Package ‘MOLHD’

May 6, 2018

Type Package

Title Multiple Objective Latin Hypercube Design

Version 0.2

Author Ruizhe Hou , Lu Lu

Maintainer Ruizhe Hou <houruizhe1210@gmail.com>

Description Generate the optimal maximin distance, minimax distance (only for low dimensions), and maximum projection designs within the class of Latin hypercube designs efficiently for computer experiments. Generate Pareto front optimal designs for each two of the three criteria and all the three criteria within the class of Latin hypercube designs efficiently. Provide criterion computing functions. References of this package can be found in Morris, M. D. and Mitchell, T. J. (1995) <doi:10.1016/0378-3758(94)00035-T>, Lu Lu and Christine M. Anderson-CookTimothy J. Robinson (2011) <doi:10.1198/Tech.2011.10087>, Joseph, V. R., Gul, E., and Ba, S. (2015) <doi:10.1093/biomet/asv002>.

Depends R (>= 3.4.0)

Imports fields, arrangements

License LGPL

NeedsCompilation yes

Repository CRAN

Date 2018-5-6

RoxygenNote 6.0.1

Date/Publication 2018-05-06 16:49:25 UTC

R topics documented:

`MOLHD-package` *Multiple objective Latin hypercube design*

Description

The MOLHD package provides useful and efficient functions for generating the optimal Maximin distance, Maximum Projection and miniMax distance (only for low dimensions) designs within the class of Latin hypercube designs for computer experiments. Ant it provides functions generating Pareto front optimal designs for each two of the three criteria and all the three criteria within the class of Latin hypercube designs. It also provides functions to compute the criteria for a given design.

Details

Package:	MOLHD
Type:	Package
Version:	0.2
Date:	2018-5-6
Lisense:	LGPL

This package contains functions for generating the optimal maximin distance designs, maximum projection designs and minimax distance designs for low dimensions within the class of Latin hypercube designs (LHDs). This packages also contains functions for generating designs on the Pareto front of each two of the three criteria as maximin distance criterion, minimax distance criterion, and maximum projection criterion. This package also contains functions to compute each criterion for a random Latin hypercube design.

Since minimax distance design is computational expensive, it is only approximately extimated when the design is at low dimension.

Author(s)

Ruizhe Hou, Lu Lu

Maintainer: Ruizhe Hou<houruizhe1210@gmail.com>

References

Morris, M. D. and Mitchell, T. J. (1995), "Exploratory Designs for Computation Experiments," *Journal of Statistical Planning and Inference*. <doi:10.1016/0378-3758(94)00035-T>

Lu Lu and Christine M. Anderson-CookTimothy J. Robinson (2011), "Optimization of Designed Experiments Based on Multiple Criteria Utilizing a Pareto Frontier," *Technometrics*. <doi:10.1198/Tech.2011.10087>

Joseph, V. R., Gul, E., and Ba, S. (2015), "Maximum Projection Designs for Computer experiments," *Biometrika*. <doi:10.1093/biomet/asv002>

---

cpf2 *Combine Pareto front designs of 2 criteria*

Description

Combine Pareto front designs of 2 criteria

Usage

```
cpf2(newdes, newpfval, curdes, curpfval)
```

Arguments

newdes	a matrix which is a column bind of new designs
newpfval	a matrix each row is 2 criteria correponding to each design
curdes	a matrix which is a column bind of current designs on Pareto front
curpfval	a matrix each row is 2 criteria correponding to each Pareto front design

Details

This function is used to combine 2 criteria Pareto front designs

Value

pfdes	The column bind of Pareto front designs
pfvals	The Pareto front values corresponding to the Pareto front designs

Examples

```
#Combine Pareto fronts each with 5 random starts for Mm and mp criteria
## Not run:
pf1=pfMp(20,2,crlim = cbind(c(4.5,6.5),c(26,36)),nstarts = 5)
pf2=pfMp(20,2,crlim = cbind(c(4.5,6.5),c(26,36)),nstarts = 5)
pfnew=cpf2(pf1$pfdes,pf1$pfvals,pf2$pfdes,pf2$pfvals)
pfnew$pfdes
pfnew$pfvals

## End(Not run)
```

