=bpop[2]*exp(b[2]),
               KA=bpop[3]*exp(b[3]),
               Favail=bpop[4],
               DOSE=a[1])
  return(parameters)
}

## -- Define initial design and design space
poped.db <- create.poped.database(ff_file="ff.PK.1.comp.oral.sd.CL",
                                   fg_file="sfg",
                                   fError_file="feps.add.prop",
                                   bpop=c(CL=0.15, V=8, KA=1.0, Favail=1),
                                   notfixed_bpop=c(1,1,1,0),
                                   d=c(CL=0.07, V=0.02, KA=0.6),
                                   sigma=c(0.01,0.25),
                                   groupsize=32,
                                   xt=c( 0.5,1,2,6,24,36,72,120),
                                   minxt=0,
                                   maxxt=120,
                                   a=70,
                                   mina=0,
                                   maxa=100)

# warfarin optimization model

# for the FO approximation
ind=1
LinMatrixH(model_switch=t(poped.db$design$model_switch[ind,,drop=FALSE]),
           xt_ind=t(poped.db$design$xt[ind,,drop=FALSE]),
           x=zeros(0,1),
           a=t(poped.db$design$a[ind,,drop=FALSE]),
           bpop=poped.db$parameters$bpop[,2,drop=FALSE],
           b_ind=zeros(poped.db$parameters$NumRanEff,1),
           bocc_ind=zeros(poped.db$parameters$NumDocc,1),
           poped.db)[["y"]]


LinMatrixL

The linearized matrix L

Description

Function computes the derivative of the model with respect to the between subject variability terms in the model (b’s and bocc’s) evaluated at a defined point (b_ind and bocc_ind).

Usage

LinMatrixL(model_switch, xt_ind, x, a, bpop, b_ind, bocc_ind, poped.db)

Arguments

model_switch  A vector that is the same size as xt, specifying which model each sample belongs to.
xt_ind        A vector of sample times.
x             A vector for the discrete design variables.
a             A vector of covariates.
bpop          The fixed effects parameter values. Supplied as a vector.
b_ind        The point at which to evaluate the derivative
bocc_ind      The point at which to evaluate the derivative
poped.db      A PopED database.

Value

As a list:

y             A matrix of size (samples per individual x number of random effects)
poped.db      A PopED database

Examples

## Warfarin example from software comparison in:
## Nyberg et al., "Methods and software tools for design evaluation
## for population pharmacokinetics-pharmacodynamics studies",

## Optimization using an additive + proportional reidual error to
## avoid sample times at very low concentrations (time 0 or very late samples).
library(PopED)

## find the parameters that are needed to define from the structural model
ff.PK.1.comp.oral.sd.CL
## CC MM parameter definition function

```
sfg <- function(x,a,bpop,bocc){
  parameters=c(CL=bpop[1]*exp(b[1]),
              V=bpop[2]*exp(b[2]),
              KA=bpop[3]*exp(b[3]),
              Favail=bpop[4],
              DOSE=a[1])
  return(parameters)
}
```

## CC MM define initial design and design space

```
poped.db <- create.poped.database(ff_file="ff.PK.1.comp.oral.sd.CL",
                                  fg_file="sfg",
                                  fError_file="feps.add.prop",
                                  bpop=c(CL=0.15, V=8, KA=1.0, Favail=1),
                                  notfixed_bpop=c(1,1,1,0),
                                  d=c(CL=0.07, V=0.02, KA=0.6),
                                  sigma=c(0.01,0.25),
                                  groupsize=32,
                                  xt=c(0.5,1,2,6,24,36,72,120),
                                  minxt=0,
                                  maxxt=120,
                                  a=70,
                                  mina=0,
                                  maxa=100)
```

# warfarin optimization model

```
# for the FO approximation
ind=1
LinMatrixLH(model_switch=t(poped.db$design$model_switch[ind,,drop=FALSE]),
            xt_ind=t(poped.db$design$xt[ind,,drop=FALSE]),
            x=zeros(0,1),
            a=t(poped.db$design$a[ind,,drop=FALSE]),
            bpop=poped.db$parameters$bpop[,2,drop=FALSE],
            b_ind=zeros(poped.db$parameters$NumRanEff,1),
            bocc_ind=zeros(poped.db$parameters$NumDocc,1),
            poped.db)["y"]
```

---

**Model linearization with respect to epsilon and eta.**

**Description**

The function performs a linearization of the model with respect to the residual variability and then the between subject variability. Derivative of model w.r.t. eps then eta, evaluated at eps=0 and b=b_ind.
LinMatrixLH

Usage

LinMatrixLH(model_switch, xt_ind, x, a, bpop, b_ind, bocc_ind, NumEPS, poped.db)

Arguments

- **model_switch**: A matrix that is the same size as xt, specifying which model each sample belongs to.
- **xt_ind**: A vector of the individual/group sample times.
- **x**: A matrix for the discrete design variables. Each row is a group.
- **a**: A matrix of covariates. Each row is a group.
- **bpop**: The fixed effects parameter values. Supplied as a vector.
- **b_ind**: Vector of individual realization of the BSV terms b.
- **bocc_ind**: Vector of individual realizations of the BOV terms bocc.
- **NumEPS**: The number of eps() terms in the model.
- **poped.db**: A PopED database.

Value

A matrix of size (samples per individual x (number of sigma x number of omega))

See Also

Other FIM: LinMatrixH; LinMatrixL_occ; calc_ofv_and_fim; ed_laplace_ofv; ed_mftot; evaluate.e.ofv.fim; evaluate.fim; gradf_eps; mf3; mf5; mf6; mf7; mf8; mftot0; mftot1; mftot2; mftot3; mftot4; mftot5; mftot6; mftot7; mftot; mf; ofv_criterion; ofv_fim

Examples

```R
## Warfarin example from software comparison in:
## Nyberg et al., "Methods and software tools for design evaluation
## for population pharmacokinetics-pharmacodynamics studies",

## Optimization using an additive + proportional residual error to
## avoid sample times at very low concentrations (time 0 or very late samples).
library(PopED)

## find the parameters that are needed to define from the structural model
ff.PK.1.comp.oral.sd.CL

## -- parameter definition function
## -- names match parameters in function ff
sfg <- function(x,a,bpop,b,bocc){
  parameters=c(CL=bpop[1]*exp(b[1]),
            V=bpop[2]*exp(b[2]),
            KA=bpop[3]*exp(b[3]),
            Favail=bpop[4],
            DOSE=a[1])
}
Model linearization with respect to occasion variability parameters.

### Description

The function performs a linearization of the model with respect to the occasion variability parameter. Derivative of model w.r.t. eta_occ, evaluated bocc_ind.

### Usage

```r
LinMatrixL_occ(model_switch, xt_ind, x, a, bpop, b_ind, bocc_ind, iCurrentOcc, poped.db)
```
### Arguments

- **model_switch**
  A matrix that is the same size as xt, specifying which model each sample belongs to.

- **xt_ind**
  A vector of the individual/group sample times

- **x**
  A matrix for the discrete design variables. Each row is a group.

- **a**
  A matrix of covariates. Each row is a group.

- **bpop**
  The fixed effects parameter values. Supplied as a vector.

- **b_ind**
  Vector of individual realization of the BSV terms b

- **bocc_ind**
  Vector of individual realizations of the BOV terms bocc

- **iCurrentOcc**
  The current occasion.

- **poped.db**
  A PopED database.

### Value

A matrix of size (samples per individual x number of iovs)

### See Also

Other FIM: LinMatrixH; LinMatrixLH; calc_ofv_and_fim; ed_laplace_ofv; ed_mftot; evaluate.e.ofv.fim; evaluate.fim; gradf_eps; mf3; mf5; mf6; mf7; mf8; mftot0; mftot1; mftot2; mftot3; mftot4; mftot5; mftot6; mftot7; mftot; mf; ofv_criterion; ofv_fim

### Examples

```
## Warfarin example from software comparison in:
## Nyberg et al., "Methods and software tools for design evaluation
## for population pharmacokinetics-pharmacodynamics studies",

## Optimization using an additive + proportional reidual error to
## avoid sample times at very low concentrations (time 0 or very late samples).
library(PopED)

## find the parameters that are needed to define from the structural model
ff.PK.1.comp.oral.sd.CL

## -- parameter definition function
## -- names match parameters in function ff
sfg <- function(x,a,bpop,b,bocc)
{
  parameters=c(CL=bpop[1]*exp(b[1]),
            V=bpop[2]*exp(b[2]),
            KA=bpop[3]*exp(b[3]),
            Favail=bpop[4],
            DOSE=a[1])

  return(parameters)
}

## -- Define initial design and design space
```
log_prior_pdf

Compute the natural log of the PDF for the parameters in an E-family design

Description

Compute the natural log of the PDF for the parameters in an E-family design

Usage

log_prior_pdf(alpha, bpopdescr, ddescr, return_gradient = F,
             return_hessian = F)

Arguments

alpha       A parameter vector.
bpopdescr   Matrix defining the fixed effects, per row (row number = parameter number) we should have:
• column 1 the type of the distribution for E-family designs (0 = Fixed, 1 = Normal, 2 = Uniform, 3 = User Defined Distribution, 4 = lognormal and 5 = truncated normal)
• column 2 defines the mean.
• column 3 defines the variance of the distribution (or length of uniform distribution).

ddescr Matrix defining the diagonals of the IIV (same logic as for the bpopdescr).
return_gradient Should the gradient be returned.
return_hessian Should the hessian be returned?

Examples

## Warfarin example from software comparison in:
## Nyberg et al., "Methods and software tools for design evaluation
## for population pharmacokinetics-pharmacodynamics studies",

## Optimization using an additive + proportional reidual error to
## avoid sample times at very low concentrations (time 0 or very late samples).
library(PopED)

## find the parameters that are needed to define from the structural model
ff.PK.1.comp.orval.sd.CL

## -- parameter definition function
## -- names match parameters in function ff
sfg <- function(x,a,bpop,b,bocc){
  parameters=c(CL=bpop[1]*exp(b[1]),
               V=bpop[2]*exp(b[2]),
               KA=bpop[3]*exp(b[3]),
               Favail=bpop[4],
               DOSE=a[1])
  return(parameters)
}

## -- Define initial design and design space
poped.db <- create.poped.database(ff_file="ff.PK.1.comp.orval.sd.CL",
                                    fg_file="sfg",
                                    fError_file="feps.add.prop",
                                    bpop=c(CL=0.15, V=8, KA=1.0, Favail=1),
                                    notfixed_bpop=c(1,1,1,0),
                                    d=c(CL=0.07, V=0.02, KA=0.6),
                                    sigma=c(0.01,0.25),
                                    groupsize=32,
                                    xt=c(0.5,1,2,6,24,36,72,120),
                                    minxt=0,
                                    maxxt=120,
                                    a=70,
                                    mina=0,
                                    maxa=100)
The full Fisher Information Matrix (FIM) for one individual

Description

Compute the full FIM for one individual given specific model(s), parameters, design and methods. This computation makes no assumption that fixed and random effects are uncorrelated.

Usage

mf(model_switch, xt_ind, x, a, bpop, d, sigma, docc, poped.db)
Arguments

- `model_switch`: A vector that is the same size as `xt`, specifying which model each sample belongs to.
- `xt_ind`: A vector of sample times.
- `x`: A vector for the discrete design variables.
- `a`: A vector of covariates.
- `bpop`: The fixed effects parameter values. Supplied as a vector.
- `d`: A between subject variability matrix (OMEGA in NONMEM).
- `sigma`: A residual unexplained variability matrix (SIGMA in NONMEM).
- `docc`: A between occasion variability matrix.
- `poped.db`: A PopED database.

Value

As a list:

- `ret`: The FIM for one individual
- `poped.db`: A PopED database

See Also

Used by `mftot0`.

Other FIM: `LinMatrixH`; `LinMatrixLH`; `LinMatrixL_occ`; `calc_ofv_and_fim`; `ed_laplace_ofv`; `ed_mftot`; `evaluate.e.ofv.fim`; `evaluate.fim`; `gradf_eps`; `mf3`; `mf5`; `mf6`; `mf7`; `mf8`; `mftot0`; `mftot1`; `mftot2`; `mftot3`; `mftot4`; `mftot5`; `mftot6`; `mftot7`; `mftot`; `ofv_criterion`; `ofv_fim`

Examples

```r
## Warfarin example from software comparison in:
## Nyberg et al., "Methods and software tools for design evaluation
## for population pharmacokinetics-pharmacodynamics studies",

library(PopED)

## find the parameters that are needed to define from the structural model
ff.PK.1.comp.oral.sd.CL

## -- parameter definition function
## -- names match parameters in function ff
sfg <- function(x,a,bpop,b,bocc){
  parameters=c(CL=bpop[1]*exp(b[1]),
              V=bpop[2]*exp(b[2]),
              KA=bpop[3]*exp(b[3]),
              Favil=bpop[4],
              DOSE=a[1])
  return(parameters)
```
## Define initial design and design space

```r
poped.db <- create.poped.database(ff_file="ff.PK.1.comp.oral.sd.CL",
   fg_file="fg",
   fError_file="feps.prop",
   bpop=c(CL=0.15, V=8, KA=1.0, Favail=1),
   notfixed_bpop=c(1,1,1,0),
   d=c(CL=0.07, V=0.02, KA=0.6),
   sigma=0.01,
   groupsize=32,
   xt=c( 0.5,1,2,6,24,36,72,120),
   minxt=0,
   maxxt=120,
   a=70)
```

# warfarin optimization model

# for the FO approximation

```r
ind=1
```

# no occasion defined in this example, so result is zero

```r
output <- mf(model_switch=t(poped.db$design$model_switch[ind,,drop=FALSE]),
   xt_ind=t(poped.db$design$xt[ind,,drop=FALSE]),
   x=zeros(0,1),
   a=t(poped.db$design$a[ind,,drop=FALSE]),
   bpop=poped.db$parameters$bpop[,2,drop=FALSE],
   d=poped.db$parameters$param.pt.val$d,
   sigma=poped.db$parameters$sigma,
   docc=poped.db$parameters$param.pt.val$docc,
   poped.db)
```

# in this simple case the full FIM is just the sum of the individual FIMs
# and all the individual FIMs are the same

```r
det(output$ret*32) == det(evaluate.fim(poped.db,fim.calc.type=0))
```

---

**mf3**

The reduced Fisher Information Matrix (FIM) for one individual

### Description

Compute the reduced FIM for one individual given specific model(s), parameters, design and methods. This computation assumes that there is no correlation in the FIM between the fixed and random effects, and set these elements in the FIM to zero.

### Usage

