Package 'mixexp'

July 2, 2014

Type Package

Title Design and analysis of mixture experiments

Version 1.1.2

Date 2014-2-13

Author John Lawson <lawson@byu.edu> and Cameron Willden <ccwillden@gmail.com>, with contributions from Greg Piepel

Maintainer John Lawson <lawson@byu.edu>

Description This package contains functions for creating designs for mixture experiments, making ternary contour plots, and making mixture effect plots.

License GPL-2

Depends gdata, lattice, grid

LazyLoad yes

Repository CRAN

Repository/R-Forge/Project daewr

Repository/R-Forge/Revision 78

Repository/R-Forge/DateTimeStamp 2014-05-19 16:38:49

Date/Publication 2014-05-28 23:56:32

NeedsCompilation yes

R topics documented:

`mixexp-package` *This package contains functions for creating designs for mixture experiments and making graphical display of results of mixture experiments.*

Description

The **mixexp** package provides functions for creating mixture experiment designs in an unconstrained simplex or constrained mixture space. Functions are also provided for making ternary contour plots, pictures of constrained regions, design points, and mixture effect plots.

Details

Package:	mixexp
Type:	Package
Version:	1.0-1
Date:	2011-05-31
License:	GPL2.0
Dependencies:	gdata, lattice, grid
LazyLoad:	yes
Packaged:	2011-05-31 19:54:07 UTC; Lawson
Built:	R 2.12.1; i386-pc-mingw32; 2011-05-31 19:54:08 UTC; windows

Index:

crvtave	function for creating extreme vertices designs and centroids; this function calls Eflags, Nrows, and Vertcen
DesignPoints	function for plotting design points and or mixture constraint in the simplex
Eflags	function for calling Piepel's fortran code cnvrt to create extreme vertices designs and prints any error messages
Effplot	function for making mixture effect plots
MixturePlot	function for making contour plots in simplex region
ModelPlot	function for making contour plots of an equation created in by the lm function
Nrows	function for calling Piepel's fortran code cnvrt to create extreme vertices designs and returns the number of rows in the resulting design
SCD	function for creating Simplex Centroid Designs
SLD	function for creating Simplex Lattice Designs
Vertcen	function for calling Piepel's fortran code cnvrt to create extreme vertices designs and returns the resulting design
Xvert	function for creating extreme vertices design and centroids, this function calls crvtave

Author(s)

John Lawson <lawson@byu.edu> and Cameron Willden <ccwillden@gmail.com>

Maintainer: John Lawson <lawson@byu.edu>

---

conmx *Example constraint matrix from Piepel 1988*

Description

This is an .rda file containing the constraint matrix.

Usage

```
data(conmx)
```

Format

An 8 x 4 matrix

Source

source

References

Piepel, G. F. (1988) Programs for Generating Extreme Vertices and Centroids of Linearly Constrained Experimental Regions, *Journal of Quality Technology*, Vol. 20, No. 2.

`crvtave` *This function creates an extreme vertices design*

Description

This function calls the function Vertcen which uses Piepel's (1988) fortran code (cnvrt) for generating extreme vertices and centroids of linearly constrained mixture experimental regions.

Usage

```
crvtave(ndm,conmx)
```

Arguments

`ndm`	is an integer representing the highest order of centroids requested. An overall centroid is always included, 0 indicates no other centroids will be created, 1 indicates edge centroids are requested, 2 indicates face centroids, etc.
`conmx`	This is the matrix of constraints.

Value

`vtcn`	This is a data frame containing the extreme vertices design. The columns are labeled x1, x2 ...xn, where n is the number of mixture variables. The last column is labeled dimen and it indicates the order of centroid where 0 is an extreme vertex, 1 is an edge centroid, 2 is a face centroid, and n is the overall centroid.

Note

This function calls the function Eflags to get error messages from cnvrt, the function Vertcen to get the extreme vertices and centroids from cnvrt, and the function Nrows to get the number of vertices and centroids from cnvrt.

Author(s)

John S. Lawson <lawson@byu.edu>

References

1. Pieple, G. F. "Programs for Generating Extreme Vertices and Centroids of Linearly Constrained Experimental Regions" *Journal of Quality Technology*, Vol 20, No. 2, pp. 125-139, 1988.

Examples

```
data(conmx)
crvtave(1,conmx)
```

`DesignPoints` *This function plots design points and or constraints in the simplex mixture space.*

Description

This function plots design points and or constraints in the simplex mixture space. It calls the function MixturePlot that does the actual plotting.