`cpf3` *Combine Pareto front designs of 3 criteria*

Description

Combine Pareto front designs of 3 criteria

Usage

```
cpf3(newdes, newpfval, curdes, curpfval)
```

Arguments

`newdes`	a matrix which is a column bind of new designs
`newpfval`	a matrix each row is 3 criteria correponding to each design
`curdes`	a matrix which is a column bind of current designs on Pareto front
`curpfval`	a matrix each row is 3 criteria correponding to each Pareto front design

Details

This function is used to combine 3 criteria Pareto front designs

Value

`pfdes`	The column bind of Pareto front designs
`pfvals`	The Pareto front values corresponding to the Pareto front designs

Examples

```
#Combine Pareto fronts each with 1 random start for Mm, mp and mM criteria
## Not run:
pf1=pfMpm(20,2,crlim = cbind(c(4.5,6.5),c(26,36),c(0.12,0.62)),num = 15,nstarts = 1)
pf2=pfMpm(20,2,crlim = cbind(c(4.5,6.5),c(26,36),c(0.12,0.62)),num = 15,nstarts = 1)
pfnew=cpf3(pf1$pfdes,pf1$pfvals,pf2$pfdes,pf2$pfvals)
pfnew$pfdes
pfnew$pfvals

## End(Not run)
```

LHD *Generate a random Latin Hypercube design*

Description

Generate a random Latin Hypercube design

Usage

```
LHD(n, p)
```

Arguments

n	number of runs desired
p	number of design factors

Value

design	a Latin Hypercube Design that is not scaled (i.e. the grid point locations are integers)
standDesign	a standard Latin Hypercube Design that is scaled to (0,1); design locaitons are placed at the centers of selected grids.

Examples

```
#Generate a random Latin hypercube design with 20 runs and 2 variables
D<-LHD(n = 20,p = 2)
D$design
D$standDesign
```

md *Minimum distance between any two points in the design*

Description

Minimum distance between any two points in the design

Usage

```
md(D)
```

Arguments

D	a design matrix, rows are design locations, columns are design factors

Value

`MinimumDistance`
: Minimum distance between any two points in the design

`number`
: number of pairs in the design achieve the minimum distance

Examples

```
#compute the minimum distance between any two points in design D
d=md(D = cbind(c(0.875,0.375,0.125,0.625),c(0.375,0.125,0.625,0.875)))
d$MinimumDistance
d$number
```

---

miM — *The miniMax criterion baesd on an approximate fill distance measure*

Description

The miniMax criterion baesd on an approximate fill distance measure

Usage

```
miM(D, num = 50)
```

Arguments

`D`
: a design matrix, rows are design locations, columns are design factors

`num`
: Optional, default is "50". The fineness of the gridded points to divide the design space. Each dimension is evenly divided by num+1 points. Lower this parameter when dimension is high to reduce computing time.

Details

This function calculates the approximate fill distance for the design by using a set of gridded points, the maximum error of the value can be computed.

Value

fill distance with 4 decimals

Examples

```
#Compute the approximate fill distance of a design D
d=miM(D = cbind(c(0.875,0.375,0.125,0.625),c(0.375,0.125,0.625,0.875)), num = 20)
```

`miMLHD` *Generate the optimal Latin Hypercube Design based on the miniMax criterion.*

Description

Generate the optimal Latin Hypercube Design based on the miniMax criterion.

Usage

```
miMLHD(n, p, num = 50, temp0 = 0, nstarts = 1, times = 300,
  maxiter = 1e+06)
```

Arguments

`n`	number of runs desired
`p`	number of variables desired
`num`	Optional, default is "50". The fineness of the gridded points to divide the design space. Each dimension is evenly divided by num+1 points. Lower this parameter when dimension is high to reduce computing time.
`temp0`	Initial temperature for simulated annealing
`nstarts`	Optional, default is "1". The number of random starts
`times`	Optional, default is "300". The maximum number of non-improving searches allowed before terminating the search.
`maxiter`	Optional, default is "1e+06".The maximum total number of iterations for each random start. Lower this number if the design is prohibitively large and you want to terminate the algorithm prematurely to report the best design found