```r
mf3(model_switch, xt, x, a, bpop, d, sigma, docc, poped.db)
```
Arguments

model_switch A vector that is the same size as xt, specifying which model each sample belongs to.
xt A vector of sample times.
x A vector for the discrete design variables.
a A vector of covariates.
bpop The fixed effects parameter values. Supplied as a vector.
d A between subject variability matrix (OMEGA in NONMEM).
sigma A residual unexplained variability matrix (SIGMA in NONMEM).
docc A between occasion variability matrix.
poped.db A PopED database.

Value

As a list:
ret The FIM for one individual
poped.db A PopED database

See Also

Used by mftot1.
Other FIM: LinMatrixH; LinMatrixLH; LinMatrixL_occ; calc_ofv_and_fim; ed_laplace_ofv; ed_mftot; evaluate.e.ofv.fim; evaluate.fim; gradf_eps; mf5; mf6; mf7; mf8; mftot0; mftot1; mftot2; mftot3; mftot4; mftot5; mftot6; mftot7; mftot8; mftot; mf; ofv_criterion; ofv_fim

Examples

## Warfarin example from software comparison in:
## Nyberg et al., "Methods and software tools for design evaluation
## for population pharmacokinetics-pharmacodynamics studies",

library(PopED)

## find the parameters that are needed to define from the structural model
ff.PK.1.comp.oral.sd.CL

## -- parameter definition function
## -- names match parameters in function ff
sfg <- function(x,a,bpop,b,bocc){
  parameters=c(CL=bpop[1]*exp(b[1]),
              V=bpop[2]*exp(b[2]),
              KA=bpop[3]*exp(b[3]),
              Favil=bpop[4],
              DOSE=a[1])
  return(parameters)
mf5

The reduced Fisher Information Matrix (FIM) for one individual, using the SD of RUV as a parameter.

Description

Compute the reduced FIM for one individual using the standard deviation of the residual unexplained variability (RUV) terms as a parameter, given specific model(s), parameters, design and methods. This computation assumes that there is no correlation in the FIM between the fixed and random effects, and set these elements in the FIM to zero. In addition all derivatives in the computation are made with respect to the standard deviation of the RUV terms (sqrt(SIGMA) in NONMEM). This matches what is done in PFIM, and assumes that the standard deviation of the residual unexplained variation is the estimated parameter (NOTE: NONMEM estimates the variance of the residual unexplained variation by default).
Usage

mf5(model_switch, xt, x, a, bpop, d, sigma, docc, poped.db)

Arguments

model_switch  A vector that is the same size as xt, specifying which model each sample belongs to.
xt  A vector of sample times.
x  A vector for the discrete design variables.
a  A vector of covariates.
bpop  The fixed effects parameter values. Supplied as a vector.
d  A between subject variability matrix (OMEGA in NONMEM).
sigma  A residual unexplained variability matrix (SIGMA in NONMEM).
docc  A between occasion variability matrix.
poped.db  A PopED database.

Value

As a list:
ret  The FIM for one individual
poped.db  A PopED database

See Also

Used by mf5t4.

Other FIM: LinMatrixH; LinMatrixLH; LinMatrixL_occ; calc_ofv_and_fim; ed_laplace_ofv; ed_mftot; evaluate.e_ofv.fim; evaluate.fim; gradf_eps; mf3; mf6; mf7; mf8; mftot0; mftot1; mftot2; mftot3; mftot4; mftot5; mftot6; mftot7; mftot; mf; ofv_criterion; ofv_fim

Examples

## Warfarin example from software comparison in:
## Nyberg et al., "Methods and software tools for design evaluation
## for population pharmacokinetics-pharmacodynamics studies",

library(PopED)

## find the parameters that are needed to define from the structural model
ff.PK.1.comp.oral.sd.CL

## -- parameter definition function
## -- names match parameters in function ff
sfg <- function(x,a,bpop,b,bocc){
  parameters=c(CL=bpop[1]*exp(b[1]),
             V=bpop[2]*exp(b[2]),
             ...)
KA=bpop[3]*exp(b[3]),
Favail=bpop[4],
DOSE=a[1])
return(parameters)
}

## Define initial design and design space
poped.db <- create.poped.database(ff_file="ff.PK.1.comp.oral.sd.CL",
fg_file="sfc",
ferror_file="feps.prop",
bpop=c(CL=0.15, V=8, KA=1.0, Favail=1),
notfixed_bpop=c(1,1,1,0),
d=c(CL=0.07, V=0.02, KA=0.6),
sigma=0.01,
groupsize=32,
xt=c(0.5,1,2,6,24,36,72,120),
minxt=0,
maxxt=120,
a=70)

# warfarin optimization model

# for the FO approximation
ind=1

# no occasion defined in this example, so result is zero
output <- mf5(model_switch=t(poped.db$design$model_switch[ind,,drop=FALSE]),
xt=t(poped.db$design$xt[ind,,drop=FALSE]),
A=zeros(0,1),
a=t(poped.db$design$a[ind,,drop=FALSE]),
bpop=poped.db$parameters$bpop[2,,drop=FALSE],
d=poped.db$parameters$param.pt.val$d,
sigma=poped.db$parameters$sigma,
docc=poped.db$parameters$param.pt.val$docc,
poped.db)

# in this simple case the full FIM is just the sum of the individual FIMs
# and all the individual FIMs are the same
det(output$xt*32) == det(evaluate.fim(poped.db,fim.calc.type=4))

---

**mf6**

*The full Fisher Information Matrix (FIM) for one individual parameterized with A,B,C matrices & using the derivative of variance.*

**Description**

Compute the full FIM for one individual given specific model(s), parameters, design and methods. This computation parameterizes the FIM calculation using A,B,C matrices (as in Retout et al.) but uses the derivative of variances. Should give the same answer as `mf` but computation times may be different.
Usage

mf6(model_switch, xt_ind, x, a, bpop, d, sigma, docc, poped.db)

Arguments

model_switch A vector that is the same size as xt, specifying which model each sample belongs to.
xt_ind A vector of sample times.
x A vector for the discrete design variables.
a A vector of covariates.
bpop The fixed effects parameter values. Supplied as a vector.
d A between subject variability matrix (OMEGA in NONMEM).
sigma A residual unexplained variability matrix (SIGMA in NONMEM).
docc A between occasion variability matrix.
poped.db A PopED database.

Value

As a list:

ret The FIM for one individual
poped.db A PopED database

References


See Also

Used by mftot5.
Other FIM: LinMatrixH; LinMatrixLH; LinMatrixL_occ; calc_ofv_and_fim; ed_laplace_ofv; ed_mftot; evaluate.e.ofv.fim; evaluate.fim; gradf_eps; mf3; mf5; mf7; mf8; mftot0; mftot1; mftot2; mftot3; mftot4; mftot5; mftot6; mftot7; mftot8; mf; ofv.criterion; ofv_fim

Examples

## Warfarin example from software comparison in:
## Nyberg et al., "Methods and software tools for design evaluation
## for population pharmacokinetics-pharmacodynamics studies",

library(PopED)

## find the parameters that are needed to define from the structural model
ff.PK.1.comp.oral.sd.CL
## -- parameter definition function
## -- names match parameters in function ff
sfg <- function(x,a,bpop,bocc){
  parameters=c(CL=bpop[1]*exp(b[1]),
        V=bpop[2]*exp(b[2]),
        KA=bpop[3]*exp(b[3]),
        Favail=bpop[4],
        DOSE=a[1])
  return(parameters)
}

## -- Define initial design and design space
poped.db <- create.poped.database(ff_file="ff.PK.1.comp.oral.sd.CL",
  fg_file="sfg",
  fError_file="feps.prop",
  bpop=c(CL=0.15, V=8, KA=1.0, Favail=1),
  notfixed_bpop=c(1,1,1,0),
  d=c(CL=0.07, V=0.02, KA=0.6),
  sigma=0.01,
  groupsize=32,
  xt=c( 0.5,1,2,6,24,36,72,120),
  minxt=0,
  maxxt=120,
  a=70)

# warfarin optimization model
# for the FO approximation
ind=1

# no occasion defined in this example, so result is zero
output <- mf6(model_switch=t(poped.db$design$model_switch[ind,,drop=FALSE]),
  xt=t(poped.db$design$xt[ind,,drop=FALSE]),
  x=zeros(0,1),
  a=t(poped.db$design$a[ind,,drop=FALSE]),
  bpop=poped.db$parameters$bpop[,2,drop=FALSE],
  d=poped.db$parameters$param.pt.val$d,
  sigma=poped.db$parameters$sigma,
  docc=poped.db$parameters$param.pt.val$docc,
  poped.db)

# in this simple case the full FIM is just the sum of the individual FIMs
# and all the individual FIMs are the same
det(output$ret*32) == det(evaluate.fim(poped.db,fim.calc.type=5))

The full Fisher Information Matrix (FIM) for one individual Calculating one model switch at a time, good for large matrices.
Description

Compute the full FIM for one individual given specific model(s), parameters, design and methods. This computation calculates the FIM for each model switch separately. Correlations between the models parameters are assumed to be zero.

Usage

mf7(model_switch, xt_ind, x, a, bpop, d, sigma, docc, poped.db)

Arguments

- model_switch: A vector that is the same size as xt, specifying which model each sample belongs to.
- xt_ind: A vector of sample times.
- x: A vector for the discrete design variables.
- a: A vector of covariates.
- bpop: The fixed effects parameter values. Supplied as a vector.
- d: A between subject variability matrix (OMEGA in NONMEM).
- sigma: A residual unexplained variability matrix (SIGMA in NONMEM).
- docc: A between occasion variability matrix.
- poped.db: A PopED database.

Value

As a list:

- ret: The FIM for one individual
- poped.db: A PopED database

See Also

Used by mf	ot6.

Other FIM: LinMatrixH; LinMatrixLH; LinMatrixL_occ; calc_ofv_and_fim; ed_laplace_ofv; ed_mftot; evaluate.e.ofv.fim; evaluate.fim; gradf_eps; mf3; mf5; mf6; mf8; m	ot0; m	ot1; m	ot2; m	ot3; m	ot4; m	ot5; m	ot6; m	ot7; m	ot8; m	ot9; ofv_criterion; ofv_fim

Examples

```r
## Warfarin example from software comparison in:
## Nyberg et al., "Methods and software tools for design evaluation
## for population pharmacokinetics-pharmacodynamics studies",

library(PopED)

## find the parameters that are needed to define from the structural model
ff.PK.1.comp.oral.sd.CL
```
## -- parameter definition function
## -- names match parameters in function ff
sfg <- function(x,a,bpop,bocc){
  parameters=c(CL=bpop[1]*exp(b[1]),
             V=bpop[2]*exp(b[2]),
             KA=bpop[3]*exp(b[3]),
             Favail=bpop[4],
             DOSE=a[1])
  return(parameters)
}

## -- Define initial design and design space
poped.db <- create.poped.database(ff_file="ff.PK.1.comp.oral.sd.CL",
                                   fg_file="sfg",
                                   fError_file="feps.prop",
                                   bpop=c(CL=0.15, V=8, KA=1.0, Favail=1),
                                   notfixed_bpop=c(1,1,1,0),
                                   d=c(CL=0.07, V=0.02, KA=0.6),
                                   sigma=0.01,
                                   groupsize=32,
                                   xt=c(0.5,1,2,6,24,36,72,120),
                                   minxt=0,
                                   maxxt=120,
                                   a=70)

# warfarin optimization model

# for the FO approximation
ind=1

# no occasion defined in this example, so result is zero
output <- mf8(model_switch=t(poped.db$design$model_switch[ind,,drop=FALSE]),
               xt=t(poped.db$design$x[t[ind,,drop=FALSE]),
               x=zeros(0,1),
               a=t(poped.db$design$a[ind,,drop=FALSE]),
               bpop=poped.db$parameters$bpop[,2,drop=FALSE],
               d=poped.db$parameters$param.pt.val$d,
               sigma=poped.db$parameters$sigma,
               docc=poped.db$parameters$param.pt.val$docc,
               poped.db)

# in this simple case the full FIM is just the sum of the individual FIMs
# and all the individual FIMs are the same
det(output$ret*32) == det(evaluate.fim(poped.db,fim.calc.type=6))
Description

Compute the reduced FIM for one individual given specific model(s), parameters, design and methods. This computation assumes that there is no correlation in the FIM between the fixed and random effects, and set these elements in the FIM to zero. This computation parameterizes the FIM calculation using A,B,C matrices (as in Retout et al.) but uses the derivative of variances. Should give the same answer as mf3 but computation times may be different.

Usage

mf8(model_switch, xt_ind, x, a, bpop, d, sigma, docc, poped.db)

Arguments

model_switch A vector that is the same size as xt, specifying which model each sample belongs to.
xt_ind A vector of sample times.
x A vector for the discrete design variables.
a A vector of covariates.
bpop The fixed effects parameter values. Supplied as a vector.
d A between subject variability matrix (OMEGA in NONMEM).
sigma A residual unexplained variability matrix (SIGMA in NONMEM).
docc A between occasion variability matrix.
poped.db A PopED database.

Value

As a list:

ret The FIM for one individual
poped.db A PopED database

References


See Also

Used by mftot7.

Other FIM: LinMatrixH; LinMatrixLH; LinMatrixL_occ; calc_ofv_and_fim; ed_laplace_ofv; ed_mftot; evaluate.e.ofv.fim; evaluate.fim; gradf_eps; mf3; mf5; mf6; mf7; mftot0; mftot1; mftot2; mftot3; mftot4; mftot5; mftot6; mftot7; mftot; mf; ofv_criterion; ofv_fim
Examples

```r
## Warfarin example from software comparison in:
## Nyberg et al., "Methods and software tools for design evaluation
## for population pharmacokinetics-pharmacodynamics studies",

library(PopED)

## find the parameters that are needed to define from the structural model
ff.PK.1.comp.oral.sd.CL

## -- parameter definition function
## -- names match parameters in function ff
sfg <- function(x,a,bpop,b,bocc){
  parameters=c(CL=bpop[1]*exp(b[1]),
             V=bpop[2]*exp(b[2]),
             KA=bpop[3]*exp(b[3]),
             Favail=bpop[4],
             DOSE=a[1])
  return(parameters)
}

## -- Define initial design and design space
poped.db <- create.poped.database(ff_file="ff.PK.1.comp.oral.sd.CL",
                                   fg_file="sfg",
                                   ferror_file="feps.prop",
                                   bpop= c(CL=0.15, V=8, KA=1, Favail=1),
                                   notfixed_bpop=c(1,1,1,0),
                                   d= c(CL=0.07, V=0.02, KA=0.6),
                                   sigma=0.01,
                                   groups=32,
                                   xt=c(0.5,1,2,6,24,36,72,120),
                                   minxt=0,
                                   maxxt=120,
                                   a=70)

# warfarin optimization model

# for the FO approximation
ind=1

# no occasion defined in this example, so result is zero
output <- mf8(model_switch=t(poped.db$design$model_switch[ind,,drop=FALSE]),
              xt=t(poped.db$design$xt[ind,,drop=FALSE]),
              x=zeros(0,1),
              a=t(poped.db$design$a[ind,,drop=FALSE]),
              bpop=poped.db$parameters$bpop[,2,drop=FALSE],
              d=poped.db$parameters$param.pt.val$d,
              sigma=poped.db$parameters$sigma,
              docc=poped.db$parameters$param.pt.val$docc,
              poped.db)

# in this simple case the full FIM is just the sum of the individual FIMs
```
# and all the individual FIMs are the same
def(output$ret*32) == def(evaluate.fim(poped.db,fim.calc.type=7))

---

**mfea**  
*Modified Federov Exchange Algorithm*

**Description**

Optimize the objective function using a modified Federov exchange algorithm. The function works for continuous and discrete optimization variables. This function takes information from the PopED database supplied as an argument. The PopED database supplies information about the the model, parameters, design and methods to use. Some of the arguments coming from the PopED database can be overwritten; if they are supplied then they are used instead of the arguments from the PopED database.

**Usage**

```r
mfea(poped.db, model_switch, ni, xt, x, a, bpopdescr, ddescr, maxxt, minxt, maxa, mina, fmf, dnf, EAStepSize = poped.db$settings$EAStepSize, ourzero = poped.db$settings$ourzero, opt_xt = poped.db$settings$optsw[2], opt_a = poped.db$settings$optsw[4], opt_x = poped.db$settings$optsw[3], trflag = T, ...)
```

**Arguments**

- `poped.db` A PopED database.
- `model_switch` A matrix that is the same size as `xt`, specifying which model each sample belongs to.
- `ni` A vector of the number of samples in each group.
- `xt` A matrix of sample times. Each row is a vector of sample times for a group.
- `x` A matrix for the discrete design variables. Each row is a group.
- `a` A matrix of covariates. Each row is a group.
- `bpopdescr` Matrix defining the fixed effects, per row (row number = parameter_number) we should have:
  - column 1 the type of the distribution for E-family designs (0 = Fixed, 1 = Normal, 2 = Uniform, 3 = User Defined Distribution, 4 = lognormal and 5 = truncated normal)
  - column 2 defines the mean.
  - column 3 defines the variance of the distribution (or length of uniform distribution).
- `ddescr` Matrix defining the diagonals of the IIV (same logic as for the `bpopdescr`).
- `maxxt` Matrix or single value defining the maximum value for each `xt` sample. If a single value is supplied then all `xt` values are given the same maximum value.
minxt  Matrix or single value defining the minimum value for each xt sample. If a single value is supplied then all xt values are given the same minimum value.

maxa  Vector defining the max value for each covariate. If a single value is supplied then all a values are given the same max value.

mina  Vector defining the min value for each covariate. If a single value is supplied then all a values are given the same max value.

f mf  The initial value of the FIM. If set to zero then it is computed.

d mf  The initial OFV. If set to zero then it is computed.

EAStepSize  Exchange Algorithm StepSize

ourzero  Value to interpret as zero in design

opt_xt  Should the sample times be optimized?

opt_a  Should the continuous design variables be optimized?

opt_x  Should the discrete design variables be optimized?

trflag  Should the optimization be output to the screen and to a file?

...  arguments passed to evaluate.fim and ofv.fim.

References


See Also

Other Optimize: Doptim; LEDoptim; RS_opt_gen; RS_opt; a_line_search; bfgsb_min; calc Autofocus; calc_ofv_and_grad; poped_optimize

Examples

