Package 'MaxPro'

September 28, 2018

Type Package

Title Maximum Projection Designs

Version 4.1-2

Date 2018-09-27

Author Shan Ba and V. Roshan Joseph

Maintainer Shan Ba <shanbatr@gmail.com>

Depends nloptr

Description Generate maximum projection (MaxPro) designs for quantitative and/or qualitative factors. Details of the MaxPro criterion can be found in: (1) Joseph, Gul, and Ba. (2015) ``Maximum Projection Designs for Computer Experiments'', Biometrika, 102, 371-380, and (2) Joseph, Gul, and Ba. (2018) ``Designing Computer Experiments with Multiple Types of Factors: The MaxPro Approach'', Journal of Quality Technology, to appear.

License LGPL-2.1

NeedsCompilation yes

Repository CRAN

Date/Publication 2018-09-28 04:40:03 UTC

R topics documented:

MaxPro-package   *MaxPro package*

Description

Maximum projection (MaxPro) designs for quantitative and/or qualitative factors

Details

Package:	MaxPro
Type:	Package
Version:	4.1-2
Date:	2018-09-27
License:	LGPL-2.1

Important functions in this package are: MaxProLHD generates the MaxPro Latin hypercube design for continuous factors, MaxPro searches for the locally optimal MaxPro design in the neighborhood of a given initial design for continuous factors, MaxProQQ generates the MaxPro design with both quantitative and qualitative factors, MaxProAugment sequentially augments an existing design with additional design points based on the MaxPro criterion for all types of factors and constraints, MaxProRunOrder finds the optimal order to run a given design in a sequential manner based on the MaxPro criterion, and MaxProMeasure reports the MaxPro criterion value for a given design.

Author(s)

Shan Ba and V. Roshan Joseph

Maintainer: Shan Ba <shanbatr@gmail.com>

References

Joseph, V. R., Gul, E., and Ba, S. (2015) "Maximum Projection Designs for Computer Experiments," *Biometrika*, 102, 371-380.

Joseph, V. R., Gul, E., and Ba, S. (2018) "Designing Computer Experiments with Multiple Types of Factors: The MaxPro Approach," *Journal of Quality Technology*, to appear.

CandPoints   *Generate Candidate Design Points Randomly for Various Types of Factors*

Description

Quickly generate random candidate design points for different types of factors which can be used as an input to the MaxProAugment function.

Usage

```
CandPoints(N,p_cont,l_disnum=NULL,l_nom=NULL)
```

Arguments

N	The number of random candidate design points to generate
p_cont	The number of continuous factors
l_disnum	Optional, default is "NULL". A vector containing the number of levels for each discrete numeric factor (NULL if there are no discrete numeric factors)
l_nom	Optional, default is "NULL". A vector containing the number of levels for each nominal factor (NULL if there are no nominal factors)

Details

This function generates uniform random numbers for each continuous factor and randomly sampled levels for each discrete numeric and nominal factor. The result can be used as the candidate design in the [MaxProAugment](#) function.

Value

The value returned from the function is a matrix containing the candidate design points.

Author(s)

Shan Ba <shanbatr@gmail.com> and V. Roshan Joseph <roshan@isye.gatech.edu>

See Also

[MaxProAugment](#)

Examples

```
CCC<-CandPoints(N=10000,p_cont=3,l_disnum=c(3,3,4),l_nom=c(2,2,3))
```

MaxPro	*Locally Optimal Maximum Projection Designs for Continuous Factors*

Description

Find the locally optimal maximum projection (MaxPro) design in the neighborhood of a given initial design for continuous factors.

Usage

```
MaxPro(InitialDesign,s=2,iteration=10)
```

Arguments

InitialDesign	The initial design matrix, which we recommend to be a MaxPro Latin hypercube design generated by the [MaxProLHD](#) function
s	Optional, default is "2". The parameter in defining the s-norm distance (2 corresponds to Euclidean distance)
iteration	Optional, default is "10". The number of iterations in running the continuous local search

Details

This function applies a continuous optimization algorithm in nloptr (Ypma 2014) to find the locally optimal MaxPro design in the neighborhood of the initial design. A MaxPro Latin hypercube design generated by the [MaxProLHD](#) function is a good choice for the initial design. Please refer to Joseph, Gul and Ba (2015) for details.