Details

This function is to search the optimal Latin Hypercube design based on the miniMax criterion using the columnwise exchange algorithm coupled with the simulated annealing algorithm, and several computational shortcuts to improve efficiency. The approximate miniMax criterion is computed by using a set of gridded points to approximate the continuous design space, the maximum error of the value can be computed.(Can only work in relatively low dimensions)

Value

`design`	The optimal miniMax design matrix
`criterion`	The opproximate miniMax criterion for the chosen fineness of the grids
`iterations`	The total iterations
`time_rec`	Time to complete the search

Examples

```
#Generate the optimal minimax distance LHD(20,2)
## Not run:
D=miMLHD(n=20,p=2)
D$design
D$criterion

## End(Not run)
```

Mm *Computer the approximate Maximin Criterion for a design.*

Description

Computer the approximate Maximin Criterion for a design.

Usage

```
Mm(D, power = 100)
```

Arguments

D	a design matrix
power	Optional, default is "100". The power parameter r in the average reciprocal inter-point distance measure. When r is approaching infinity, minimizing the average reciprocal inter-point distance measure is equivalent to maximizing the minimum distance among the design points.

Value

The approximate Maximin criterion with 4 decimals

Examples

```
#Compute the maximin criterion of a random LHD(20,2)
des=LHD(n = 20,p = 2)$standDesign
Mm(D=des, power=150)
```

MmLHD *Generate the optimal Maximin Latin Hypercube Design.*

Description

Generate the optimal Maximin Latin Hypercube Design.

Usage

```
MmLHD(n, p, power = 100, temp0 = 0, nstarts = 1, times = 300,
  maxiter = 1e+06)
```

Arguments

`n`	number of runs desired
`p`	number of variables desired
`power`	Optional, default is "100". The power parameter r in the average reciprocal inter-point distance measure. When r turns to infinity, minimizing the average reciprocal inter-point distance measure is equivalent to maximizing the minimum distance among the design points.
`temp0`	Initial temperature
`nstarts`	Optional, default is "1". The number of random starts
`times`	Optional, default is "300". The maximum number of non-improving searches allowed. Lower this parameter if you expect the search to converge faster.
`maxiter`	Optional, default is "1e+06".The maximum total number of iterations for each random start. Lower this number if the design is prohibitively large and you want to terminate the algorithm prematurely to report the best design found

Details

This function is to search the optimal Maximin design using columnwise exchange algorithm coupled with the simulated annealing algorithm and several computational shortcuts to improve efficiency.

Value

`design`	The optimal Maximin design matrix
`criterion`	The opproximate Maximin criterion of the design under chosen "power" parameter
`iterations`	The total iterations
`time_rec`	Time to complete the search

Examples

```
#Generate the optimal maximin distance LHD(20,2)
D=MmLHD(n=20,p=2)
D$design
D$criterion
```

---

mp *Computer the MaxPro Criterion for a design.*

Description

Computer the MaxPro Criterion for a design.

Usage

```
mp(D)
```

Arguments

D	a design matrix

Details

This function is to compute the MaxPro criterion for measuring projection characteristic of a computer experiment.

Value

The MaxPro Criterion with 4 decimals

Examples

```
#compute the mp criterion of a random LHD(20,2)
D=LHD(20,2)$standDesign
mp(D)
```

mpLHD *Generate the optimal MaxPro Latin Hypercube Design.*

Description

Generate the optimal MaxPro Latin Hypercube Design.

Usage

```
mpLHD(n, p, temp0 = 0, nstarts = 1, times = 300, maxiter = 1e+06)
```

Arguments

`n`	number of runs desired
`p`	number of design factors desired
`temp0`	Initial temperature for simulated annealing
`nstarts`	Optional, default is "1". The number of random starts
`times`	Optional, default is "300". The maximum number of non-improving searches allowed. Lower this parameter if you expect the search to converge faster.
`maxiter`	Optional, default is "1e+06".The maximum total number of iterations. Lower this number if the design is prohibitively large and you want to terminate the search prematurely to report the best design found

Details

This function is to search the optimal Latin Hypercube Design based on the MaxPro criterion using the columnwise exchange algorithm coupled with the simulated annealing algorithm, and several computational shortcuts to improve efficiency.