Usage

```
DesignPoints(des = NULL, x = NULL, y = NULL, z = NULL, x1lower = 0,
    x1upper = 0, x2lower = 0, x2upper = 0, x3lower = 0, x3upper = 0,
    axislabs=c("Fraction x1","Fraction x2","Fraction x3"),
    cornerlabs = c("x1","x2","x3"))
```

Arguments

`des`	data frame containing x1 x2 and x3 coordinates of data points to be plotted
`x`	vector of x3 coordinates of design points to be plotted
`y`	vector of x2 coordinates of design points to be plotted
`z`	vector of x1 coordinates of design points to be plotted
`x1lower`	This is the lower constraint on x1
`x1upper`	This is the upper constraint on x1
`x2lower`	This is the lower constraint on x2
`x2upper`	This is the upper constraint on x2
`x3lower`	This is the lower constraint on x3
`x3upper`	This is the upper constraint on x3
`axislabs`	This is a vector of text labels for the x1, x2 and x3 axis.
`cornerlabs`	This is a vector of text labels for the x1, x2 and x3 vertices.

Note

This function calls MixturePlot. If either des and x,y,z are missing no design points will be plotted, and if x1lower, x1upper, etc. are all zero no constraints will be plotted.

Author(s)

John S. Lawson <lawson@byu.edu>

References

1. Piepel, G. F. "Programs for Generating Extreme Vertices and Centroids of Linearly Consrtained Experimental Regions" *Journal of Quality Technology*, Vol 20, No. 2, pp. 125-139, 1988.

Examples

```
dat<-SCD(3)
DesignPoints(des=dat)

x1<-c(1,0,0,.5,.5, 0,.33333)
x2<-c(0,1,0,.5,0,.5,.33333)
x3<-c(0,0,1,0,.5,.5,.33333)
DesignPoints(x=x3,y=x2,z=x1)

dat<-data.frame(x1,x2,x3)
DesignPoints(des=dat)

DesignPoints(x1lower=0,x1upper=.8,x2lower=.10,x2upper=.95,x3lower=.05,x3upper=.50)
```

Effplot *This function creates mixture effect plots*

Description

This function makes effect plots using the Cox or Pieple directions in constrained mixture space.

Usage

```
Effplot(des=NULL,nfac=3,mod=1,dir=1,ufunc=NULL,dimensions = list(NULL),
x1=c(0,1),x2=c(0,1),x3=c(0,1),x4=c(0,1),x5=c(0,1),x6=c(0,1),x7=c(0,1),
x8=c(0,1),x9=c(0,1),x10=c(0,1),x11=c(0,1),x12=c(0,1))
```

Arguments

des	data frame containing the design points and response data for a mixture experiment. The data frame must contain the variables x1, x2 ...xn for the mixture variables, and y for the response. n must be between 2 and 12. Only effect plots for linear models can be made when the number of factors is greater than 6.
nfac	The number of mixture components in the model.
mod	an interger representing the model to be traced: 1 for a linear model, 2 for a quadratic model, and 3 for a special cubic model, 4 for a user model.
dir	an interger representing the direction for which the effect plot is made: 1 for Pieple direction, 2 for Cox direction.

ufunc	A user function, this should be an lm object.
dimensions	A vector of names of mixture components in the lm object if it is supplied.
x1	a vector containing lower and upper constraints on x1.
x2	a vector containing lower and upper constraints on x2.
x3	a vector containing lower and upper constraints on x3.
x4	a vector containing lower and upper constraints on x4.
x5	a vector containing lower and upper constraints on x5.
x6	a vector containing lower and upper constraints on x6.
x7	a vector containing lower and upper constraints on x7.
x8	a vector containing lower and upper constraints on x8.
x9	a vector containing lower and upper constraints on x9.
x10	a vector containing lower and upper constraints on x10.
x11	a vector containing lower and upper constraints on x11.
x12	a vector containing lower and upper constraints on x12.

Value

PX	This is a matrix containing the coordinates of the effect plot traces that are plotted.

Note

This function calls the function crvtave to get the design centroid from cnvrt.

Author(s)

John S. Lawson <lawson@byu.edu>

References

1. Piepel, G. F. "Measuring Component Effects in Constrained Mixture Experiments" *Technometrics*, Vol 25, pp. 97-105, 1982.