```
## Warfarin example from software comparison in:
## Nyberg et al., "Methods and software tools for design evaluation
## for population pharmacokinetics-pharmacodynamics studies",

## Optimization using an additive + proportional reidual error to
## avoid sample times at very low concentrations (time 0 or very late samoples).
library(PopED)

## find the parameters that are needed to define from the structural model
ff.PK.1.com.oral.sd.CL

## -- parameter definition function
## -- names match parameters in function ff
sfg <- function(x,a,bpop,b,bocc){
  parameters=c(CL=bpop[1]*exp(b[1]),
               V=bpop[2]*exp(b[2]),
               KA=bpop[3]*exp(b[3]),
               ...}
```
```r
Favail=bpop[4],
DOSE=a[1])
return(parameters)
}

## -- Define initial design and design space
poped.db <- create.poped.database(ff_file="ff.PK.1.comp.oral.sd.CL",
fg_file="sfg",
fError_file="feps.add.prop",
bpop=c(CL=0.15, V=8, KA=1.0, Favail=1),
notfixed_bpop=c(1,1,1,0),
d=c(CL=0.07, V=0.02, KA=0.6),
sigma=c(0.01,0.25),
groupsize=32,
xt=c(0.5,1,2,6,24,36,72,120),
minxt=0,
maxxt=120,
a=70,
mina=0,
maxa=100)

# warfarin optimize model

## Not run:

###################
# typically one will use poped_optimize
# This then calls mfea
###################

# MFEA optimization with only integer times allowed
mfea.output <- poped_optimize(poped.db,opt_xt=1,
bUseExchangeAlgorithm=1,
EAStepSize=1)
get_rse(mfea.output$fmf,mfea.output$poped.db)
plot_model_prediction(mfea.output$poped.db)

###################
# If you really want to you can use mfea directly
###################
dsl <- downsizing_general_design(poped.db)

output <- mfea(poped.db,
model_switch=dsl$model_switch,
ni=ds1$ni,
xt=ds1$xt,
x=ds1$x,
a=ds1$a,
bpopdescr=ds1$bpop,
ddescr=ds1$d,
maxxt=ds1$maxxt,
minxt=ds1$minxt,
maxa=ds1$maxa,
```
Evaluate the Fisher Information Matrix (FIM)

Description

Compute the FIM given specific model(s), parameters, design and methods.

Usage

mftot(model_switch, groupsize, ni, xt, x, a, bpop, d, sigma, docc, poped.db)

Arguments

- model_switch: A matrix that is the same size as xt, specifying which model each sample belongs to.
- groupsize: A vector of the number of individuals in each group.
- ni: A vector of the number of samples in each group.
- xt: A matrix of sample times. Each row is a vector of sample times for a group.
- x: A matrix for the discrete design variables. Each row is a group.
- a: A matrix of covariates. Each row is a group.
- bpop: The fixed effects parameter values. Supplied as a vector.
- d: A between subject variability matrix (OMEGA in NONMEM).
- sigma: A residual unexplained variability matrix (SIGMA in NONMEM).
- docc: A between occasion variability matrix.
- poped.db: A PopED database.

Value

As a list:

- ret: The FIM
- poped.db: A PopED database
See Also

For an easier function to use, please see `evaluate.fim`.

Other FIM: `LinMatrixH`; `LinMatrixLH`; `LinMatrixL_occ`; `calc_ofv_and_fim`; `ed_laplace_ofv`; `ed_mftot`; `evaluate.e.ofv.fim`; `evaluate.fim`; `gradf_eps`; `mf3`; `mf5`; `mf6`; `mf7`; `mf8`; `mftot0`; `mftot1`; `mftot2`; `mftot3`; `mftot4`; `mftot5`; `mftot6`; `mftot7`; `mf`; `ofv.criterion`; `ofv_fim`

Examples

```r
## Warfarin example from software comparison in:
## Nyberg et al., "Methods and software tools for design evaluation
## for population pharmacokinetics-pharmacodynamics studies",

## Optimization using an additive + proportional residual error to
## avoid sample times at very low concentrations (time 0 or very late samples).
library(PopED)

## find the parameters that are needed to define from the structural model
ff.PK.1.comp.oral.sd.CL

## -- parameter definition function
## -- names match parameters in function ff
sfg <- function(x,a,bpop,b,bocc){
  parameters=c(CL=bpop[1]*exp(b[1]),
               V=bpop[2]*exp(b[2]),
               KA=bpop[3]*exp(b[3]),
               Favail=bpop[4],
               DOSE=a[1])
  return(parameters)
}

## -- Define initial design and design space
poped.db <- create.poped.database(ff_file="ff.PK.1.comp.oral.sd.CL",
                                   fg_file="sfg",
                                   fError_file="feps.add.prop",
                                   bpop=c(CL=0.15, V=8, KA=1.0, Favail=1),
                                   notfixed_bpop=c(1,1,1,0),
                                   d=c(CL=0.07, V=0.02, KA=0.6),
                                   sigma=c(0.01,0.25),
                                   groupsize=32,
                                   xt=c(0.5,1,2,6,24,36,72,120),
                                   minxt=0,
                                   maxxt=120,
                                   a=70,
                                   mina=0,
                                   maxa=100)

# warfarin optimization model

mftot(model_switch=poped.db$design$model_switch,
```
mftot0

The full Fisher Information Matrix (FIM)

Description

Compute the full FIM given specific model(s), parameters, design and methods. This computation makes no assumption that fixed and random effects are uncorrelated.

Usage

mftot0(model_switch, groupsize, ni, xt, x, a, bpop, d, sigma, docc, poped.db)

Arguments

model_switch
A matrix that is the same size as xt, specifying which model each sample belongs to.

groupsize
A vector of the number of individuals in each group.

ni
A vector of the number of samples in each group.

xt
A matrix of sample times. Each row is a vector of sample times for a group.

x
A matrix for the discrete design variables. Each row is a group.

a
A matrix of covariates. Each row is a group.

bpop
The fixed effects parameter values. Supplied as a vector.

d
A between subject variability matrix (OMEGA in NONMEM).

sigma
A residual unexplained variability matrix (SIGMA in NONMEM).

docc
A between occasion variability matrix.

poped.db
A PopED database.

Value

As a list:

ret
The FIM

poped.db
A PopED database
See Also

For an easier function to use, please see `evaluate.fim`.

Other FIM: `LinMatrixH`; `LinMatrixLH`; `LinMatrixL_occ`; `calc_ofv_and_fim`; `ed_laplace_ofv`; `ed_mftot`; `evaluate.e.ofv.fim`; `evaluate.fim`; `gradf_eps`; `mf3`; `mf5`; `mf6`; `mf7`; `mf8`; `mftot1`; `mftot2`; `mftot3`; `mftot4`; `mftot5`; `mftot6`; `mftot7`; `mftot8`; `mf`; `ofv.criterion`; `ofv_fim`

Examples

```r
## Warfarin example from software comparison in:
## Nyberg et al., "Methods and software tools for design evaluation
## for population pharmacokinetics-pharmacodynamics studies",

## Optimization using an additive + proportional reidual error to
## avoid sample times at very low concentrations (time 0 or very late samoples).
library(PopED)

## find the parameters that are needed to define from the structural model
ff.PK.1.comp.oral.sd.CL

## -- parameter definition function
## -- names match parameters in function ff
sfg <- function(x,a,bpop,bboc){
  parameters=c(CL=bpop[1]*exp(b[1]),
               V=bpop[2]*exp(b[2]),
               KA=bpop[3]*exp(b[3]),
               Favail=bpop[4],
               DOSE=a[1])
  return(parameters)
}

## -- Define initial design and design space
poped.db <- create.poped.database(ff_file="ff.PK.1.comp.oral.sd.CL",
                                    fg_file="sfg",
                                    fError_file="feps.add.prop",
                                    bpop=c(CL=0.15, V=8, KA=1.0, Favail=1),
                                    notfixed_bpop=c(1,1,1,0),
                                    d=c(CL=0.07, V=0.02, KA=0.6),
                                    sigma=c(0.01,0.05),
                                    groupsize=32,
                                    xt=c( 0.5,1,2,6,24,36,72,120),
                                    minxt=0,
                                    maxxt=120,
                                    a=70,
                                    mina=0,
                                    maxa=100)

# warfarin optimization model

mftot0(model_switch=poped.db$design$model_switch,
```
The reduced Fisher Information Matrix (FIM)

Description

Compute the reduced FIM given specific model(s), parameters, design and methods. This computation assumes that there is no correlation in the FIM between the fixed and random effects, and set these elements in the FIM to zero.

Usage

mftot1(model_switch, groupsize, ni, xt, x, a, bpop, d, sigma, docc, poped.db)

Arguments

- `model_switch` A matrix that is the same size as `xt`, specifying which model each sample belongs to.
- `groupsize` A vector of the number of individuals in each group.
- `ni` A vector of the number of samples in each group.
- `xt` A matrix of sample times. Each row is a vector of sample times for a group.
- `x` A matrix for the discrete design variables. Each row is a group.
- `a` A matrix of covariates. Each row is a group.
- `bpop` The fixed effects parameter values. Supplied as a vector.
- `d` A between subject variability matrix (OMEGA in NONMEM).
- `sigma` A residual unexplained variability matrix (SIGMA in NONMEM).
- `docc` A between occasion variability matrix.
- `poped.db` A PopED database.

Value

As a list:

- `ret` The FIM
- `poped.db` A PopED database
See Also

For an easier function to use, please see `evaluate.fim`.

Other FIM: `LinMatrixH; LinMatrixLH; LinMatrixL_occ; calc_ofv_and_fim; ed_laplace_ofv; ed_mftot; evaluate.e.ofv.fim; evaluate.fim; gradf_eps; mf3; mf5; mf6; mf7; mf8; mftot0; mftot2; mftot3; mftot4; mftot5; mftot6; mftot7; mftot8; ofv_criterion; ofv_fim`

Examples

```r
## Warfarin example from software comparison in:
## Nyberg et al., "Methods and software tools for design evaluation
## for population pharmacokinetics-pharmacodynamics studies",

## Optimization using an additive + proportional residual error to
## avoid sample times at very low concentrations (time 0 or very late samples).
library(PopED)

## find the parameters that are needed to define from the structural model
ff.PK.1.comp.oral.sd.CL

## -- parameter definition function
## -- names match parameters in function ff
sfg <- function(x,a,bpop,bocc)
{
  parameters=c(CL=bpop[1]*exp(b[1]),
              V=bpop[2]*exp(b[2]),
              KA=bpop[3]*exp(b[3]),
              Favail=bpop[4],
              DOSE=a[1])
  return(parameters)
}

## -- Define initial design and design space
poped.db <- create.poped.database(ff_file="ff.PK.1.comp.oral.sd.CL",
                                    fg_file="sfg",
                                    fError_file="feps.add.prop",
                                    bpop=c(CL=0.15, V=8, KA=1.0, Favail=1),
                                    notfixed_bpop=c(1,1,1,0),
                                    d=c(CL=0.07, V=0.02, KA=0.6),
                                    sigma=c(0.01,0.25),
                                    groupsize=32,
                                    xt=c(0.5,1,2,6,24,36,72,120),
                                    minxt=0,
                                    maxxt=120,
                                    a=70,
                                    mina=0,
                                    maxa=100)

# warfarin optimization model

mftot1(model_switch=poped.db$design$model_switch,
```
The Fisher Information Matrix (FIM) using weighted models

**Description**

Compute the FIM using weighted models given specific model(s), parameters, design and methods. Not currently available.

**Usage**

```r
mftot2(model_switch, groupsize, ni, xt, x, a, bpop, d, sigma, docc, poped.db)
```

**Arguments**

- `model_switch`: A matrix that is the same size as xt, specifying which model each sample belongs to.
- `groupsize`: A vector of the number of individuals in each group.
- `ni`: A vector of the number of samples in each group.
- `xt`: A matrix of sample times. Each row is a vector of sample times for a group.
- `x`: A matrix for the discrete design variables. Each row is a group.
- `a`: A matrix of covariates. Each row is a group.
- `bpop`: The fixed effects parameter values. Supplied as a vector.
- `d`: A between subject variability matrix (OMEGA in NONMEM).
- `sigma`: A residual unexplained variability matrix (SIGMA in NONMEM).
- `docc`: A between occasion variability matrix.
- `poped.db`: A PopED database.

**See Also**

For an easier function to use, please see `evaluate.fim`.

Other FIM: `LinMatrixH`; `LinMatrixLH`; `LinMatrix_occ`; `calc_ofv_and_fim`; `ed_laplace_ofv`; `ed_mftot2`; `evaluate.e.ofv.fim`; `evaluate.fim`; `gradf_eps`; `mf3`; `mf5`; `mf6`; `mf7`; `mf8`; `mftot0`; `mftot1`; `mftot2`; `mftot3`; `mftot4`; `mftot5`; `mftot6`; `mftot7`; `mftot`; `mf`; `ofv_criterion`; `ofv_fim`
The Fisher Information Matrix (FIM) some other method

Description

Compute the FIM using some other method given specific model(s), parameters, design and methods. This is a placeholder.

Usage

mftot3(model_switch, groupsize, ni, xt, x, a, bpop, d, sigma, docc, poped.db)

Arguments

model_switch A matrix that is the same size as xt, specifying which model each sample belongs to.

•

groupsize A vector of the number of individuals in each group.

•

ni A vector of the number of samples in each group.

•

xt A matrix of sample times. Each row is a vector of sample times for a group.

•

x A matrix for the discrete design variables. Each row is a group.

•

a A matrix of covariates. Each row is a group.

•

bpop The fixed effects parameter values. Supplied as a vector.

•

d A between subject variability matrix (OMEGA in NONMEM).

•

sigma A residual unexplained variability matrix (SIGMA in NONMEM).

•

docc A between occasion variability matrix.

•

poped.db A PopED database.

See Also

For an easier function to use, please see evaluate.fim.

Other FIM: LinMatrixH; LinMatrixLH; LinMatrixL_occ; calc_ofv_and_fim; ed_laplace_ofv; ed_mftot; evaluate.e.ofv.fim; evaluate.fim; gradf_eps; mf3; mf5; mf6; mf7; mf8; mftot0; mftot1; mftot2; mftot4; mftot5; mftot6; mftot7; mftot; mf; ofv_criterion; ofv_fim
The reduced Fisher Information Matrix (FIM) using the SD of RUV as a parameter.

Description

Compute the reduced FIM using the standard deviation of the residual unexplained variability (RUV) terms as a parameter, given specific model(s), parameters, design and methods. This computation assumes that there is no correlation in the FIM between the fixed and random effects, and set these elements in the FIM to zero. In addition all derivatives in the computation are made with respect to the standard deviation of the RUV terms (sqrt(SIGMA) in NONMEM). This matches what is done in PFIM, and assumes that the standard deviation of the residual unexplained variation is the estimated parameter (NOTE: NONMEM estimates the variance of the residual unexplained variation by default).

Usage

mftot4(model_switch, groupsize, ni, xt, x, a, bpop, d, sigma, docc, poped.db)

Arguments

model_switch A matrix that is the same size as xt, specifying which model each sample belongs to.
groupsize A vector of the number of individuals in each group.
ni A vector of the number of samples in each group.
xt A matrix of sample times. Each row is a vector of sample times for a group.
x A matrix for the discrete design variables. Each row is a group.
a A matrix of covariates. Each row is a group.
bpop The fixed effects parameter values. Supplied as a vector.
d A between subject variability matrix (OMEGA in NONMEM).
sigma A residual unexplained variability matrix (SIGMA in NONMEM).
docc A between occasion variability matrix.
poped.db A PopED database.

Value

As a list:

ret The FIM
poped.db A PopED database
See Also

For an easier function to use, please see evaluate.fim.

Other FIM: LinMatrixH; LinMatrixLH; LinMatrixL_occ; calc_ofv_and_fim; ed_laplace_ofv;
ed_mftot; evaluate.e.ofv.fim; evaluate.fim; gradf_eps; mf3; mf5; mf6; mf7; mf8; mftot0;
mftot1; mftot2; mftot3; mftot4; mftot5; mftot6; mftot7; mftot8; ofv_criterion; ofv_fim

Examples