Value

`design`	The optimal LHD design matrix based on the MaxPro criterion
`criterion`	The MaxPro criterion of the selected optimal LHD design
`iterations`	The total iterations
`time_rec`	Time to complete the search

Examples

```
#Generate a optimal maximum projection LHD(20,2) design
D=mpLHD(n=20,p=2)
D$design
D$criterion
```

pfMm — *Generate the Pareto front for the optimal Latin Hypercube Designs based on both the Maximin and miniMax criteria.*

Description

Generate the Pareto front for the optimal Latin Hypercube Designs based on both the Maximin and miniMax criteria.

Usage

```
pfMm(n, p, crlim, num, nstarts = 1, times = 300, maxiter = 1e+06,
  temp0 = 0, wtset = cbind(c(1, 0), c(0.8, 0.2), c(0.6, 0.4), c(0.4, 0.6),
  c(0.2, 0.8), c(0, 1)))
```

Arguments

`n`	number of runs desired
`p`	number of design factors desired
`crlim`	a matrix saving the best and worst values for each criterion to be used for defining the scaling choices for converting the natural criteria values onto a desirability scale between 0 and 1. Each column corresponds to one criterion. The best and worst values are recommended based on the values from each single criterion search. It is recommended that slightly wider range is used for defining the scaling choice for the Pareto front search.
`num`	The fineness of the grids to approximiate the approximate the continuous design space. Lower this parameter when dimension is high to reduce computing time.
`nstarts`	Optional, default is "1". The number of random starts
`times`	Optional, default is "300". The maximum number of non-improving searches allowed before terminating the search. Lower this parameter if you expect the search to converge faster.
`maxiter`	Optional, default is "1e+06".The maximum total number of iterations. Lower this number if the design is prohibitively large.
`temp0`	Initial temperature for simualted annealing
`wtset`	Optional, default is "cbind(c(1,0),c(0.8,0.2),c(0.6,0.4),c(0.4,0.6),c(0.2,0.8),c(0,1))". The set of weight combinations to guide the search in varied directions. Each column is a weight vector that guides the search in a certain direction.

Details

This function is to search for the Pareto front and the Pareto set of LHDs based on the Maximin and miniMax criteria. Each design on Pareto front is not dominated by any other design.This function utilizes a version of simulated annealing algorithm and several computational shortcuts to efficiently generate the optimal Latin hypercube designs. Choose a lower maximum limit of the criteria but high enough for Pareto front designs will save the computing time.

Value

pfdes — The column bind of Pareto front designs whose criteria values are on the Pareto front.

pfvals — The Pareto front of criteria values based on the Maximin and miniMax criteria. Columns are the optimization criteria.

time_rec — Time to complete the search

Examples

```
#Generate the Pareto front designs of maximin and minimax distance criterion for LHD(10,2)
## Not run:
D1=MmLHD(n=10,p=2,nstarts=30)
D2=miMLHD(n=10,p=2,num=15,nstarts=30)
Mmlim=c(D1$criterion-0.2,D1$criterion-0.2+2)
mMlim=c(D2$criterion-0.05,D2$criterion-0.05+0.5)
crlim=cbind(Mmlim,mMlim)
pf=pfMm(10,2,crlim,num = 15,nstarts = 30)
pf$pfvals
pf$pfdes
pf$time_rec

## End(Not run)
```

---

pfMp — *Generate the Pareto front for optimal Latin Hypercube Designs based on both the Maximin and the MaxPro criteria.*

Description

Generate the Pareto front for optimal Latin Hypercube Designs based on both the Maximin and the MaxPro criteria.

Usage

```
pfMp(n, p, crlim, nstarts = 1, times = 300, maxiter = 1e+06, temp0 = 0,
  wtset = cbind(c(1, 0), c(0.8, 0.2), c(0.6, 0.4), c(0.4, 0.6), c(0.2, 0.8),
  c(0, 1)))
```

Arguments

n — number of runs desired

p — number of design factors desired

crlim — a matrix saving the best and worst values for each criterion to be used for defining the scaling choices for converting the natural criteria values onto a desirability scale between 0 and 1. Each column corresponds to one criterion. The best and worst values are recommended based on the values from each single criterion search. It is recommended that slightly wider range is used for defining the scaling choice for the Pareto front search.