Examples

```
#Example from Li, Tolley, Lee(2010) response is perm
x1<-c(.572,.358,.286,.286,.286,.143,.357)
x2<-c(.214,.428,.500,.357,.214,.500,.500)
x3<-c(.214,.214,.214,.357,.500,.357,.143)
y<-c(7.7,18.4,24.2,9.8,5.9,23.0,19.4)
des<-data.frame(x1,x2,x3,y)
Effplot(des,2,2)
```

```
#Example from Snee, Marquart(1976)
x1<-c(.1,.1,.1,.15,.1,.1,.1,.4,.35,.30,.1,.45,.45,.45,.45,.45,.259,.259,.259,.259)
x2<-c(.5,.05,.5,.05,.05,.5,.05,.05,.05,.5,.5,.05,.2,.15,.25,.1,.222,.222,.222,.222)
x3<-c(0,0,0,0,.1,.1,.1,.1,.1,0,.1,0,0,0,.1,.1,.05,.05,.05,.05)
x4<-c(0,0,.1,.1,0,.1,.1,.1,.1,0,0,0,.1,.1,0,0,.05,.05,.05,.05)
x5<-c(.1,.55,.1,.6,.55,.1,.55,.1,.1,.1,.2,.45,.1,.1,.1,.1,.244,.244,.244,.244)
x6<-c(.2,.2,.2,.05,.2,.05,.05,.2,.2,.05,.05,.05,.05,.2,.05,.2,.125,.125,.125,.125)
x7<-c(.05,.05,0,.05,0,0,0,.05,.05,0,.05,0,.05,0,.05,0,.025,.025,.025,.025)
x8<-c(.05,.05,0,0,0,.05,.05,0,.05,.05,0,0,.05,0,0,.05,.025,.025,.025,.025)
y<-c(30,113,17,94,89,18,90,20,21,15,28,48,18,7,16,19,38,30,35,40)
des<-data.frame(x1,x2,x3,x4,x5,x6,x7,x8,y)
Effplot(des,mod=1,dir=1)



# Weed control example from Lawson & Erjavec
x1<-c(1,0,0,.5,.5,0,.33333,.33333,.33333)
x2<-c(0,1,0,.5,0,.5,.33333,.33333,.33333)
x3<-c(0,0,1,0,.5,.5,.33333,.33333,.33333)
y<-c(73,68,80,77,86,75,92,93,88)
des<-data.frame(x1,x2,x3,y)
Effplot(des,3)



# Polvoron Example from Lawson
des<-Xvert(x1=c(0,.8),x2=c(.10,.95),x3=c(.05,.50),ndm=1)
dat<-as.matrix(des)
# remove the edge centroid at the top
dat<-dat[c(1:6,8:11), ]
# add two more centroids
dat<-rbind(dat,dat[10, ],dat[10,])
# response vector
y<-c(5.75,3.69,5.33,5.68,3.85,3.83,5.88,5.87,5.23,6.54,6.82,6.41)
# make the data frame for plotting
des<-data.frame(dat[,1:3],y)
Effplot(des,3)
```

`Eflags` *Loads compiled fortran in shared file cnvrt and returs the error messages*

Description

This function loads and runs the compiled fortran code cnvrt and prints error messages. cnvrt is Piepel's 1988 JQT fortran code for extreme vertices designs.

Usage

```
Eflags(ndm,nvrr,ncon2,rtheta2)
```

Arguments

ndm	This is the order of centroids desired (0=none, 1=edge centroids, 2=face centroids etc.)
nvrr	This is the number of mixture variables (maximum is 12)
ncon2	This is the number of constraints (maximum is 45)
rtheta2	This is the constraint matrix stored as a vector of columns.

Value

ifa	This is the vector of error flags. A negative value for flag 1 indicates that there are inconsistent constraints. A negative value for flag2 indicates there are two many vertices and centroids, this program only works when # vertices + # centroids <=1000. A negative value for flag 3 indicates an error encountered when calling subroutine allnr.

Note

This function is called by the function crtave.

Author(s)

John S. Lawson <lawson@byu.edu>

References

1. Piepel, G. F. "Programs for Generating Extreme Vertices and Centroids of Linearly Consrtained Experimental Regions" *Journal of Quality Technology*, Vol 20, No. 2, pp. 125-139, 1988.

Fillv *This function duplicates SAS macro adxfill.*

Description

This function creates interior points in a mixture design by averaging all possible pairs of design points.