```r
## Warfarin example from software comparison in:
## Nyberg et al., "Methods and software tools for design evaluation
## for population pharmacokinetics-pharmacodynamics studies",

## Optimization using an additive + proportional residual error to
## avoid sample times at very low concentrations (time 0 or very late samples).
library(PopED)

## find the parameters that are needed to define from the structural model
ff.PK.1.comp.oral.sd.CL

## -- parameter definition function
## -- names match parameters in function ff
sfg <- function(x,a,bpop,bocc){
    parameters=c(CL=bpop[1]*exp(b[1]),
                V=bpop[2]*exp(b[2]),
                KA=bpop[3]*exp(b[3]),
                Favail=bpop[4],
                DOSE=a[1])
    return(parameters)
}

## -- Define initial design and design space
poped.db <- create.poped.database(ff.file="ff.PK.1.comp.oral.sd.CL",
                                   fg.file="sfg",
                                   fError_file="feps.add.prop",
                                   bpop=c(CL=0.15, V=8, KA=1.0, Favail=1),
                                   notfixed_bpop=c(1,1,1,0),
                                   d=c(CL=0.07, V=0.02, KA=0.6),
                                   sigma=c(0.01,0.25),
                                   groupsize=32,
                                   xt=c(0.5,1,2,6,24,36,72,120),
                                   minxt=0,
                                   maxxt=120,
                                   a=70,
                                   mina=0,
                                   maxa=100)

# warfarin optimization model

mftot4(model_switch=poped.db$design$model_switch,
```
The full Fisher Information Matrix (FIM) parameterized with A,B,C matrices & using the derivative of variance.

Description

Compute the full FIM given specific model(s), parameters, design and methods. This computation parameterizes the FIM calculation using A,B,C matrices (as in Retout et al.) but uses the derivative of variances. Should give the same answer as mftotP but computation times may be different.

Usage

mftot5(model_switch, groupsize, ni, xt, x, a, bpop, d, sigma, docc, poped.db)

Arguments

- **model_switch**: A matrix that is the same size as xt, specifying which model each sample belongs to.
- **groupsize**: A vector of the number of individuals in each group.
- **ni**: A vector of the number of samples in each group.
- **xt**: A matrix of sample times. Each row is a vector of sample times for a group.
- **x**: A matrix for the discrete design variables. Each row is a group.
- **a**: A matrix of covariates. Each row is a group.
- **bpop**: The fixed effects parameter values. Supplied as a vector.
- **d**: A between subject variability matrix (OMEGA in NONMEM).
- **sigma**: A residual unexplained variability matrix (SIGMA in NONMEM).
- **docc**: A between occasion variability matrix.
- **poped.db**: A PopED database.

Value

As a list:

- **ret**: The FIM
- **poped.db**: A PopED database
References


See Also

For an easier function to use, please see evaluate.fim.

Other FIM: LinMatrixH; LinMatrixLH; LinMatrixL_occ; calc_ofv_and_fim; ed_laplace_ofv; ed_mftot; evaluate.e.ofv.fim; evaluate.fim; gradf_eps; mf3; mf5; mf6; mf7; mf8; mftot0; mftot1; mftot2; mftot3; mftot4; mftot6; mftot7; mftot; mf; ofv_criterion; ofv_fim

Examples

```r
## Warfarin example from software comparison in:
## Nyberg et al., "Methods and software tools for design evaluation
## for population pharmacokinetics-pharmacodynamics studies",
##
## Optimization using an additive + proportional residual error to
## avoid sample times at very low concentrations (time 0 or very late samples).
library(PopED)

## find the parameters that are needed to define from the structural model
ff.PK.1.comp.oral.sd.CL

## -- parameter definition function
## -- names match parameters in function ff
sfg <- function(x,a,bpop,b,bocc){
  parameters=c(CL=bpop[1]*exp(b[1]),
                V=bpop[2]*exp(b[2]),
                KA=bpop[3]*exp(b[3]),
                Favail=bpop[4],
                DOSE=a[1])
  return(parameters)
}

## -- Define initial design and design space
poped.db <- create.poped.database(ff_file="ff.PK.1.comp.oral.sd.CL",
                                    fg_file="sfg",
                                    fError_file="feps.add.prop",
                                    bpop=c(CL=0.15, V=8, KA=1.0, Favail=1),
                                    notfixed_bpop=c(1,1,1,0),
                                    d=c(CL=0.07, V=0.02, KA=0.6),
                                    sigma=c(0.01,0.25),
                                    groupsize=32,
                                    xt=c(0.5,1,2,6,24,36,72,120),
                                    minxt=0,
                                    maxxt=120,
                                    a=70,
                                    mina=0,
                                    maxa=100)
```
The full Fisher Information Matrix (FIM) Calculating one model switch at a time, good for large matrices.

Description
Compute the full FIM given specific model(s), parameters, design and methods. This computation calculates the FIM for each model switch separately. Correlations between the models parameters are assumed to be zero.

Usage
mftot6(model_switch, groupsize, ni, xt, x, a, bpop, d, sigma, docc, poped.db)

Arguments
- **model_switch**: A matrix that is the same size as xt, specifying which model each sample belongs to.
- **groupsize**: A vector of the number of individuals in each group.
- **ni**: A vector of the number of samples in each group.
- **xt**: A matrix of sample times. Each row is a vector of sample times for a group.
- **x**: A matrix for the discrete design variables. Each row is a group.
- **a**: A matrix of covariates. Each row is a group.
- **bpop**: The fixed effects parameter values. Supplied as a vector.
- **d**: A between subject variability matrix (OMEGA in NONMEM).
- **sigma**: A residual unexplained variability matrix (SIGMA in NONMEM).
- **docc**: A between occasion variability matrix.
- **poped.db**: A PopED database.
Value

As a list:

ret The FIM
poped.db A PopED database

See Also

For an easier function to use, please see `evaluate.fim`.

Other FIM: `LinMatrixH`; `LinMatrixLH`; `LinMatrixL_occ`; `calc_ofv_and_fim`; `ed_laplace_ofv
ed_mftot`; `evaluate.e.ofv.fim`; `evaluate.fim`; `gradf_eps`; `mf3`; `mf5`; `mf6`; `mf7`; `mf8`; `mftot0`; `mftot1`; `mftot2`; `mftot3`; `mftot4`; `mftot5`; `mftot7`; `mftot9`; `mf`; `ofv_criterion`; `ofv_fim`

Examples

```r
## Warfarin example from software comparison in:
## Nyberg et al., "Methods and software tools for design evaluation
## for population pharmacokinetics-pharmacodynamics studies",

## Optimization using an additive + proportional residual error to
## avoid sample times at very low concentrations (time 0 or very late samples).
library(PopED)

## find the parameters that are needed to define from the structural model
ff.PK.1.comp.oral.sd.CL

## -- parameter definition function
## -- names match parameters in function ff
sfg <- function(x,a,bpop,bcc){
  parameters=c(CL=bpop[1]*exp(b[1]),
             V=bpop[2]*exp(b[2]),
             KA=bpop[3]*exp(b[3]),
             Favail=bpop[4],
             DOSE=a[1])
  return(parameters)
}

## -- Define initial design and design space
poped.db <- create.poped.database(ff_file="ff.PK.1.comp.oral.sd.CL",
                                    fg_file="sfg",
                                    fError_file="feps.add.prop",
                                    bpop=c(CL=0.15, V=8, KA=1.0, Favail=1),
                                    notfixed_bpop=c(1,1,1,0),
                                    d=c(CL=0.07, V=0.02, KA=0.6),
                                    sigma=c(0.01,0.25),
                                    groupsize=32,
                                    xt=c(0.5,1,2,6,24,36,72,120),
                                    minxt=0,
                                    maxxt=120,
                                    a=70,
                                    b=1.0,
                                    Xmax=180,
                                    Xmin=0,
                                    Xaxis="time",
                                    Xlabel="time",
                                    Ymax=4,
                                    Ymin=0,
                                    Yaxis="value",
                                    Ylabel="value",
                                    Fmax=20,
                                    Fmin=0,
                                    Faxis="value",
                                    Flabel="value",
                                    Omax=0,
                                    Omin=0,
                                    Oaxis="value",
                                    Olabel="value",
                                    n=1600,
                                    nubi=10,
                                    nubi_method="log",
                                    nubi_step=1,
                                    nubi_min=1,
                                    nubi_max=1600,
                                    n3min=10,
                                    n3max=1600,
                                    n3step=1,
                                    n3min_method="log",
                                    n3max_method="log",
                                    n3step_method="log",
                                    n3max_step=1,
                                    n3max_min=1,
                                    n3max_max=1600,
                                    n3max_min_method="log",
                                    n3max_max_method="log",
                                    n3max_min_step=1,
                                    n3max_max_step=1,
                                    n3min_min_method="log",
                                    n3min_max_method="log",
                                    n3min_min_step=1,
                                    n3min_max_step=1,
                                    n3min_min_min_method="log",
                                    n3min_max_min_method="log",
                                    n3min_max_min_step=1,
                                    n3min_max_min_min_method="log",
                                    n3min_max_min_min_step=1,
                                    n3min_max_min_min_min_method="log",
                                    n3min_max_min_min_min_step=1,
                                    n3min_max_min_min_min_min_method="log",
                                    n3min_max_min_min_min_min_step=1,
                                    n3min_max_min_min_min_min_min_method="log",
                                    n3min_max_min_min_min_min_min_step=1,
                                    n3min_max_min_min_min_min_min_min_method="log",
                                    n3min_max_min_min_min_min_min_min_step=1,
                                    n3min_max_min_min_min_min_min_min_min_method="log",
                                    n3min_max_min_min_min_min_min_min_min_step=1,
                                    n3min_max_min_min_min_min_min_min_min_min_method="log",
                                    n3min_max_min_min_min_min_min_min_min_min_step=1,
                                    n3min_max_min_min_min_min_min_min_min_min_min_method="log",
                                    n3min_max_min_min_min_min_min_min_min_min_min_step=1,
                                    n3min_max_min_min_min_min_min_min_min_min_min_min_method="log",
                                    n3min_max_min_min_min_min_min_min_min_min_min_min_min_step=1
                                    )
```
```r
```
mftot7

The reduced Fisher Information Matrix (FIM) parameterized with A,B,C matrices & using the derivative of variance.

Description

Compute the reduced FIM given specific model(s), parameters, design and methods. This computation assumes that there is no correlation in the FIM between the fixed and random effects, and set these elements in the FIM to zero. This computation parameterizes the FIM calculation using A,B,C matrices (as in Retout et al.) but uses the derivative of variances. Should give the same answer as mftot1 but computation times may be different.

Usage

mftot7(model_switch, groupsize, ni, xt, x, a, bpop, d, sigma, docc, poped.db)
sigma: A residual unexplained variability matrix (SIGMA in NONMEM).
docc: A between occasion variability matrix.
poped.db: A PopED database.

Value

As a list:

- ret: The FIM
- poped.db: A PopED database

References

S. Retout and F. Mentre, "Further developments of the Fisher Information Matrix in nonlinear mixed

See Also

For an easier function to use, please see `evaluate.fim`.

Other FIM: `LinMatrixH`; `LinMatrixLH`; `LinMatrixL_occ`; `calc_ofv_and_fim`; `ed_laplace_ofv`;
`ed_mftot`; `evaluate.e.ofv.fim`; `evaluate.fim`; `gradf_eps`; `mf3`; `mf5`; `mf6`; `mf7`; `mf8`; `mftot0`;
`mftot1`; `mftot2`; `mftot3`; `mftot4`; `mftot5`; `mftot6`; `mftot7`; `ofv_criterion`; `ofv_fim`

Examples

```r
## Warfarin example from software comparison in:
## Nyberg et al., "Methods and software tools for design evaluation
## for population pharmacokinetics-pharmacodynamics studies",

## Optimization using an additive + proportional residual error to
## avoid sample times at very low concentrations (time 0 or very late samples).
library(PopED)

## find the parameters that are needed to define from the structural model
ff.PK.1.comp.oral.sd.CL

## -- parameter definition function
## -- names match parameters in function ff
sfg <- function(x,a,bpop,b,bocc){
  parameters=c(CL=bpop[1]*exp(b[1]),
         V=bpop[2]*exp(b[2]),
         KA=bpop[3]*exp(b[3]),
         Favail=bpop[4],
         DOSE=a[1])
  return(parameters)
}

## -- Define initial design and design space
poped.db <- create.poped.database(ff_file="ff.PK.1.comp.oral.sd.CL",
                                   fg_file="sfg")
```
model_prediction

Description

Function generates a data frame of model predictions for the typical value in the population, individual predictions and data predictions. The function can also be used to generate datasets without predictions using the design specified in the arguments.

Usage

```r
model_prediction(poped.db = NULL, design = list(x = poped.db$design["xt"], groupsize = poped.db$design$groupsize, m = poped.db$design["m"], a = poped.db$design["a"], ni = poped.db$design$ni, model_switch = poped.db$design$model_switch), model = list(fg_pointer = poped.db$model$fg_pointer, ff_pointer = poped.db$model$ff_pointer, ferror_pointer = poped.db$model$ferror_pointer), parameters = list(docc = poped.db$parameters$docc, d = poped.db$parameters$d, bpop = poped.db$parameters$bpop, covd = poped.db$parameters$covd, covdocc =
```
model_prediction

poped.db$parameters$covddcc, sigma = poped.db$parameters$sigma),
IPRED = FALSE, DV = FALSE, dosing = NULL, predictions = NULL,
filename = NULL, models_to_use = "all", model_num_points = NULL,
model_minxt = NULL, model_maxxt = NULL, include_sample_times = T,
groups_to_use = "all", include_a = TRUE, include_x = TRUE,
manipulation = NULL)

Arguments

poped.db A PopED database created by create.poped.database.
design A list that is passed as arguments to the function create_design to create a
design object.
model A list containing the model elements to use for the predictions
parameters A list of parameters to use in the model predictions.
IPRED Should we simulate individual predictions?
DV should we simulate observations?
dosing A list of lists that adds dosing records to the data frame (Each inner list corre-
sponding to a group in the design).
predictions Should the resulting data frame have predictions? Either TRUE or FALSE or NULL
in which case the function decides based on other arguments.
filename A filename that the data frame should be written to in comma separate value
(csv) format.
models_to_use Which model numbers should we use? Model numbers are defined in design
below using model_switch. For an explanation see create_design.
model_num_points How many extra observation rows should be created in the data frame for each
group or individual per model. If used then the points are placed evenly between
model_minxt and model_maxxt. This option is used by plot_model_prediction
to simulate the response of the model on a finer grid then the defined design. If
NULL then only the input design is used. Can be a single value or a vector the
same length as the number of models.
model_minxt The minimum time value for extra observation rows indicated by model_num_points.
A vector the same length as the number of models
model_maxxt The minimum time value for extra observation rows indicated by model_num_points.
A vector the same length as the number of models
include_sample_times Should observations rows in the output data frame include the times indicated
in the input design?
groups_to_use Which groups should we include in the output data frame? Allowed values are
"all" or a vector of numbers indicating the groups to include, e.g. c(1,3,6).
include_a Should we include the continuous design variables in the output?
include_x Should we include the discrete design variables in the output?
**manipulation**

A list of one or more expression arguments. Each expression is evaluated using the code for(i in 1:length(manipulation))(df <- within(df,eval(manipulation[[i]])))). Can be used to transform or create new columns in the resulting data frame. Note that these transformations are created after any model predictions occur, so transformations in columns having to do with input to model predictions will not affect the predictions.

**Value**

A dataframe containing a design and (potentially) simulated data with some dense grid of samples and/or based on the input design.

**See Also**

Other Simulation: plot_efficiency_of_windows; plot_model_prediction
Other evaluate design: evaluate.fim; get_rse; plot_efficiency_of_windows; plot_model_prediction

**Examples**