`nstarts` Optional, default is "1". The number of random starts

`times` Optional, default is "300". The maximum number of non-improving searches allowed before terminating the search. Lower this parameter if you expect the search to converge faster.

`maxiter` Optional, default is "1e+06".The maximum total number of iterations. Lower this number if the design is prohibitively large and you want to terminate the search earlier to report the best design found.

`temp0` Initial temperature for simualted annealing

`wtset` Optional, default is "cbind(c(1,0),c(0.8,0.2),c(0.6,0.4),c(0.4,0.6),c(0.2,0.8),c(0,1))". The set of weight combinations to guide the search in varied directions. Each column is a weight vector that guides the search in a certain direction.

Details

This function is to search the Pareto front and the Pareto set of designs based on the Maximin and Maxpro criteria. Each design on Pareto front is not dominated by any other design.This function utilizes a version of simulated annealing algorithm and several computational shortcuts to efficiently generate the optimal Latin hypercube designs. Choose a lower maximum limit of the criteria but high enough for Pareto front designs will save the computing time.

Value

`pfdes` The column bind of Pareto front designs whose criteria values are on the Pareto front.

`pfvals` The Pareto front of criteria values based on the Maximin and MaxPro criteria. Columns are the optimization criteria.

`time_rec` Time to complete the search

Examples

```
#Generate the Pareto designs of maximin distance and maximum projection for LHD(10,5)
## Not run:
D1=MmLHD(n=10,p=5,nstarts=30)
D2=mpLHD(n=10,p=5,nstarts=30)
Mmlim=c(D1$criterion-0.2,D1$criterion-0.2+2)
mplim=c(D2$criterion-2,D2$criterion-2+10)
crlim=cbind(Mmlim,mplim)
pf=pfMp(10,5,crlim,nstarts = 30)
pf$pfvals
pf$pfdes
pf$time_rec

## End(Not run)
```

---

pfMpm — *Generate the Pareto front for the optimal Latin Hypercube Designs based on the Maximin, MaxPro and miniMax criteria.*

Description

Generate the Pareto front for the optimal Latin Hypercube Designs based on the Maximin, MaxPro and miniMax criteria.

Usage

```
pfMpm(n, p, crlim, num, nstarts = 1, times = 300, maxiter = 1e+06,
  temp0 = 0, wtset = cbind(c(1, 0, 0), c(0.5, 0.5, 0), c(0.5, 0, 0.5), c(0,
  0.5, 0.5), c(0, 1, 0), c(0, 0, 1), c(1/3, 1/3, 1/3)))
```

Arguments

`n`	number of runs desired
`p`	number of design factors desired
`crlim`	a matrix saving the best and worst values for each criterion to be used for defining the scaling choices for converting the natural criteria values onto a desirability scale between 0 and 1. Each column corresponds to one criterion. The best and worst values are recommended based on the values from each single criterion search. It is recommended that slightly wider range is used for defining the scaling choice for the Pareto front search.
`num`	The fineness of the grids to approximiate the approximate the continuous design space. Lower this parameter when dimension is high to reduce computing time.
`nstarts`	Optional, default is "1". The number of random starts
`times`	Optional, default is "300". The maximum number of non-improving searches allowed before terminating the search. Lower this parameter if you expect the search to converge faster.
`maxiter`	Optional, default is "1e+06".The maximum total number of iterations. Lower this number if the design is prohibitively large.
`temp0`	Initial temperature for simualted annealing
`wtset`	Optional, default is "cbind(c(1,0,0),c(0.5,0.5,0),c(0.5,0,0.5),c(0,0.5,0.5),c(0,1,0), c(0,0,1),c(1/3,1/3,1/3))". The set of weight combinations to guide the search in varied directions. Each column is a weight vector that guides the search in a certain direction.

Details

This function is to search for the Pareto front and the Pareto set of LHDs based on the Maximin, Maxpro and miniMax criteria. Each design on Pareto front is not dominated by any other design.This function utilizes a version of simulated annealing algorithm and several computational shortcuts to efficiently generate the optimal Latin hypercube designs. Choose a lower maximum limit of the criteria but high enough for Pareto front designs will save the computing time.