Value

The value returned from the function is a list containing the following components:

Design	The locally optimal MaxPro design matrix
measure	The MaxPro criterion measure for the locally optimal design

Author(s)

Shan Ba <shanbatr@gmail.com> and V. Roshan Joseph <roshan@isye.gatech.edu>

References

Joseph, V. R., Gul, E., and Ba, S. (2015) "Maximum Projection Designs for Computer Experiments," *Biometrika*, 102, 371-380.

Ypma, J. (2014) "Introduction to nloptr: an R interface to NLopt", R Package Version 1.0.0.

See Also

[MaxProLHD](#), [MaxProRunOrder](#), [MaxProAugment](#)

Examples

```
InitialDesign<-MaxProLHD(n = 10, p = 4)$Design
DOX<-MaxPro(InitialDesign)
DOX$Design
```

MaxProAugment   *Augment an Existing Design based on the Maximum Projection Crite-*
                             rion

Description

Select the best set of design points to augment a given design matrix by optimizing the maximum projection (MaxPro) criterion sequentially. When multiple types of factors exist, all continuous factors should be placed as the leftmost columns in the design matrix, discrete numeric factors should be placed as middle columns, and all nominal factors should be placed at the rightmost columns in the design matrix. The ordinal factors, if any, should be pre-converted into discrete numeric factors through the scoring method (see, e.g., Wu and Hamada 2009, Section 14.10). This function can also be used for creating a nonadaptive sequential design, designs to run in batches, validation runs, designs in constrained regions, and nested designs.

Usage

```
MaxProAugment(ExistDesign, CandDesign, nNew,
p_disnum=0, l_disnum=NULL, p_nom=0, l_nom=NULL)
```

Arguments

ExistDesign	The existing design matrix to be augmented, where each row is an experimental run and each column is a factor. All columns of the continuous and discrete numeric factors should be standardized into the unit range of [0,1].
CandDesign	The matrix containing candidate design points, whose columns should match those in the ExistDesign matrix. All columns of the continuous and discrete numeric factors should be standardized into the unit range of [0,1]. Function [CandPoints](#) can be used to quickly generate candidate design points.
nNew	The number of new design points to add
p_disnum	Optional, default is 0. The number of discrete numeric factors
l_disnum	Optional, default is a vector containing the number of distinct levels for each discrete numeric factor in the ExistDesign and CandDesign. The vector specifying the number of levels for each discrete numeric factor
p_nom	Optional, default is 0. The number of nominal factors
l_nom	Optional, default is a vector containing the number of levels for each nominal factor in the ExistDesign and CandDesign. The vector specifying the number of levels for each nominal factor

Details

This function uses a greedy search algorithm to select the best nNew points out of the CandDesign matrix to augment the ExistDesign matrix based on the MaxPro criterion. Factor constraints (irregular-shaped design space), if any, can be incorporated through the specification of valid candidate design points in CandDesign (please see the example below).

Value

The value returned from the function is a list containing the following components:

Design	The augmented design matrix (including the original existing design points)
measure	The MaxPro criterion measure of the augmented design
time_rec	Time to complete the search

Author(s)

Shan Ba <shanbatr@gmail.com> and V. Roshan Joseph <roshan@isye.gatech.edu>

References

Joseph, V. R., Gul, E., and Ba, S. (2015) "Maximum Projection Designs for Computer Experiments," *Biometrika*, 102, 371-380.

Joseph, V. R. (2016) "Rejoinder," *Quality Engineering*, 28, 42-44.

Joseph, V. R., Gul, E., and Ba, S. (2018) "Designing Computer Experiments with Multiple Types of Factors: The MaxPro Approach," *Journal of Quality Technology*, to appear.