```r
## Warfarin example from software comparison in:
## Nyberg et al., "Methods and software tools for design evaluation
## for population pharmacokinetics-pharmacodynamics studies",

library(PopED)

## find the parameters that are needed to define from the structural model
ff.PK.1.comp.oral.md.CL

## -- parameter definition function
## -- names match parameters in function ff
sfg <- function(x,a,bpop,b,bocc){
  parameters=c(CL=bpop[1]*exp(b[1]),
              V=bpop[2]*exp(b[2]),
              KA=bpop[3]*exp(b[3]),
              Favail=bpop[4],
              DOSE=a[1])
  return(parameters)
}

## -- Define initial design and design space
poped.db <- create.poped.database(ff_file="ff.PK.1.comp.oral.sd.CL",
                                   fg_file="sfg",
                                   fError_file="feps.prop",
                                   bpop=c(CL=0.15, V=8, KA=1.0, Favail=1),
                                   notfixed_bpop=c(1,1,1,0),
                                   d=c(CL=0.07, V=0.02, KA=0.6),
                                   sigma=0.01,
                                   groupsize=32,
                                   xt=c(0.5,1,2,6,24,36,72,120),
                                   minxt=0,
                                   maxxt=120,
```
```r
a=70

# data frame with model predictions
df_1 <- model_prediction(poped.db)

# data frame with variability
df_2 <- model_prediction(poped.db, DV=TRUE)

# data frame with variability (only IPRED, no DV)
df_3 <- model_prediction(poped.db, IPRED=TRUE)

# data frame with model predictions, no continuous design variables in data frame
df_4 <- model_prediction(poped.db, include_a = FALSE)

# 2 groups
poped.db.2 <- create.poped.database(ff_file="ff.PK.2.comp.oral.sd.CL",
  fg_file="sfg",
  fError_file="feps.prop",
  bpop=c(CL=0.15, V=8, KA=1.0, Favail=1),
  notfixed_bpop=c(1,1,0),
  d=c(CL=0.07, V=0.02, KA=0.6),
  sigma=0.01,
  groupsize=rbind(3,3),
  m=2,
  xt=c( 0.5,1,2,6,24,36,72,120),
  minxt=0,
  maxxt=120,
  a=rbind(70,50))

df_5 <- model_prediction(poped.db.2, DV=TRUE)

# without a poped database, just describing the design
# Useful for creating datasets for use in other software (like NONMEM)
design_1 <- list(
  xt=c( 0.5,1,2,6,24,36,72,120),
  m=2,
  groupsize=3)

design_2 <- list(
  xt=c( 0.5,1,2,6,24,36,72,120),
  m=2,
  groupsize=3,
  a=c(WT=70,AGE=50))

design_3 <- list(
  xt=c( 0.5,1,2,6,24,36,72,120),
  m=2,
  groupsize=3,
  a=list(c(WT=70,AGE=50),c(AGE=45,Wt=60)))

df_6 <- model_prediction(design=design_1)
df_7 <- model_prediction(design=design_2)
df_8 <- model_prediction(design=design_3)
```
df_9 <- model_prediction(design=design_3, DV=TRUE)

# generate random deviations in WT for each individual
df_10 <- model_prediction(design=design_3, DV=TRUE,
                          manipulation=expression({
  for (id in unique(ID))
    WT[ID==id] = rnorm(1, WT[ID==id], WT[ID==id]*0.1); id <- NULL}))

# generate random deviations in WT and AGE for each individual
df_11 <- model_prediction(design=design_3, DV=TRUE,
                          manipulation=list(
    expression(for (id in unique(ID))
      WT[ID==id] = rnorm(1, WT[ID==id], WT[ID==id]*0.1)),
    expression(for (id in unique(ID))
      AGE[ID==id] = rnorm(1, AGE[ID==id], AGE[ID==id]*0.2)),
    expression(id <- NULL)
  ))

# create dosing rows
dosing_1 <- list(list(AMT=1000, RATE=NA, Time=0.5), list(AMT=3000, RATE=NA, Time=0.5))
dosing_2 <- list(list(AMT=1000, RATE=NA, Time=0.5))
dosing_3 <- list(list(AMT=1000, Time=0.5))
dosing_4 <- list(list(AMT=c(1000,20), Time=c(0.5,10)))  # multiple dosing

df_12 <- model_prediction(design=design_3, DV=TRUE, dosing=dosing_1)
df_13 <- model_prediction(design=design_3, DV=TRUE, dosing=dosing_2)
df_14 <- model_prediction(design=design_3, DV=TRUE, dosing=dosing_3)
df_15 <- model_prediction(design=design_3, DV=TRUE, dosing=dosing_4)
df_16 <- model_prediction(design=design_3, DV=TRUE, dosing=dosing_4, filename="test.csv")

model_prediction(design=design_3, DV=TRUE, dosing=dosing_4, model_num_points = 10)
model_prediction(design=design_3, DV=TRUE, dosing=dosing_4, model_num_points = 10, model_minxt=20)

design_4 <- list(
  xt=c(0.5,1,2,6,24,36,72,120),
  model_switch=c(1,1,1,2,2,2,2),
  m=2,
  groupsize=3,
  a=list(c(WT=70,AGE=50), c(AGE=45,WT=60)))

model_prediction(design=design_4, DV=TRUE, dosing=dosing_4)
model_prediction(design=design_4, DV=TRUE, dosing=dosing_4, model_num_points = 10)
model_prediction(design=design_4, DV=TRUE, dosing=dosing_4, model_num_points = 10, model_minxt=10, model_maxxt=100)
model_prediction(design=design_4, DV=TRUE, dosing=dosing_4, model_num_points = 10, model_minxt=c(20,20), model_maxxt=c(100,100))
model_prediction(design=design_4, DV=TRUE, dosing=dosing_4, model_num_points = 10, model_minxt=c(20,20), model_maxxt=c(100,100))
ofv_criterion

Normalize an objective function by the size of the FIM matrix

Description

Compute a normalized OFV based on the size of the FIM matrix. This value can then be used in efficiency calculations. This is NOT the OFV used in optimization, see ofv_fim.

Usage

ofv_criterion(ofv_f, num_parameters, poped.db, ofv_calc_type = poped.db$settings$ofv_calc_type)

Arguments

ofv_f An objective function
num_parameters The number of parameters to use for normalization
poped.db a poped database
ofv_calc_type OFV calculation type for FIM
  • 1 = "D-optimality". Determinant of the FIM: det(FIM)
  • 2 = "A-optimality". Inverse of the sum of the expected parameter variances: 1/trace_matrix(inv(FIM))
  • 4 = "lnD-optimality". Natural logarithm of the determinant of the FIM: log(det(FIM))
  • 6 = "Ds-optimality". Ratio of the Determinant of the FIM and the Determinant of the uninteresting rows and columns of the FIM: det(FIM)/det(FIM_u)
  • 7 = Inverse of the sum of the expected parameter RSE: 1/sum(get_rse(FIM,poped.db,use_percent=FALSE))

Value

The specified criterion value.

See Also

Other FIM: LinMatrixH; LinMatrixLH; LinMatrixL_occ; calc_ofv_and_fim; ed_laplace_ofv; ed_mftot; evaluate.e.ofv.fim; evaluate_fim; gradf_eps; mf3; mf5; mf6; mf7; mf8; mftot0; mftot1; mftot2; mftot3; mftot4; mftot5; mftot6; mftot7; mftot; mf; ofv_fim

Examples

## Warfarin example from software comparison in:
## Nyberg et al., "Methods and software tools for design evaluation
## for population pharmacokinetics-pharmacodynamics studies",

## Optimization using an additive + proportional reidual error to
## avoid sample times at very low concentrations (time 0 or very late samoples).
library(PopED)
## find the parameters that are needed to define from the structural model

```
ff.PK.1.comp.oral.sd.CL
```

## parameter definition function

```r
# names match parameters in function ff
sfg <- function(x,a,bpop,bocc){
  parameters=c(CL=bpop[1]*exp(b[1]),
               V=bpop[2]*exp(b[2]),
               KA=bpop[3]*exp(b[3]),
               Favail=bpop[4],
               DOSE=a[1])
  return(parameters)
}
```

## Define initial design and design space

```r
# Create initial database
poped.db <- create.poped.database(ff_file="ff.PK.1.comp.oral.sd.CL",
                                   fg_file="sfg",
                                   ferror_file="feps.add.prop",
                                   bpop=c(CL=0.15, V=8, KA=1.0, Favail=1),
                                   notfixed_bpop=c(1,1,1,0),
                                   d=c(CL=0.07, V=0.02, KA=0.6),
                                   sigma=c(0.01,0.25),
                                   groupsize=32,
                                   xt=c( 0.5,1,2,6,24,36,72,120),
                                   minxt=0,
                                   maxxt=120,
                                   a=70,
                                   mina=0,
                                   maxa=100)
```

## evaluate initial design

```r
FIM <- evaluate.fim(poped.db) # new name for function needed
FIM
get_rse(FIM,poped.db)
```

```r
ofv_criterion(ofv_fim(FIM,poped.db,ofv_calc_type=1),
              length(get_unfixed_params(poped.db)["all"])),
              poped.db,
              ofv_calc_type=1) # det(FIM)
```

```r
ofv_criterion(ofv_fim(FIM,poped.db,ofv_calc_type=2),
              length(get_unfixed_params(poped.db)["all"])),
              poped.db,
              ofv_calc_type=2)
```

```r
ofv_criterion(ofv_fim(FIM,poped.db,ofv_calc_type=4),
              length(get_unfixed_params(poped.db)["all"])),
              poped.db,
              ofv_calc_type=4)
```

```r
ofv_criterion(ofv_fim(FIM,poped.db,ofv_calc_type=6),
              length(get_unfixed_params(poped.db)["all"])),
```
Evaluate a criterion of the Fisher Information Matrix (FIM)

Description

Compute a criterion of the FIM given the model, parameters, design and methods defined in the PopED database.

Usage

```r
ofv_fim(fmf, poped.db, ofv_calc_type = poped.db$settings$ofv_calc_type, ds_index = poped.db$params$ds_index, ...)
```

Arguments

- `fmf` The FIM
- `poped.db` A poped database
- `ofv_calc_type` OFV calculation type for FIM
  - 1 = "D-optimality". Determinant of the FIM: det(FIM)
  - 2 = "A-optimality". Inverse of the sum of the expected parameter variances: \(1/\text{trace}_{\text{matrix}}(\text{inv}(\text{FIM}))\)
  - 4 = "lnD-optimality". Natural logarithm of the determinant of the FIM: \(\log(\text{det}(\text{FIM}))\)
  - 6 = "Ds-optimality". Ratio of the Determinant of the FIM and the Determinant of the uninteresting rows and columns of the FIM: \(\text{det}(\text{FIM})/\text{det}(\text{FIM}_u)\)
  - 7 = Inverse of the sum of the expected parameter RSE: \(1/\text{sum}(\text{get_rse}(\text{FIM}, \text{poped.db}, \text{use_percent}=\text{FALSE}))\)
- `ds_index` `ds_index` is a vector set to 1 if a parameter is uninteresting, otherwise 0. Size=(1,num unfixed parameters). First unfixed bpop, then unfixed d, then unfixed docc and last unfixed sigma. Default is the fixed effects being important, everything else not important. Used in conjunction with `ofv_calc_type=6`.

... arguments passed to `evaluate.fim` and `ofv_fim`.

Value

The specified criterion value.
See Also

Other FIM: LinMatrixH; LinMatrixLH; LinMatrix_occ; calc_ofv_and_fim; ed_laplace_ofv; ed_mftot; evaluate.e_ofv.fim; evaluate.fim; gradfEPS; mf3; mf5; mf6; mf7; mf8; mftot0; mftot1; mftot2; mftot3; mftot4; mftot5; mftot6; mftot7; mf; ofv_criterion

Other evaluate_FIM: calc_ofv_and_fim; evaluate.e.ofv.fim; evaluate.fim

Examples

## Warfarin example from software comparison in:
## Nyberg et al., "Methods and software tools for design evaluation
## for population pharmacokinetics-pharmacodynamics studies",

## Optimization using an additive + proportional reidual error to
## avoid sample times at very low concentrations (time 0 or very late samples).
library(PopED)

## find the parameters that are needed to define from the structural model
ff.PK.1.comp.oral.sd.CL

## -- parameter definition function
## -- names match parameters in function ff
sfg <- function(x,a,bpop,bocc){
  parameters=c(CL=bpop[1]*exp(b[1]),
              V=bpop[2]*exp(b[2]),
              KA=bpop[3]*exp(b[3]),
              Favail=bpop[4],
              DOSE=a[1])
  return(parameters)
}

## -- Define initial design and design space
poped.db <- create.poped.database(ff_file="ff.PK.1.comp.oral.sd.CL",
                                   fg_file="sfg",
                                   fError_file="feps.add.prop",
                                   bpop=c(CL=0.15, V=8, KA=1.0, Favail=1),
                                   notfixed_bpop=c(1,1,1,0),
                                   d=c(CL=0.07, V=0.02, KA=0.6),
                                   sigma=c(0.01,0.25),
                                   groupsize=32,
                                   xt=c( 0.5,1,2,6,24,36,72,120),
                                   minxt=0,
                                   maxxt=120,
                                   a=78,
                                   mina=0,
                                   maxa=100)

## evaluate initial design
FIM <- evaluate.fim(poped.db) # new name for function needed
FIM
get_rse(FIM,poped.db)
ones

*Description*

Function creates a matrix of ones of size (dim1 x dim2). Written to match MATLAB's `ones` function.

*Usage*

```matlab
ones(dim1, dim2 = NULL)
```

*Arguments*

- **dim1**  
  The dimension of the matrix (if square) or the number of rows.
- **dim2**  
  The number of columns

*Value*

A matrix of ones

*See Also*

Other MATLAB: `cell; diag_matlab; feval; fileparts; isempty; isfield; randn; randperm; rand; size; tic; toc; zeros`

*Examples*

```matlab
ones(4)
ones(3, 4)
```
pargen  Parameter simulation

Description

Function generates random samples for a list of parameters

Usage

pargen(par, user_dist_pointer, sample_size, blhs, sample_number, poped.db)

Arguments

- **par**: A matrix describing the parameters. Each row is a parameter and the matrix has three columns:
  1. First column - Type of distribution (0-fixed, 1-normal, 2-uniform, 3-user specified, 4-lognormal, 5-Truncated normal).
  2. Second column - Mean of distribution.
  3. Third column - Variance or range of distribution.
- **user_dist_pointer**: A text string of the name of a function that generates random samples from a user defined distribution.
- **sample_size**: The number of random samples per parameter to generate.
- **blhs**: Logical, indicating if Latin Hypercube Sampling should be used.
- **sample_number**: The sample number to extract from a user distribution.
- **poped.db**: A PopED database.

Value

A matrix of random samples of size (sample_size x number_of_parameters)

Examples

```r
## Warfarin example from software comparison in:
## Nyberg et al., "Methods and software tools for design evaluation
## for population pharmacokinetics-pharmacodynamics studies",

## Optimization using an additive + proportional residual error to
## avoid sample times at very low concentrations (time 0 or very late samples).
library(PopED)