Usage

```
Fillv(nfac,des)
```

Arguments

nfac	an integer representing the number of mixture variables in the design
des	a data frame containing a mixture design created by one of the functions SLD, SCD or Xvert

Author(s)

John S. Lawson <lawson@byu.edu>

Examples

```
# Example 1 fills interior of Simplex Lattice Design
des<-SLD(3,3)
DesignPoints(des)
des2<-Fillv(3,des)
DesignPoints(des2)

# Example 2 fills interior of Simplex Centroid Design
des<-SCD(4)
Fillv(4,des)

# Example 3 fills interior of Extreme vertices design
ev<-Xvert(3,x1=c(0,.1),x2=c(0,.1),x3=c(0,1),ndm=1)
ev2<-Fillv(3,ev)
```

MixturePlot *This function makes contour plots in the simplex mixture space.*

Description

This function makes contour plots in the simplex mixture space, it also can draw constraint lines and show design points.

Usage

```
MixturePlot(x,y,z,w,des,res,lims,color.palette,constrts,contrs,n.breaks,levels,
            cols,despts,mod,x3lab,x2lab,x1lab, corner.labs,
            colorkey=list(dx=0.04,x0=0.95,y0=0.45,y1=0.90,add=TRUE,mode="all"),
            pseudo=FALSE,user.func=NULL)
```

Arguments

x	x3 locations for known points
y	x2 locations for known points
z	x1 locations for known points

w	y locations for known points
des	data frame with x1,x2,x3, and y locations for known points
res	number of color blocks between 0 and 1 of x
lims	vector of lower and upper constraints for x1,x2,x3
color.palette	is the color palette to use
constrts	if TRUE constraints found in lines will be added to the graph
contrs	if TRUE contour lines will be added to the graph
n.breaks	number of breaks between levels, this is used if levels is not specified
levels	vector of contour levels to be plotted
cols	if TRUE regions between contour lines will be colored
despts	if TRUE plots the design points in data frame des
mod	is an indicator for the model 1=linear, 2=quadratic, 3=special cubic, or by default=NA to signify to signify the model is supplied in the user.func
x3lab	label for the x3 axis
x2lab	label for the x2 axis
x1lab	label for the x1 axis
corner.labs	labels for x3, x2 and x1 vertices
colorkey	a list with the location of the color key
pseudo	if pseudo=TRUE uses pseudo components to zoom in on constrained region. By default pseudo=FALSE
user.func	a function supplied by the user that takes as arguments a dataframe called 'grid' containing columnns 'x', 'y', and 'z' and returns a predicted 'w' for each row in 'grid'.

Author(s)

John S. Lawson <lawson@byu.edu>

References

1. Cornell, J. A. *Experiments with Mixtures: Models and Analysis of Mixture Data*, John Wiley & Sons, New York, third edition, 2002.

2. See R Ternary Level Plot Function http://www.siftp.net/index.shtml

Examples

```
##Usage and Examples - Example from page 458 DAE with SAS
dat = data.frame(
          "x1"=c(1,.8,.6,.5,.5,.33333,.3,.3,.1,.1,0,0,0),
          "x2"=c(0,.1,.2,0,.5,.33333,.2,.5,.1,.8,0,.5,1),
          "x3"=c(0,.1,.2,.5,0,.33333,.5,.2,.8,.1,1.0,.5,0),
          "y"=c(48.7,49.5,50.2,52.8,49.3,51.1,52.7,50.3,60.7,49.9,64.9,53.5,50.6)
                )
```

```
MixturePlot(dat$x3,dat$x2,dat$x1,dat$y, x3lab="Fraction x3",
  x2lab="Fraction x2", x1lab="Fraction x1", corner.labs=c("x3","x2","x1"),
  constrts=FALSE,contrs=TRUE,cols=TRUE, mod=2,n.breaks=9)

# Weed control example from Lawson & Erjavec
x1<-c(1,0,0,.5,.5,0,.33333,.33333,.33333)
x2<-c(0,1,0,.5,0,.5,.33333,.33333,.33333)
x3<-c(0,0,1,0,.5,.5,.33333,.33333,.33333)
y<-c(73,68,80,77,86,75,92,93,88)
des<-data.frame(x1,x2,x3,y)
MixturePlot(des=des,x3lab="Fraction C",x2lab="Fraction B",
   x1lab="Fraction A",corner.labs=c("C","B","A"),mod=3,n.breaks=5,cols=TRUE)

# User defined equation from Lawson and Willden
mymod=function(grid){
x=grid$x #Glycine
y=grid$y #Saccharine
z=grid$z  #Enhancer
w=12.27*z + 10.27*y + 6.27*z + 29.10*z*y + 25.10*z*x + 9.10*y*x +
547.03*z^2*y*x - 388.97*z*y^2*x - 676.97*z*y*x^2
return(w)}
# Create vectors of design points for each variable
x1=c(0,0,1,0,1/2,1/2,1/3,1/6,1/6,2/3)
x2=c(0,1,0,1/2,0,1/2,1/3,1/6,2/3,1/6)
x3=c(1,0,0,1/2,1/2,0,1/3,2/3,1/6,1/6)
MixturePlot(x1,x2,x3,n.breaks=10,user.func=mymod,constrts=FALSE,
cols=FALSE,despts=TRUE,x1lab="Fraction of Glycine (X1)",
x2lab="Fraction of Saccharin (X2)",x3lab="Fraction of Enhancer (X3)")
```