Wu, C. F. J., and Hamada, M. (2009), *Experiments: Planning, Analysis, and Parameter Design Optimization, 2nd Edition*, New York: Wiley.

See Also

CandPoints, MaxProLHD, MaxProQQ

Examples

```
ExistDesign=matrix(c(0.11,0.96,0.42,0.99),ncol=2,byrow=TRUE)

#Generate candidate design points that satisfy the factor constraints
S=CandPoints(N=10000,p_cont=2)
S=S[S[,1]+S[,2]<=1.5,] #constraint 1
S=S[(S[,1]-.5)^2+(S[,2]-.5)^2>=(2*.25^2),] #constraint 2
plot(S[,1],S[,2],bty="n",cex=.5,xlab=expression(x[1]),ylab=expression(x[2]),xlim=c(0,1),ylim=c(0,1))

result=MaxProAugment(ExistDesign, CandDesign=S, nNew=20)
D=result$Design
points(D[,1],D[,2],col='red',pch=16)
text(D[,1],D[,2],1:nrow(D),col=2,cex=2)
```

MaxProLHD	*Maximum Projection Latin Hypercube Designs for Continuous Factors*

Description

Generate the maximum projection (MaxPro) Latin hypercube design for continuous factors based on a simulated annealing algorithm. If nominal, discrete numeric, or ordinal factors exist, please see the function MaxProQQ.

Usage

```
MaxProLHD(n, p, s=2, temp0=0, nstarts = 1, itermax = 400, total_iter = 1e+06)
```

Arguments

n	The number of runs (design points)
p	The number of input factors (variables)
s	Optional, default is "2". The parameter in defining the s-norm distance (2 corresponds to Euclidean distance)
temp0	Optional, The initial temperature in the simulated annealing algorithm. Change this value if you want to start with a higher or lower temperature
nstarts	Optional, default is "1". The number of random starts
itermax	Optional, default is "400". The maximum number of non-improving searches allowed under each temperature. Lower this parameter if you want the algorithm to converge faster
total_iter	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 algorithm prematurely to report the best design found so far

Details

This function utilizes a version of the simulated annealing algorithm to efficiently generate the optimal Latin hypercube designs for continuous factors based on the MaxPro criterion. Parameters in the algorithm may need to be properly tuned to achieve global convergence. Please refer to Joseph, Gul and Ba (2015) for details.

Value

The value returned from the function is a list containing the following components:

Design	Design matrix
temp0	Initial temperature
measure	The MaxPro criterion measure
time_rec	Time to complete the search
ntotal	The total number of iterations

Author(s)

Shan Ba <shanbatr@gmail.com> and V. Roshan Joseph <roshan@isye.gatech.edu>

References

Joseph, V. R., Gul, E., and Ba, S. (2015) "Maximum Projection Designs for Computer Experiments," *Biometrika*, 102, 371-380.

See Also

MaxProRunOrder, MaxProAugment, MaxProQQ

Examples

```
obj<-MaxProLHD(n = 10, p = 4)
obj$Design
```

MaxProMeasure   *Maximum Projection Criterion Measure*

Description

Calculate the maximum projection (MaxPro) criterion measure for a given design.

Usage

```
MaxProMeasure(Design, p_nom=0)
```

Arguments

Design   The design matrix where each row is an experimental run and each column is a
                factor. The rightmost p_nom columns correspond to the p_nom nominal factors,
                and the columns on the left are for continuous factors and discrete numeric fac-
                tors. The ordinal factors, if any, should be pre-converted into discrete numeric
                factors through the scoring method (see, e.g., Wu and Hamada 2009, Section
                14.10). All columns of the continuous and discrete numeric factors should be
                standardized into the unit range of [0,1].

p_nom   Optional, default is "0". The number of nominal factors

Details

This function computes the MaxPro criterion measure for a given design. Please refer to Joseph,
Gul and Ba (2015) and Joseph, Gul and Ba (2018) for details.