## find the parameters that are needed to define from the structural model
ff.PK.1.comp.oral.sd.CL

## -- parameter definition function
```
### Define initial design and design space

```r
poped.db <- create.poped.database(ff_file = "ff.PK.1.comp.oral.sd.CL", 
                                  fg_file = "sfg", 
                                  fError_file = "feps.add.prop", 
                                  bpop = c(CL = 0.15, V = 8, KA = 1.0, Favail = 1), 
                                  notfixed_bpop = c(1, 1, 1, 0), 
                                  d = c(CL = 0.07, V = 0.02, KA = 0.6), 
                                  sigma = c(0.01, 0.25), 
                                  groupsize = 32, 
                                  xt = c(0.5, 1, 2, 6, 24, 36, 72, 120), 
                                  minxt = 0, 
                                  maxxt = 120, 
                                  a = 70, 
                                  mina = 0, 
                                  maxa = 100)
```

# warfarin optimize model

# Adding 40% Uncertainty to fixed effects log-normal (not Favail)
```r
bpop_vals <- c(CL = 0.15, V = 8, KA = 1.0, Favail = 1) 
bbpop_vals_ed_ln <- cbind(ones(length(bpop_vals), 1)*4, # log-normal distribution 
                          bpop_vals, 
                          ones(length(bpop_vals), 1)*(bpop_vals*0.4)*2) # 40% of bpop value 
bbpop_vals_ed_ln["Favail",] <- c(0, 1, 0)
```

```r
pars-ln <- pargen(par = bpop_vals_ed_ln, 
                  user_dist_pointer = NULL, 
                  sample_size = 100, 
                  bLHS = 1, 
                  sample_number = NULL, 
                  poped.db)
```

# Adding 10% Uncertainty to fixed effects normal-distribution (not Favail)
```r
bpop_vals_ed_n <- cbind(ones(length(bpop_vals), 1)*1, # log-normal distribution 
                         bpop_vals, 
                         ones(length(bpop_vals), 1)*(bpop_vals*0.1)*2) # 10% of bpop value 
bbpop_vals_ed_n["Favail",] <- c(0, 1, 0)
```

```r
pars-n <- pargen(par = bpop_vals_ed_n, 
                 user_dist_pointer = NULL, 
                 sample_size = 100, 
                 bLHS = 1, 
                 sample_number = NULL,
```
Description

Function plots the efficiency of windows around the optimal design points. The maximal and minimal allowed values for all design variables as defined in poped.db are respected (e.g. poped.db$design_space$minxt and poped.db$design_space$maxxt).

Usage

\[
\text{plot\_efficiency\_of\_windows(poped.db, xt\_windows = \text{NULL, xt\_plus = xt\_windows, xt\_minus = xt\_windows, iNumSimulations = 100, y\_eff = T, y\_rse = T, ...)}
\]

Arguments

- **poped.db**: A poped database
- **xt\_windows**: The distance on one direction from the optimal sample times. Can be a number or a matrix of the same size as the xt matrix found in poped.db$design$xt.
- **xt\_plus**: The upper distance from the optimal sample times (xt + xt\_plus). Can be a number or a matrix of the same size as the xt matrix found in poped.db$design$xt.
- **xt\_minus**: The lower distance from the optimal sample times (xt - xt\_minus). Can be a number or a matrix of the same size as the xt matrix found in poped.db$design$xt.
- **iNumSimulations**: The number of design simulations to make within the specified windows.
- **y\_eff**: Should one of the plots created have efficiency on the y-axis?
- **y\_rse**: Should created plots include the relative standard error of each parameter as a value on the y-axis?
- **...**: Extra arguments passed to evaluate.fim
Value

A ggplot2 object.

See Also

Other Graphics: plot_model_prediction
Other Simulation: model_prediction; plot_model_prediction
Other evaluate_design: evaluate.fim; get_rse; model_prediction; plot_model_prediction

Examples

## Warfarin example from software comparison in:
## Nyberg et al., "Methods and software tools for design evaluation
## for population pharmacokinetics-pharmacodynamics studies",

library(PopED)

## find the parameters that are needed to define from the structural model ff.PK.1.comp.oral.sd.CL

## -- parameter definition function
## -- names match parameters in function ff
sfg <- function(x,a,bpop,b,bocc){
    parameters=c(CL=bpop[1]*exp(b[1]),
                 V=bpop[2]*exp(b[2]),
                 KA=bpop[3]*exp(b[3]),
                 Favail=bpop[4],
                 DOSE=a[1])
    return(parameters)
}

## -- Define initial design and design space
poped.db <- create.poped.database(ff_file="ff.PK.1.comp.oral.sd.CL",
                                   fg_file="sfg",
                                   fError_file="feps.prop",
                                   bpop=c(CL=0.15, V=8, KA=1.0, Favail=1),
                                   notfixed_bpop=c(1,1,1,0),
                                   d=c(CL=0.07, V=0.02, KA=0.6),
                                   sigma=0.01,
                                   groupsize=32,
                                   xt=c(0.5,1,2,6,24,36,72,120),
                                   minxt=0,
                                   maxxt=120,
                                   a=70)

# Examine efficiency of sampling windows
plot_efficiency_of_windows(poped.db,xt_windows=0.5)

plot_efficiency_of_windows(poped.db,
                           xt_plus=c(0.5,1,2,1,2,3,7,1),
plot_model_prediction

xt_minus = c(0.1, 2, 5, 4, 2, 3, 6, 2))

## Not run:

plot_efficiency_of_windows(poped.db, xt_windows = c(0.5, 1, 2, 1, 2, 3, 7, 1))

plot_efficiency_of_windows(poped.db,
                         xt_plus = c(0.5, 1, 2, 1, 2, 3, 7, 1),
                         xt_minus = c(0.1, 2, 5, 4, 2, 3, 6, 2),
                         y_rse = FALSE)

plot_efficiency_of_windows(poped.db,
                         xt_plus = c(0.5, 1, 2, 1, 2, 3, 7, 1),
                         xt_minus = c(0.1, 2, 5, 4, 2, 3, 6, 2),
                         y_eff = FALSE)

## End(Not run)

---

plot_model_prediction  Plot model predictions

Description

Function plots model predictions for the typical value in the population, individual predictions and data predictions.

Usage

plot_model_prediction(poped.db, model_num_points = 100, separate.groups = F,
                       sample.times = T, sample.times.IPRED = F, sample.times.DV = F,
                       PRED = T, IPRED = F, IPRED.lines = F, IPRED.lines.pctls = F,
                       alpha.IPRED.lines = 0.1, alpha.IPRED = 0.3, sample.times.size = 4,
                       DV = F, alpha.DV = 0.3, DV.lines = F, DV.points = F,
                       alpha.DV.lines = 0.3, alpha.DV.points = 0.3, sample.times.DV.points = F,
                       sample.times.DV.lines = F, alpha.sample.times.DV.points = 0.3,
                       alpha.sample.times.DV.lines = 0.3, y_lab = "Model Predictions",
                       facet_scales = "fixed", facet_label_names = T, ...)

Arguments

poped.db  A PopED database.
model_num_points

How many extra observation rows should be created in the data frame for each group or individual per model. If used then the points are placed evenly between model_minxt and model_maxxt. This option is used by plot_model_prediction to simulate the response of the model on a finer grid than the defined design. If NULL then only the input design is used. Can be a single value or a vector the same length as the number of models.
plot_model_prediction

separate.groups
Should there be separate plots for each group.
sample.times
Should sample times be shown on the plots.
sample.times.IPRED
Should sample times be shown based on the IPRED y-values.
sample.times.DV
Should sample times be shown based on the DV y-values.
PRED
Should a PRED line be drawn.
IPRED
Should we simulate individual predictions?
IPRED.lines
Should IPRED lines be drawn?
IPRED.lines.pctils
Should lines be drawn at the chosen percentiles of the IPRED values?
alpha.IPRED.lines
What should the transparency for the IPRED.lines be?
alpha.IPRED
What should the transparency of the IPRED CI be?
sample.times.size
What should the size of the sample.times be?
DV
Should we simulate observations?
alpha.DV
What should the transparency of the DV CI be?
DV.lines
Should DV lines be drawn?
DV.points
Should DV points be drawn?
alpha.DV.lines
What should the transparency for the DV.lines be?
alpha.DV.points
What should the transparency for the DV.points be?
sample.times.DV.points
TRUE or FALSE.
sample.times.DV.lines
TRUE or FALSE.
alpha.sample.times.DV.points
What should the transparency for the sample.times.DV.points be?
alpha.sample.times.DV.lines
What should the transparency for the sample.times.DV.lines be?
y_lab
The label of the y-axis.
facet_scales
Can be "free", "fixed", "free_x" or "free_y"
facet_label_names
TRUE or FALSE
... arguments passed to evaluate.fim and ofv_fim.

Value
A ggplot2 object.
See Also

Other Graphics: plot_efficiency_of_windows
Other Simulation: model_prediction; plot_efficiency_of_windows
Other evaluate_design: evaluate.fim; get_rse; model_prediction; plot_efficiency_of_windows

Examples

## Warfarin example from software comparison in:
## Nyberg et al., "Methods and software tools for design evaluation
## for population pharmacokinetics-pharmacodynamics studies",

library(PopED)

## find the parameters that are needed to define from the structural model
ff.PK.1.comp.oral.md.CL

## -- parameter definition function
## -- names match parameters in function ff
sfg <- function(x,a,bpop,bocc){
  parameters=c(CL=bpop[[1]]*exp(b[1]),
             V=bpop[[2]]*exp(b[2]),
             KA=bpop[[3]]*exp(b[3]),
             Favail=bpop[[4]],
             DOSE=a[1])
  return(parameters)
}

## -- Define initial design and design space
poped.db <- create.poped.database(ff_file="ff.PK.1.comp.oral.sd.CL",
                                   fg_file="sfg",
                                   fError_file="feps.prop",
                                   bpop=c(CL=0.15, V=8, KA=1.0, Favail=1),
                                   notfixed_bpop=c(1,1,1,0),
                                   d=c(CL=0.07, V=0.02, KA=0.6),
                                   sigma=0.01,
                                   groupsize=32,
                                   xt=c( 0.5,1,2,6,24,36,72,120),
                                   minxt=0,
                                   maxxt=120,
                                   a=70)

## create plot of model without variability
plot_model_prediction(poped.db)

## create plot of model with variability
plot_model_prediction(poped.db,IPRED=TRUE,DV=TRUE)
**poped_choose**  
*Choose between arg1 and arg2*

**Description**

Function chooses arg1 unless it is NULL in which case arg2 is chosen.

**Usage**

```plaintext
poped.choose(arg1, arg2)
```

**Arguments**

- **arg1**  
The first argument
- **arg2**  
The second argument

**See Also**

Other `poped_input`: `convert_variables`; `create.poped.database`; `create_design_space`; `create_design`; `downsizing_general_design`

**Examples**