ModelPlot *This function makes contour plots of a user supplied function in the simplex mixture space.*

Description

This function makes contour plots in the simplex mixture space, it also can draw constraint lines and show design points.

Usage

```
ModelPlot(user.func = NULL, dimensions = list(x1=NULL,x2=NULL,x3=NULL),
          slice=NULL,lims=rep(0,6), constraints = FALSE,
          constraint.pars = list(lty=2,lwd=2),
          contour = FALSE, contour.pars = list(lwd=0.5,cex.lab=1.3),
          cuts = 10,at = NULL, res=300, pseudo=FALSE,
          fill=FALSE, color.palette = heat.colors,
          main=NULL, axislabs=c("Fraction X1","Fraction X2","Fraction X3"),
          axislab.pars = list(),
          axislab.offset=0,
```

```
            cornerlabs = c("X1", "X2", "X3"),
            cornerlab.pars = list(),
            grid=TRUE, grid.pars = list(col='darkgrey',lty=3,lwd=0.5),
            colorkey = FALSE,
            labels=TRUE, label.style="align", ...)
```

Arguments

user.func	function supplied by the user that takes as arguments a dataframe called 'grid' and returns the predictions. Typically, this will be a wrapper function for the predict() (e.g. predict(model,newdata=grid)). Additional arguments for the user.func besides 'grid' will typically be used to fix values of process variables.
dimensions	list argument that specifies the mixture variables to be plotted on the ternary plot. Values must correspond to variable names from the user-supplied model.
slice	list argument that specifies the value of fixed mixture components.
lims	vector of lower and upper constraints for ternary plot components (top, left, right).
constraints	if TRUE constraints found in lims will be added to the graph.
constraint.pars	list of graphical parameters controlling the appearance of the constraint lines.
contour	if TRUE contour lines will be added to the graph.
contour.pars	list of graphical parameters controlling the appearance of the contour lines.
cuts	number of breaks between levels (used for contours if 'at' not specified).
at	list of contour levels (e.g. at=c(1,3,5,10) will draw contours at those heights). Overrides cuts argument.
res	resolution of the grid. Corresponds to number equally spaced values along the baseline of the simplex.
pseudo	if TRUE uses pseudo components to zoom in on constrained region. Will create the smallest equilateral triangle that still contains the whole constrained region.
fill	if TRUE regions between contour lines will be colored.
color.palette	is the color palette to use.
main	character value for main title or list containing character value and graphical parameters (e.g. main=list("main title",cex=2)).
axislabs	character vector of axis labels for ternary components.
axislab.pars	list of graphical parameters controlling the appearance of the axislabels.
axislab.offset	numeric value that creates or eliminates space between the angled axislabels and the tickmarks. Prevents axis labels from overlapping with tickmarks. Typically, absolute value would not exceed 0.05.
cornerlabs	character vector of corner labels for x1, x2 and x3 vertices.
cornerlab.pars	list of graphical parameters controlling the appearance of the axislabels.
grid	logical argument. If true, adds grid lines to the ternary plot.
grid.pars	list of graphical parameters controlling the appearance of the gridlines.

colorkey	logical or list of parameters. See levelplot documentation for more details.
labels	logical argument. If true, labels contour lines.
label.style	controls placement of contour labels. Choose from "mixed","flat", or "align." See panel.levelplot documentation for more details.
...	expressions evaluated in the context of the function that creates the grid. Typically, will be used to fix values of process variables.