Value

The value returned from the function is the MaxPro criterion measure

Author(s)

Shan Ba <shanbatr@gmail.com> and V. Roshan Joseph <roshan@isye.gatech.edu>

References

Joseph, V. R., Gul, E., and Ba, S. (2015) "Maximum Projection Designs for Computer Experiments," *Biometrika*, 102, 371-380.

Joseph, V. R., Gul, E., and Ba, S. (2018) "Designing Computer Experiments with Multiple Types of Factors: The MaxPro Approach," *Journal of Quality Technology*, to appear.

Wu, C. F. J., and Hamada, M. (2009), *Experiments: Planning, Analysis, and Parameter Design Optimization, 2nd Edition*, New York: Wiley.

See Also

MaxProLHD, MaxProQQ, MaxProAugment

Examples

```
D<-MaxProLHD(10,2)$Design
MaxProMeasure(D,p_nom=0)
```

MaxProQQ	*Maximum Projection Designs with Quantitative and Qualitative Factors*

Description

Generate the maximum projection (MaxPro) design with quantitative and qualitative (QQ) factors. The quantitative factors can be continuous factors and/or discrete numeric factors, and the qualitative factors can be nominal factors and/or ordinal factors. If all factors are continuous, please use MaxProLHD directly.

To use this function, first convert all the ordinal factors into discrete numeric factors through the scoring method (see, e.g., Wu and Hamada 2009, Section 14.10). That is, if the ordinal factor has levels, "poor", "fair" and "good", then depending on the nature of the classification, the experimenter can choose some discrete numeric levels such as (1,2,3) or (1,4,5) to represent the three ordinal levels. Also, all columns of the continuous and discrete numeric factors should be standardized into the unit range of [0,1]

The users need to provide their desirable sub-design matrix for the nominal factors as part of the initial design input to this function. This function only optimizes the sub-design matrix of the continuous and discrete numeric factors as well as their row orders in joining with the pre-determined sub-design matrix of the nominal factors.

Usage

```
MaxProQQ(InitialDesign, p_nom=0, temp0=0, nstarts=1, itermax=400, total_iter=1e+06)
```

Arguments

InitialDesign   The initial design matrix where each row is an experimental run and each column is a factor. The rightmost p_nom columns correspond to the p_nom nominal factors, and the columns on the left are for continuous factors and discrete numeric factors (including ordinal factors). It is recommended to use a random Latin hypercube design as the initial design for continuous factors, and all columns of the continuous and discrete numeric factors should be standardized into the unit range of [0,1]. This function only optimizes the element permutations within the columns for continuous and discrete numeric factors. Elements in the last p_nom columns are pre-determined by the user and fixed in this function.

p_nom   Optional, default is "0". The number of nominal factors

temp0   Optional, The initial temperature in the simulated annealing algorithm. Change this value if you want to start with a higher or lower temperature

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

itermax   Optional, default is "400". The maximum number of non-improving searches allowed under each temperature. Lower this parameter if you want the algorithm to converge faster

total_iter   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 algorithm prematurely to report the best design found so far

Details

This function utilizes a version of the simulated annealing algorithm to efficiently generate the optimal design for continuous, nominal, discrete numeric and ordinal types of factors, where the design columns for nominal factors are pre-determined by the users and all the other columns are optimized based on the MaxProQQ criterion for the combined design. Parameters in the algorithm may need to be properly tuned to achieve global convergence. Please refer to Joseph, Gul and Ba (2018) for details.

Value

The value returned from the function is a list containing the following components:

Design   Design matrix

temp0   Initial temperature

measure   The MaxProQQ criterion measure

time_rec   Time to complete the search

ntotal   The total number of iterations

Author(s)

Shan Ba <shanbatr@gmail.com> and V. Roshan Joseph <roshan@isye.gatech.edu>

References

Joseph, V. R., Gul, E., and Ba, S. (2015) "Maximum Projection Designs for Computer Experiments," *Biometrika*, 102, 371-380.