```plaintext
poped.choose(2,5)
poped.choose("foo",66)
poped.choose(NULL,"hello")
```

---

**poped_optimize**  
*Optimization main module for PopED*

**Description**

Optimize the objective function. The function works for both discrete and continuous optimization variables. This function takes information from the PopED database supplied as an argument. The PopED database supplies information about the model, parameters, design and methods to use. Some of the arguments coming from the PopED database can be overwritten; if they are supplied then they are used instead of the arguments from the PopED database.
Usage

poped_optimize(poped.db, ni = NULL, xt = NULL, model_switch = NULL,
   x = NULL, a = NULL, bpop = NULL, d = NULL, maxxt = NULL,
   minxt = NULL, maxa = NULL, mina = NULL, fmf = 0, dmf = 0,
   trflag = TRUE, opt_xt = poped.db$settings$optsw[2],
   opt_a = poped.db$settings$optsw[4], opt_x = poped.db$settings$optsw[3],
   opt_samps = poped.db$settings$optsw[1],
   opt_inds = poped.db$settings$optsw[5], cfaxt = poped.db$settings$cfaxt,
   cfaa = poped.db$settings$cfaa, rsit = poped.db$settings$rsit,
   rsit_output = poped.db$settings$rsit_output,
   fimNcalcNtype = poped.db$settings$ifIMCalculationType,
   ofv_calc_type = poped.db$settings$ofv_calc_type,
   approx_type = poped.db$settings$approximationMethod,
   bUseExchangeAlgorithm = poped.db$settings$bUseExchangeAlgorithm, iter = 1,
   d_switch = poped.db$settings$d_switch,
   ED_samp_size = poped.db$settings$ED_samp_size,
   bLHS = poped.db$settings$bLHS,
   use_laplace = poped.db$settings$useLaplace, ...
)

Arguments

poped.db  A PopED database.

ni  A vector of the number of samples in each group.

xt  A matrix of sample times. Each row is a vector of sample times for a group.

model_switch  A matrix that is the same size as xt, specifying which model each sample belongs to.

x  A matrix for the discrete design variables. Each row is a group.

a  A matrix of covariates. Each row is a group.

bpop  Matrix defining the fixed effects. per row (row number = parameter number) we should have:
  • column 1 the type of the distribution for E-family designs (0 = Fixed, 1 = Normal, 2 = Uniform, 3 = User Defined Distribution, 4 = lognormal and 5 = truncated normal)
  • column 2 defines the mean.
  • column 3 defines the variance of the distribution (or length of uniform distribution).

Can also just supply the parameter values as a vector c()

d  Matrix defining the diagonals of the IIV (same logic as for the fixed effects). can also just supply the parameter values as a c().

maxxt  Matrix or single value defining the maximum value for each xt sample. If a single value is supplied then all xt values are given the same maximum value.

minxt  Matrix or single value defining the minimum value for each xt sample. If a single value is supplied then all xt values are given the same minimum value.

maxa  Vector defining the max value for each covariate. IF a single value is supplied then all a values are given the same max value.
<table>
<thead>
<tr>
<th>Variable</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>mina</code></td>
<td>Vector defining the min value for each covariate. If a single value is supplied then all values are given the same max value.</td>
</tr>
<tr>
<td><code>fimf</code></td>
<td>The initial value of the FIM. If set to zero then it is computed.</td>
</tr>
<tr>
<td><code>dfm</code></td>
<td>The initial OFV. If set to zero then it is computed.</td>
</tr>
<tr>
<td><code>trflag</code></td>
<td>Should the optimization be output to the screen and to a file?</td>
</tr>
<tr>
<td><code>opt_xt</code></td>
<td>Should the sample times be optimized?</td>
</tr>
<tr>
<td><code>opt_a</code></td>
<td>Should the continuous design variables be optimized?</td>
</tr>
<tr>
<td><code>opt_x</code></td>
<td>Should the discrete design variables be optimized?</td>
</tr>
<tr>
<td><code>opt_samps</code></td>
<td>Are the number of sample times per group being optimized?</td>
</tr>
<tr>
<td><code>opt_inds</code></td>
<td>Are the number of individuals per group being optimized?</td>
</tr>
<tr>
<td><code>cfaa</code></td>
<td>Stochastic Gradient search first step factor for covariates</td>
</tr>
<tr>
<td><code>rsit</code></td>
<td>Number of Random search iterations</td>
</tr>
<tr>
<td><code>rsit_output</code></td>
<td>Number of iterations in random search between screen output</td>
</tr>
<tr>
<td><code>fim.calc.type</code></td>
<td>The method used for calculating the FIM. Potential values:</td>
</tr>
<tr>
<td></td>
<td>• 0 = Full FIM. No assumption that fixed and random effects are uncorrelated. See <code>mftot0</code>.</td>
</tr>
<tr>
<td></td>
<td>• 1 = Reduced FIM. Assume that there is no correlation in the FIM between the fixed and random effects, and set these elements in the FIM to zero. See <code>mftot1</code>.</td>
</tr>
<tr>
<td></td>
<td>• 2 = Weighted models (placeholder).</td>
</tr>
<tr>
<td></td>
<td>• 3 = Not currently used.</td>
</tr>
<tr>
<td></td>
<td>• 4 = Reduced FIM and computing all derivatives with respect to the standard deviation of the residual unexplained variation (sqrt(SIGMA) in NONMEM). This matches what is done in PFIM, and assumes that the standard deviation of the residual unexplained variation is the estimated parameter (NOTE: NONMEM estimates the variance of the residual unexplained variation by default). See <code>mftot4</code>.</td>
</tr>
<tr>
<td></td>
<td>• 5 = Full FIM parameterized with A,B,C matrices &amp; derivative of variance. See <code>mftot5</code>.</td>
</tr>
<tr>
<td></td>
<td>• 6 = Calculate one model switch at a time, good for large matrices. See <code>mftot6</code>.</td>
</tr>
<tr>
<td></td>
<td>• 7 = Reduced FIM parameterized with A,B,C matrices &amp; derivative of variance. See <code>mftot7</code>.</td>
</tr>
<tr>
<td><code>ofv_calc.type</code></td>
<td>OFV calculation type for FIM</td>
</tr>
<tr>
<td></td>
<td>• 1 = &quot;D-optimality&quot;. Determinant of the FIM: det(FIM)</td>
</tr>
<tr>
<td></td>
<td>• 2 = &quot;A-optimality&quot;. Inverse of the sum of the expected parameter variances: 1/trace_matrix(inv(FIM))</td>
</tr>
<tr>
<td></td>
<td>• 4 = &quot;lnD-optimality&quot;. Natural logarithm of the determinant of the FIM: log(det(FIM))</td>
</tr>
<tr>
<td></td>
<td>• 6 = &quot;Ds-optimality&quot;. Ratio of the Determinant of the FIM and the Determinant of the uninteresting rows and columns of the FIM: det(FIM)/det(FIM_u)</td>
</tr>
</tbody>
</table>
• 7 = Inverse of the sum of the expected parameter RSE: 1/sum(get_rse(FIM,poped.db,use_percent=FALSE))

approx_type
Approximation method for model, 0=FO, 1=FOCE, 2=FOCEI, 3=FOI.

bUseExchangeAlgorithm
Use Exchange algorithm (1=TRUE, 0=FALSE)

iter
The number of iterations entered into the blockheader function.

d_switch

• *****START OF CRITERION SPECIFICATION OPTIONS**********

D-family design (1) or ED-family design (0) (with or without parameter uncertainty)

ED_samp_size
Sample size for E-family sampling

bLHS
How to sample from distributions in E-family calculations. 0=Random Sampling, 1=LatinHyperCube –

use_laplace
Should the Laplace method be used in calculating the expectation of the OFV?

... arguments passed to other functions. See Doptim.

References


See Also

Other Optimize: Doptim; LEDoptim; RS_opt_gen; RS_opt; a_line_search; bfgsb_min; calc_autofocus; calc_ofv_and_grad; mfea

Examples

## Warfarin example from software comparison in:
## Nyberg et al., "Methods and software tools for design evaluation
## for population pharmacokinetics-pharmacodynamics studies",

## Optimization using an additive + proportional reidual error to
## avoid sample times at very low concentrations (time 0 or very late samoples).
library(PopED)

## find the parameters that are needed to define from the structural model
ff.PK.1.comp.oral.sd.CL

## -- parameter definition function
## -- names match parameters in function ff
sfg <- function(x,a,bpop,b,bocc){
  parameters=c(CL=bpop[1]*exp(b[1]),
              V=bpop[2]*exp(b[2]),
              KA=bpop[3]*exp(b[3]),
...
Favail=bpop[4],
DOSE=a[1])
return(parameters)
}

## -- Define initial design and design space
poped.db <- create.poped.database(ff_file="ff_PK.1.comp.oral.sd.CL",
fG_file="sfg",
fError_file="feps.add.prop",
bpop=cb(CL=0.15, V=8, KA=1.0, Favail=1),
notfixed_bpop=cb(1,1,1,0),
d=cb(CL=0.07, V=0.02, KA=0.6),
sigma=cb(0.01,0.25),
groupsize=32,
x=cb(0.5,1,2,6,24,36,72,120),
minx=0,
maxx=120,
a=70,
mina=0,
maxa=100)

# warfarin optimize model

## Not run:

#################
# D-family Optimization
#################

# below are a number of ways to optimize the problem

# RS+SG+LS optimization of sample times
# optimization with just a few iterations
# only to check that things are working
output <- poped_optimize(poped.db,opt_xt=T,
rsit=5,sgit=5,ls_step_size=5)

# RS+SG+LS optimization of sample times
# (longer run time than above but more likely to reach a maximum)
output <- poped_optimize(poped.db,opt_xt=T)
get_rse(output$fmf,output$poped.db)
plot_model_prediction(output$poped.db)

# MFEA optimization with only integer times allowed
mfea.output <- poped_optimize(poped.db,opt_xt=1,
biUseExchangeAlgorithm=1,
EAStepSize=1)
get_rse(mfea.output$fmf,mfea.output$poped.db)
plot_model_prediction(mfea.output$poped.db)

# Examine efficiency of sampling windows
plot_efficiency_of_windows(mfea.output$poped.db,xt_windows=0.5)
plot_efficiency_of_windows(mfea.output$poped.db,xt_windows=1)
# Random search (just a few samples here)
rs.output <- poped_optimize(poped.db_opt_xt=1, opt_a=1, rsit=20,
  bUseRandomSearch= 1,
  bUseStochasticGradient = 0,
  bUseBFGSMinimizer = 0,
  bUseLineSearch = 0)
get_rse(rs.output$mfm, rs.output$poped.db)

# line search, DOSE and sample time optimization
ls.output <- poped_optimize(poped.db_opt_xt=1, opt_a=1,
  bUseRandomSearch= 0,
  bUseStochasticGradient = 0,
  bUseBFGSMinimizer = 0,
  bUseLineSearch = 1,
  ls_step_size=10)

# Stochastic gradient search, DOSE and sample time optimization
sg.output <- poped_optimize(poped.db_opt_xt=1, opt_a=1,
  bUseRandomSearch= 0,
  bUseStochasticGradient = 1,
  bUseBFGSMinimizer = 0,
  bUseLineSearch = 0,
  sgit=20)

# BFGS search, DOSE and sample time optimization
bfgs.output <- poped_optimize(poped.db_opt_xt=1, opt_a=1,
  bUseRandomSearch= 0,
  bUseStochasticGradient = 0,
  bUseBFGSMinimizer = 1,
  bUseLineSearch = 0)

# E-family Optimization

# Adding 10% log-normal Uncertainty to fixed effects (not Favail)
bpop_vals <- c(CL=0.15, V=8, KA=1.0, Favail=1)
bpop_vals_ed_ln <- cbind(ones(length(bpop_vals),1)*4, # log-normal distribution
  bpop_vals,
  ones(length(bpop_vals),1)*(bpop_vals*0.1)^2) # 10% of bpop value
bpop_vals_ed_ln

# -- Define initial design and design space
poped.db <- create.poped.database(ff_file="ff_PK.1.comp.oral.sd.CL",
  fg_file="sfg",
  fError_file="feps.add.prop",
  bpop=bpop_vals_ed_ln,
  notfixed_bpop=c(1,1,0,
  d=c(CL=0.07, V=0.02, KA=0.6),
  sigma=c(0.01,0.25),
Function written to match MATLAB’s rand function

Description
Generate random samples from a uniform distribution [0,1] and return in matrix form

Usage
rand(dim1, dim2 = NULL)

Arguments
- dim1: The dimension of the matrix (if square), otherwise the number of rows.
- dim2: The number of columns, if different from the number of rows.

Value
Matrix of random generated samples.

See Also
Other MATLAB: cell; diag_matlab; feval; fileparts; isempty; isfield; ones; randn; randperm; size; tic; toc; zeros
Examples

randn(2,3)

randn(5)

randn

Function written to match MATLAB’s randn function

Description

Generate random samples from a standardized normal distribution and return in matrix form.

Usage

randn(dim1, dim2 = NULL)

Arguments

- dim1: The dimension of the matrix (if square), otherwise the number of rows.
- dim2: The number of columns, if different from the number of rows.

Value

Matrix of random generated samples.

See Also

Other MATLAB: cell; diag_matlab; feval; fileparts; isempty; isfield; ones; randperm; rand; size; tic; toc; zeros

Examples

randn(2,3)

randn(5)
rs_opt

Optimize the objective function using an adaptive random search algorithm for D-family designs.

Description

Optimize the objective function using an adaptive random search algorithm. The function works for both discrete and continuous optimization variables. This function takes information from the PopED database supplied as an argument. The PopED database supplies information about the model, parameters, design and methods to use. Some of the arguments coming from the PopED database can be overwritten; by default these arguments are NULL in the function, if they are supplied then they are used instead of the arguments from the PopED database.

randperm

Function written to match MATLAB’s randperm function

Description

A wrapper for the sample function.

Usage

randperm(num)

Arguments

num Either a vector of one or more elements from which to choose, or a positive integer.

Value

See sample

See Also

Other MATLAB: cell; diag_matlab; feval; fileparts; isempty; isfield; ones; randn; rand; size; tic; toc; zeros

Examples

randperm(c(2,3,4,5,6))

randperm(10)
Usage

RS_opt(poped.db, ni = NULL, xt = NULL, model_switch = NULL, x = NULL,  
a = NULL, bpopdescr = NULL, ddescr = NULL, maxxt = NULL, 
minxt = NULL, maxa = NULL, mina = NULL, fmf = 0, dmf = 0, 
trflag = TRUE, opt_xt = poped.db$settings$optsw[2], 
opt_a = poped.db$settings$optsw[4], opt_x = poped.db$settings$optsw[3], 
cf = poped.db$settings$cf, cfa = poped.db$settings$cfa, 
rsit = poped.db$settings$rsit,  
rsit_output = poped.db$settings$rsit_output, 
fim.calc.type = poped.db$settings$fIMCalculationType, 
approx_type = poped.db$settings$ApproximationMethod, iter = 1, ...)

Arguments

poped.db A PopED database.

ni A vector of the number of samples in each group.

xt A matrix of sample times. Each row is a vector of sample times for a group.

model_switch A matrix that is the same size as xt, specifying which model each sample belongs to.

x A matrix for the discrete design variables. Each row is a group.

a A matrix of covariates. Each row is a group.

bpopdescr Matrix defining the fixed effects, per row (row number = parameter_number) we should have:

• column 1 the type of the distribution for E-family designs (0 = Fixed, 1 = Normal, 2 = Uniform, 3 = User Defined Distribution, 4 = lognormal and 5 = truncated normal)
• column 2 defines the mean.
• column 3 defines the variance of the distribution (or length of uniform distribution).

ddescr Matrix defining the diagonals of the IIV (same logic as for the bpopdescr).

maxxt Matrix or single value defining the maximum value for each xt sample. If a single value is supplied then all xt values are given the same maximum value.

minxt Matrix or single value defining the minimum value for each xt sample. If a single value is supplied then all xt values are given the same minimum value.

maxa Vector defining the max value for each covariate. IF a single value is supplied then all a values are given the same max value

mina Vector defining the min value for each covariate. IF a single value is supplied then all a values are given the same min value

fmf The initial value of the FIM. If set to zero then it is computed.

dmf The initial OFV. If set to zero then it is computed.

trflag Should the optimization be output to the screen and to a file?

opt_xt Should the sample times be optimized?

opt_a Should the continuous design variables be optimized?
opt_x Should the discrete design variables be optimized?
cfxt First step factor for sample times
cfaa Stochastic Gradient search first step factor for covariates
rsit Number of Random search iterations
rsit_output Number of iterations in random search between screen output
fim.calc.type The method used for calculating the FIM. Potential values:
  • 0 = Full FIM. No assumption that fixed and random effects are uncorrelated. See mftot0.
  • 1 = Reduced FIM. Assume that there is no correlation in the FIM between the fixed and random effects, and set these elements in the FIM to zero. See mftot1.
  • 2 = weighted models (placeholder).
  • 3 = Not currently used.
  • 4 = Reduced FIM and computing all derivatives with respect to the standard deviation of the residual unexplained variation (sqrt(SIGMA) in NONMEM). This matches what is done in PFIM, and assumes that the standard deviation of the residual unexplained variation is the estimated parameter (NOTE: NONMEM estimates the variance of the residual unexplained variation by default). See mftot4.
  • 5 = Full FIM parameterized with A,B,C matrices & derivative of variance. See mftot5.
  • 6 = Calculate one model switch at a time, good for large matrices. See mftot6.
  • 7 = Reduced FIM parameterized with A,B,C matrices & derivative of variance. See mftot7.
approx_type Approximation method for model, 0=FO, 1=FOCE, 2=FOCEI, 3=FOI.
iter The number of iterations entered into the blockheader function.
... arguments passed to evaluate.fim and ofv_fim.

References


See Also

Other Optimize: Doptim; LEDoptim; RS_opt_gen; a_line_search; bfgsb_min; calc Autofocus; calc_ofv_and_grad; mfea; poped_optimize
Examples

```r
## Warfarin example from software comparison in:
## Nyberg et al., "Methods and software tools for design evaluation
## for population pharmacokinetics-pharmacodynamics studies",

## Optimization using an additive + proportional residual error to
## avoid sample times at very low concentrations (time 0 or very late samples).
library(PopED)

## find the parameters that are needed to define from the structural model
ff.PK.1.comp.oral.sd.CL

## -- parameter definition function
## -- names match parameters in function ff
sfg <- function(x,a,bpop,bocc){
  parameters=c(CL=bpop[1], V=bpop[2], KA=bpop[3], Favail=bpop[4], DOSE=a[1])
  return(parameters)
}

## -- Define initial design and design space
poped.db <- create.poped.database(ff_file="ff.PK.1.comp.oral.sd.CL",
                                    fg_file="sfg",
                                    fError_file="feps.add.prop",
                                    bpop=c(CL=0.15, V=8, KA=1.0, Favail=1),
                                    notfixed_bpop=c(1,1,1,0),
                                    d=c(CL=0.07, V=0.02, KA=0.6),
                                    sigma=c(0.01,0.25),
                                    groupsize=32,
                                    xt=c(0.5,1,2,6,24,36,72,120),
                                    minxt=0,
                                    maxxt=120,
                                    a=70,
                                    mina=0,
                                    maxa=100)

# warfarin optimize model

# Just a few iterations, optimize on DOSE and sample times using the full FIM
rs.output <- RS_opt(poped.db, opt_xt=1, opt_a=1, rsit=20, fim.calc.type=0)

## Not run:

## More iterations
rs.output <- RS_opt(poped.db)

## End(Not run)
```
RS_opt_gen

**Optimize the objective function using an adaptive random search algorithm for D-family and E-family designs.**

**Description**

Optimize the objective function using an adaptive random search algorithm. Optimization can be performed for both D-family and E-family designs. The function works for both discrete and continuous optimization variables. This function takes information from the PopED database supplied as an argument. The PopED database supplies information about the model, parameters, design and methods to use. Some of the arguments coming from the PopED database can be overwritten; by default these arguments are NULL in the function, if they are supplied then they are used instead of the arguments from the PopED database.

**Usage**