Value

pfdes	The column bind of Pareto front designs whose criteria values are on the Pareto front.
pfvals	The Pareto front of criteria values based on the Maximin, MaxPro and miniMax criteria. Columns are the optimization criteria.
time_rec	Time to complete the search

Examples

```
#Generate the Pareto front designs of maximin distance,
#minimax diatance, and maximum projection criterion for LHD(10,2)
## Not run:
D1=MmLHD(n=10,p=2,nstarts=30)
D2=mpLHD(n=10,p=2,nstarts=30)
D3=miMLHD(n=10,p=2,num=15,nstarts=30)
Mmlim=c(D1$criterion-0.2,D1$criterion-0.2+2)
mplim=c(D2$criterion-2,D2$criterion-2+10)
mMlim=c(D3$criterion-0.05,D3$criterion-0.05+0.5)
crlim=cbind(Mmlim,mplim,mMlim)
pf=pfMpm(10,2,crlim,num = 15,nstarts = 30)
pf$pfvals
pf$pfdes
pf$time_rec

## End(Not run)
```

---

pfpm *Generate the Pareto front for the optimal Latin Hypercube Designs based on both the MaxPro and miniMax criteria.*

Description

Generate the Pareto front for the optimal Latin Hypercube Designs based on both the MaxPro and miniMax criteria.

Usage

```
pfpm(n, p, crlim, num, nstarts = 1, times = 300, maxiter = 1e+06,
  temp0 = 0, wtset = cbind(c(1, 0), c(0.8, 0.2), c(0.6, 0.4), c(0.4, 0.6),
  c(0.2, 0.8), c(0, 1)))
```

Arguments

n	number of runs desired
p	number of design factors desired

`crlim` a matrix saving the best and worst values for each criterion to be used for defining the scaling choices for converting the natural criteria values onto a desirability scale between 0 and 1. Each column corresponds to one criterion. The best and worst values are recommended based on the values from each single criterion search. It is recommended that slightly wider range is used for defining the scaling choice for the Pareto front search.

`num` The fineness of the grids to approximiate the approximate the continuous design space. Lower this parameter when dimension is high to reduce computing time.

`nstarts` Optional, default is "1". The number of random starts

`times` Optional, default is "300". The maximum number of non-improving searches allowed before terminating the search. Lower this parameter if you expect the search to converge faster.

`maxiter` Optional, default is "1e+06".The maximum total number of iterations. Lower this number if the design is prohibitively large.

`temp0` Initial temperature for simualted annealing

`wtset` Optional, default is "cbind(c(1,0),c(0.8,0.2),c(0.6,0.4),c(0.4,0.6),c(0.2,0.8),c(0,1))". The set of weight combinations to guide the search in varied directions. Each column is a weight vector that guides the search in a certain direction.

Details

This function is to search for the Pareto front and the Pareto set of LHDs based on the MaxPro and miniMax criteria. Each design on Pareto front is not dominated by any other design.This function utilizes a version of simulated annealing algorithm and several computational shortcuts to efficiently generate the optimal Latin hypercube designs. Choose a lower maximum limit of the criteria but high enough for Pareto front designs will save the computing time.

Value

`pfdes` The column bind of Pareto front designs whose criteria values are on the Pareto front.

`pfvals` The Pareto front of criteria values based on the Maximin and miniMax criteria. Columns are the optimization criteria.

`time_rec` Time to complete the search

Examples

```
#Generate the Pareto front designs of maximum projection and minimax distance criteria for LHD(10,2)
## Not run:
D1=mpLHD(n=10,p=2,times=1000,nstarts=30)
D2=miMLHD(n=10,p=2,num=15,nstarts=30)
mplim=c(D1$criterion-2,D1$criterion-2+20)
mMlim=c(D2$criterion-0.05,D2$criterion-0.05+0.5)
crlim=cbind(mplim,mMlim)
pf=pfpm(10,2,crlim,num = 15,nstarts = 30)
pf$pfvals
pf$pfdes
```

```
pf$time_rec

## End(Not run)
```

Index