Joseph, V. R., Gul, E., and Ba, S. (2018) "Designing Computer Experiments with Multiple Types of Factors: The MaxPro Approach," *Journal of Quality Technology*.

Wu, C. F. J., and Hamada, M. (2009), *Experiments: Planning, Analysis, and Parameter Design Optimization, 2nd Edition*, New York: Wiley.

See Also

MaxProRunOrder, MaxProAugment, MaxProLHD

Examples

```
#18-run design
#3 continuous factors, 1 discrete numeric factor (6 levels), 2 nominal factors (3 levels)

#Generate a random Latin hypercube design (each factor standardized into [0,1])
#as the initial design for continuous factors
rand_design_part1=apply(matrix(rep(seq(from=0,to=1,length=18),3),ncol=3),2,sample)

#Generate a random initial design (standardized into [0,1]) for discrete numeric factors
rand_design_part2=sample(rep(seq(from=0,to=1,length=6),each=3))

#Construct an optimal design for the two nominal factors
OA_matrix=cbind(rep(1:3,each=6),rep(1:3,6))

#Initial design matrix
InitialDesign=cbind(rand_design_part1,rand_design_part2,OA_matrix)

#Optimize the design based on MaxProQQ criterion
obj=MaxProQQ(InitialDesign, p_nom=2)
obj$Design
```

MaxProRunOrder | *Find the Optimal Sequential Order to Run a Given Experimental Design*

Description

Given a computer experimental design matrix, this function searches for an optimal run (row) order based on the maximum projection (MaxPro) criterion. This optimal order enables the given design to be run in a sequential manner: when terminated at any step, the previous design points form a nearly optimal subset based on the MaxPro criterion.

Usage

```
MaxProRunOrder(Design,p_nom=0,initial_row=1)
```

Arguments

Design   The design matrix, where each row is an experimental run and each column is a factor. The rightmost p_nom columns correspond to the p_nom nominal factors, and the columns on the left are for continuous factors and discrete numeric factors. The ordinal factors, if any, should be pre-converted into discrete numeric factors through the scoring method (see, e.g., Wu and Hamada 2009, Section 14.10). All columns of the continuous and discrete numeric factors should be standardized into the unit range of [0,1].

p_nom   Optional, default is 0. The number of nominal factors

initial_row   Optional, default is 1. The vector specifying the row number of each design point in the given design matrix that should be run at first or have already been run.

Details

This function utilizes a greedy search algorithm to find the optimal row order to run the given experimental design based on the MaxPro criterion.

Value

The value returned from the function is a list containing the following components:

Design   The design matrix in optimal run (row) order. The run sequence ID is added as the first column

measure   The MaxPro criterion measure of the given design

time_rec   Time to complete the search

Author(s)

Shan Ba <shanbatr@gmail.com> and V. Roshan Joseph <roshan@isye.gatech.edu>

References

Joseph, V. R., Gul, E., and Ba, S. (2015) "Maximum Projection Designs for Computer Experiments," *Biometrika*, 102, 371-380.

Joseph, V. R. (2016) "Rejoinder," *Quality Engineering*, 28, 42-44.

Joseph, V. R., Gul, E., and Ba, S. (2018) "Designing Computer Experiments with Multiple Types of Factors: The MaxPro Approach," *Journal of Quality Technology*, to appear.

Wu, C. F. J., and Hamada, M. (2009), *Experiments: Planning, Analysis, and Parameter Design Optimization, 2nd Edition*, New York: Wiley.

See Also

[MaxProLHD](#), [MaxProQQ](#), [MaxProAugment](#)

Examples

```
D0=MaxProLHD(25,2)$Design

#Assume the first two rows of the design have already been executed
#Find the optimal run orders

D=MaxProRunOrder(D0,p_nom=0,initial_row=c(1,2))$Design
plot(D[,2],D[,3],xlim=c(0,1),ylim=c(0,1),type="n",
xlab=expression(x[1]),ylab=expression(x[2]),cex.lab=1.5)
text(D[,2],D[,3],labels=D[,1],col='red')
```

Index