```r
to use this function call RS_opt_gen with poped.db, ni = NULL, xt = NULL, model_switch = NULL,
  x = NULL, a = NULL, bpopdescr = NULL, ddescr = NULL, maxxt = NULL,
  minxt = NULL, maxa = NULL, mina = NULL, fm = NULL, dmf = NULL,
  trflag = TRUE, opt_xt = poped.db$settings$optsw[2],
  opt_a = poped.db$settings$optsw[4], opt_x = poped.db$settings$optsw[3],
  cfa = poped.db$settings$cfa, cfaa = poped.db$settings$cfaa,
  rsit = poped.db$settings$rsit,
  rsit_output = poped.db$settings$rsit_output,
  fim.calc.type = poped.db$settings$fIMCalculationType,
  approx_type = poped.db$settings$approximationMethod, iter = NULL,
  d_switch = poped.db$settings$d_switch,
  use_laplace = poped.db$settings$laplaceCalculationType, laplace.fim = FALSE,
  header_flag = TRUE, footer_flag = TRUE, out_file = NULL,
  compute_inv = TRUE, ...
```

**Arguments**

- `poped.db`: A PopED database.
- `ni`: A vector of the number of samples in each group.
- `xt`: A matrix of sample times. Each row is a vector of sample times for a group.
- `model_switch`: A matrix that is the same size as `xt`, specifying which model each sample belongs to.
- `x`: A matrix for the discrete design variables. Each row is a group.
- `a`: A matrix of covariates. Each row is a group.
- `bpopdescr`: Matrix defining the fixed effects, per row (row number = parameter_number) we should have:
  - column 1 the type of the distribution for E-family designs (0 = Fixed, 1 = Normal, 2 = Uniform, 3 = User Defined Distribution, 4 = lognormal and 5 = truncated normal)
• column 2 defines the mean.
• column 3 defines the variance of the distribution (or length of uniform distribution).

**ddescr** Matrix defining the diagonals of the IIV (same logic as for the bpopdescr).

**maxxt** Matrix or single value defining the maximum value for each xt sample. If a single value is supplied then all xt values are given the same maximum value.

**minxt** Matrix or single value defining the minimum value for each xt sample. If a single value is supplied then all xt values are given the same minimum value.

**maxa** Vector defining the max value for each covariate. IF a single value is supplied then all a values are given the same max value.

**mina** Vector defining the min value for each covariate. IF a single value is supplied then all a values are given the same max value.

**fmf** The initial value of the FIM. If set to zero then it is computed.

**dmf** The initial OFV. If set to zero then it is computed.

**trflag** Should the optimization be output to the screen and to a file?

**opt_xt** Should the sample times be optimized?

**opt_a** Should the continuous design variables be optimized?

**opt_x** Should the discrete design variables be optimized?

**cfaxt** First step factor for sample times

**cfaa** Stochastic Gradient search first step factor for covariates

**rsit** Number of Random search iterations

**rsit_output** Number of iterations in random search between screen output

**fim.ccall.type** The method used for calculating the FIM. Potential values:

• 0 = Full FIM. No assumption that fixed and random effects are uncorrelated. See mftot0.

• 1 = Reduced FIM. Assume that there is no correlation in the FIM between the fixed and random effects, and set these elements in the FIM to zero. See mftot1.

• 2 = weighted models (placeholder).

• 3 = Not currently used.

• 4 = Reduced FIM and computing all derivatives with respect to the standard deviation of the residual unexplained variation (sqrt(SIGMA) in NONMEM). This matches what is done in PFIM, and assumes that the standard deviation of the residual unexplained variation is the estimated parameter (NOTE: NONMEM estimates the variance of the residual unexplained variation by default). See mftot4.

• 5 = Full FIM parameterized with A,B,C matrices & derivative of variance. See mftot5.

• 6 = Calculate one model switch at a time, good for large matrices. See mftot6.

• 7 = Reduced FIM parameterized with A,B,C matrices & derivative of variance See mftot7.
approx_type  Approximation method for model, 0=FO, 1=FOCE, 2=FOCEI, 3=FOI.
iter           The number of iterations entered into the blockheader_2 function.
d_switch       • ******START OF CRITERION SPECIFICATION OPTIONS***********
                D-family design (1) or ED-family design (0) (with or without parameter uncer-
use_laplace    tainty)
laplace.fim     Should an E(FIM) be calculated when computing the Laplace approximated
                E(OFV). Typically the FIM does not need to be computed and, if desired, this
                calculation is done using the standard MC integration technique, so can be slow.
header_flag     Should the header text be printed out?
footer_flag     Should the footer text be printed out?
out_file        Which file should the output be directed to? A string, a file handle using file
                or "" will output to the screen.
compute_inv     should the inverse of the FIM be used to compute expected RSE values? Often
                not needed except for diagnostic purposes.
                ... arguments passed to evaluate.fim and ofv.fim.

References

1. M. Foracchia, A.C. Hooker, P. Vicini and A. Ruggeri, "PopED, a software for optimal experi-
   mental design in population kinetics", Computer Methods and Programs in Biomedicine, 74,
   2004.

   An extended, parallelized, nonlinear mixed effects models optimal design tool", Computer

See Also

Other Optimize: Doptim; LEDoptim; RS_opt; a_line_search; bfgsb_min; calc Autofocus;
calc_ofv_and_grad; mfea; poped_optimize

Examples

## Warfarin example from software comparison in:
## Nyberg et al., "Methods and software tools for design evaluation
## for population pharmacokinetics-pharmacodynamics studies",

## Optimization using an additive + proportional residual error to
## avoid sample times at very low concentrations (time 0 or very late samples).
library(PopED)

## find the parameters that are needed to define from the structural model
ff.PK.1.comp.oral.sd.CL

## -- parameter definition function
## -- names match parameters in function ff
RS_opt_gen

sfg <- function(x, a_bpop, b_bocc){
  parameters=c(CL=bpop[1]*exp(b[1]),
              V=bpop[2]*exp(b[2]),
              KA=bpop[3]*exp(b[3]),
              Favail=bpop[4],
              DOSE=a[1])
  return(parameters)
}

# Adding 10% log-normal Uncertainty to fixed effects (not Favail)
bpop_vals <- c(CL=0.15, V=8, KA=1.0, Favail=1)
bpop_vals_ed_ln <- cbind(ones(length(bpop_vals),1)*4, # log-normal distribution
                         bpop_vals,
                         ones(length(bpop_vals),1)*(bpop_vals*0.1)^2) # 10% of bpop value
bpop_vals_ed_ln["Favail",] <- c(0,1,0)
bpop_vals_ed_ln

## -- Define initial design and design space
poped.db <- create.poped.database(ff_file="ff.PK.1.comp.oral.sd.CL",
                               fg_file="sfg",
                               fError_file="feps.add.prop",
                               bpop=bpop_vals_ed_ln,
                               notfixed_bpop=c(1,1,1,0),
                               d=c(CL=0.07, V=0.02, KA=0.6),
                               sigma=c(0.01,0.25),
                               groupsize=32,
                               xt=c(0.5,1,2,6,24,36,72,120),
                               minxt=0,
                               maxxt=120,
                               a=70,
                               mina=0,
                               maxa=100)

# warfarin ed model

# Just a few iterations, optimize on sample times
output <- RS_opt_gen(poped.db, opt_xt=TRUE, rsit=20)

# Just a few iterations, optimize on DOSE and sample times using the full FIM
output <- RS_opt_gen(poped.db, opt_xt=1, opt_a=1, rsit=20, fim.calc.type=0)

## Not run:
RS_opt_gen(poped.db)

RS_opt_gen(poped.db, opt_xt=TRUE, rsit=100, compute_inv=F)
RS_opt_gen(poped.db, opt_xt=TRUE, rsit=20, d_switch=0)
RS_opt_gen(poped.db, opt_xt=TRUE, rsit=10, d_switch=0, use_laplace=T)
RS_opt_gen(poped.db, opt_xt=TRUE, rsit=10, d_switch=0, use_laplace=T, laplace.fim=T)

## Different headers and footers of output
RS_opt_gen(poped.db, opt_xt=TRUE, rsit=10, out_file="foo.txt")
output <- RS_opt_gen(poped.db, opt_xt=TRUE, rsit=100, trflag=FALSE)
RS_opt_gen(poped.db, opt_xt=TRUE, rsit=10, out_file="")
size

Function written to match MATLAB’s size function

Description

Function written to match MATLAB’s size function

Usage

```
size(obj, dimension.index = NULL)
```

Arguments

- **obj**: An object you want to know the various dimensions of. Typically a matrix.
- **dimension.index**: Which dimension you are interested in.

Value

The dimensions of the object or specific dimension you are interested in.

See Also

Other MATLAB: cell; diag_matlab; feval; fileparts; isempty; isfield; ones; randn; randperm; rand; tic; toc; zeros

Examples

```
size(c(2,3,4,5,6))
size(10)
size(zeros(4,7))
```
Description

Function tests is any matrix element is above a maximum value. For those elements the function sets those values to the maximum value.

Usage

test_for_max(mat, max_mat)

Arguments

mat A matrix.
max_mat A matrix the same size as mat with the maximum allowed value of that element.

Value

A matrix

See Also

Other matrix_manipulation: diag_matlab; test_for_min; test_for_zeros

Examples

test_for_max(cbind(2,3,4,5,6),cbind(4,4,4,4,4))
test_for_max(ones(6)\*45,ones(6)\*40)

Description

Function tests is any matrix element is above a minimum value. For those elements the function sets those values to the minimum value.

Usage

test_for_min(mat, min_mat)
**test_for_zeros**

**Arguments**
- **mat**: A matrix.
- **min_mat**: A matrix the same size as mat with the minimum allowed value of that element.

**Value**
A matrix

**See Also**
Other matrix_manipulation: `diag_matlab`; `test_for_max`; `test_for_zeros`

**Examples**

```r
test_for_min(cbind(2,3,4,5,6),cbind(4,4,4,4))
test_for_min(ones(6)*45,ones(6)*40)
```

---

**Description**
Function tests is any matrix element is zero. For those elements the function sets those values to the minimum value allowed (not zero). This is used to avoid numerical problems in the FIM calculation.

**Usage**

```r
test_for_zeros(mat, ourzero)
```

**Arguments**
- **mat**: A matrix.
- **ourzero**: A matrix the same size as mat with the value that zero should be reassigned to.

**Value**
A matrix

**See Also**
Other matrix_manipulation: `diag_matlab`; `test_for_max`; `test_for_min`
**Examples**

```r
test_for_zeros(cbind(2,3,0,5,6), 1e-18)
test_for_zeros(zeros(6), 1e-7)
```

---

**test_mat_size**  
*Test to make sure that matrices are the right size*

**Description**

Test to make sure that matrices are the right size.

**Usage**

```r
test_mat_size(correct_size, mat, name)
```

**Arguments**

- `correct_size`: the correct size of a matrix
- `mat`: The matrix to test.
- `name`: The name of the matrix as a string.

**Examples**

```r
test_mat_size(c(2,3), zeros(2,3), "foo")
```

```r
# Not run:
test_mat_size(c(2,3), zeros(2,6), "foo")
```

```r
# End(Not run)
test_mat_size(c(1,3), c(2,6,7), "foo")
```
tic

Timer function (as in MATLAB)

Description

Function to start a timer. Stop with toc().

Usage

tic(gcFirst = FALSE, name = "poped_savedTime")

Arguments

gcFirst Perform garbage collection?
name The saved name of the time object.

Note

This is a modified version of the same function in the matlab package tic

See Also

Other MATLAB: cell; diag_matlab; feval; fileparts; isempty; isfield; ones; randn; randperm; rand; size; toc; zeros

Examples

tic()
toc()
tic(name="foo")
toc()
tic()
toc()
tic()
toc()
tic()
toc(name="foo")
Description

Funtion to stop a timer. Start with tic().

Usage

toc(echo = TRUE, name = ".poped_savedTime")

Arguments

echo Print time to screen?
name The saved name of the time object.

Note

This is a modified version of the same function in the matlab package toc

See Also

Other MATLAB: cell; diag_matlab; feval; fileparts; isempty; isfield; ones; randn; randperm; rand; size; tic; zeros

Examples

tic()
toc()
tic(name="foo")
toc()
tic()
zeros  

*Create a matrix of zeros.*

---

**Description**

Function creates a matrix of zeros of size (dim1 x dim2). Written to match MATLAB’s `zeros` function.

**Usage**

```matlab
zeros(dim1, dim2 = NULL)
```

**Arguments**

- `dim1`  
  The dimension of the matrix (if square) or the number of rows.

- `dim2`  
  The number of columns

**Value**

A matrix of zeros.

**See Also**

Other MATLAB: `cell`; `diag_matlab`; `feval`; `fileparts`; `isempty`; `isfield`; `ones`; `randn`; `randperm`; `rand`; `size`; `tic`; `toc`

**Examples**

```matlab
zeros(3)
zeros(0, 3)
zeros(4, 7)
zeros(1, 4)
```
Index

a_line_search, 4, 6, 9, 21, 27, 47, 49, 100, 125, 163, 170, 174
bfgsb_min, 7, 9, 21, 26, 27, 49, 100, 125, 163, 170, 174
blockexp, 11, 14, 17, 19
blockfinal, 12, 13, 17, 19
blockheader, 12, 14, 15, 19, 163, 170
blockopt, 12, 14, 17, 18
calc Autofocus, 7, 9, 20, 27, 49, 100, 125, 163, 170, 174
calc_ofv_and_grad, 7, 9, 21, 26, 49, 100, 125, 163, 170, 174
cell, 29, 29, 44, 47, 77, 89, 97, 98, 152, 166–168, 176, 180–182
convert_variables, 30, 40, 42, 45, 52, 160
create.poped.database, 4, 30, 30, 42, 45, 52, 71, 73, 75, 77, 79, 81, 82, 84, 86, 144, 160
create_design, 30, 40, 41, 44, 45, 52, 144, 160
create_design_space, 30, 40, 42, 43, 52, 160
diag, 46
diag матлаб, 29, 46, 77, 89, 97, 98, 152, 166–168, 176–178, 180–182
do.call, 76
Dooptim, 7, 9, 21, 27, 47, 100, 125, 163, 170, 174
downscaling_general_design, 30, 40, 42, 45, 52, 160
Dtrace, 53
expression, 145
fdHess, 58
feps.add, 4, 71, 73, 75, 78, 79, 81, 83, 85, 87
feps.add.prop, 4, 72, 73, 75, 78, 79, 81, 83, 85, 87
feps.prop, 4, 72, 73, 74, 78, 79, 81, 83, 85, 87
feval, 29, 47, 76, 89, 97, 98, 152, 166–168, 176, 180–182
ff.PK.1.comp.oral.md.CL, 4, 72, 73, 75, 77, 79, 81, 83, 85, 87
ff.PK.1.comp.oral.md.KE, 4, 72, 73, 75, 78, 79, 81, 83, 85, 87
ff.PK.1.comp.oral.sd.CL, 4, 72, 73, 75, 78, 79, 80, 83, 85, 87
ff.PK.1.comp.oral.sd.KE, 4, 72, 73, 75, 78, 79, 81, 82, 85, 87
ff.PKPD.1.comp.oral.md.CL.imax, 4, 72, 73, 75, 78, 79, 81, 83, 84, 87
ff.PKPD.1.comp.sd.CL.emax, 4, 72, 73, 75, 78, 79, 81, 83, 85, 86
file, 14, 17, 100, 174
fileparts, 29, 47, 77, 88, 89, 97, 98, 152, 166–168, 176, 180–182

183
get_all_params, 91
get_rse, 4, 69, 92, 145, 156, 159
getfulld, 89
getTruncatedNormal, 90
ggplot2, 156, 158
isempty, 29, 47, 77, 89, 97, 98, 152, 166–168, 176, 180–182
isfield, 29, 47, 77, 89, 97, 98, 152, 166–168, 176, 180–182
LEDoptim, 7, 9, 21, 27, 49, 98, 125, 163, 170, 174
LinMatrixL, 104
log_prior_pdf, 109
mf9a, 7, 9, 21, 27, 49, 100, 124, 163, 170, 174
model_prediction, 4, 69, 93, 143, 156, 159
ofv_criterion, 24, 58, 62, 66, 69, 96, 102
INDEX


ones, 29, 47, 77, 89, 97, 98, 152, 166–168, 176, 180–182
optim, 58
pargen, 153
plot_efficiency_of_windows, 4, 69, 93, 145, 155, 159
plot_model_prediction, 4, 69, 93, 144, 145, 156, 157, 157
PopED (PopED-package), 3
poped (PopED-package), 3
PopED-package, 3
poped.choose, 30, 40, 42, 45, 52, 160
poped_optimize, 4, 7, 9, 21, 27, 49, 100, 125, 160, 170, 174
rand, 29, 47, 77, 89, 97, 98, 152, 166, 167, 168, 176, 180–182
randn, 29, 47, 77, 89, 97, 98, 152, 166, 167, 168, 176, 180–182
randperm, 29, 47, 77, 89, 97, 98, 152, 166, 167, 168, 176, 180–182
RS_opt, 4, 7, 9, 21, 27, 47, 49, 100, 125, 163, 168, 174
RS_opt_gen, 7, 9, 21, 27, 49, 98, 100, 125, 163, 170, 172
sample, 168
size, 29, 47, 77, 89, 97, 98, 152, 166–168, 176, 180–182
test_for_max, 47, 177, 178
test_for_min, 47, 177, 177, 178
test_for_zeros, 47, 177, 178, 178
test_mat_size, 179
tic, 29, 47, 77, 89, 97, 98, 152, 166–168, 176, 180, 180, 181, 182
toc, 29, 47, 77, 89, 97, 98, 152, 166–168, 176, 180, 181, 181, 182
zeros, 29, 47, 77, 89, 97, 98, 152, 166–168, 176, 180, 181, 182