Author(s)

Cameron Willden <ccwillden@gmail.com>

References

1. Cornell, J. A. *Experiments with Mixtures: Models and Analysis of Mixture Data*, John Wiley & Sons, New York, third edition, 2002.

2. See R Ternary Level Plot Function http://www.siftp.net/index.shtml

Examples

```
mite<-SCD(4)
yavg<-c(1.8,25.4,28.6,38.5,4.9,3.1,28.7,3.4,37.4,10.7,22.0,2.6,2.4,
11.1,0.8)
mitemd<-lm(yavg~-1+x1+x2+x3+x4+x1:x2+x1:x3+x1:x4+x2:x3+x2:x4+x3:x4+
x1:x2:x3+x1:x2:x4+x1:x3:x4+x2:x3:x4+x1:x2:x3:x4, data=cbind(mite,yavg))
miteprd=function(grid)
 { ygrid=predict(mitemd,grid)
     return(ygrid)
 }
ModelPlot(miteprd,dimensions = list(x1="x1",x2="x2",x3="x3"),
slice = list(x4=0.0), main="Dibrom = 0.0",
constraints=FALSE,contour=TRUE,at=c(5,10,15,20),fill=FALSE,
axislabs=c("Vendex","Omite","Kelthane"),
cornerlabs = c("X1", "X2", "X3"),pseudo=FALSE)

ModelPlot(miteprd,dimensions = list(x1="x1",x2="x2",x3="x3"),
slice = list(x4=0.4), main="Dibrom = 0.4",
constraints=FALSE,contour=TRUE,cuts=5,fill=FALSE,
axislabs=c("Vendex","Omite","Kelthane"),
cornerlabs = c("X1", "X2", "X3"),pseudo=FALSE)
```

Nrows *Loads compiled fortran in shared file cnvrt and returns the number of rows in the resulting design*

Description

This function loads and runs the compiled fortran code cnvrt. cnvrt is Piepel's 1988 JQT fortran code for extreme vertices designs.

Usage

```
Nrows(ndm,nvrr,ncon2,rtheta2)
```

Arguments

ndm	This is the order of centroids desired (0=none, 1=edge centroids, 2=face centroids etc.)
nvrr	This is the number of mixture variables (maximum is 12)
ncon2	This is the number of constraints (maximum is 45)
rtheta2	This is the constraint matrix stored as a vector of columns.

Value

nvrtr

nvrtr	This is the number of rows in rxvt the matrix of extreme vertices and centroids

Note

This function is called by the function crtave.

Author(s)

John S. Lawson <lawson@byu.edu>

References

1. Piepel, G. F. "Programs for Generating Extreme Vertices and Centroids of Linearly Consrtained Experimental Regions" *Journal of Quality Technology*, Vol 20, No. 2, pp. 125-139, 1988.

SCD *This function creates simplex centroid mixture designs*

Description

This function creates simplex centroid designs in unconstrained mixture experiment space.

Usage

```
SCD(fac)
```

Arguments

fac	This is the number of factors

Value

SC	This is a data frame containing the simplex centroid design. The columns are labeled x1, x2 ...xn, where n is the number of mixture variables.

Author(s)

John S. Lawson <lawson@byu.edu>

References

1. Cornell, J. A. *Experiments with Mixtures: Models and Analysis of Mixture Data*, John Wiley & Sons, New York, third edition, 2002.

Examples

```
SCD(3)

des<-SCD(5)

des<-SCD(12)
```

SLD *This function creates simplex lattice mixture designs*

Description

This function creates simplex lattice designs in unconstrained mixture experiment space.

Usage

```
SLD(fac, lev)
```

Arguments

fac	This is the number of factors, this must be between 2 and 12
lev	This is the number of levels

Value

SL	This is a data frame containing the simplex lattice design. The columns are labeled x1, x2 ...xn, where n is the number of mixture variables.

Author(s)

John S. Lawson <lawson@byu.edu>

References

1. Cornell, J. A. *Experiments with Mixtures: Models and Analysis of Mixture Data*, John Wiley \& Sons, New York, third edition, 2002.

Examples

```
des<-SLD(3,2)

des<-SLD(4,3)
```

SneeMq — *Data from Snee and Marquart's Screening Experiment with contreined mixture components*

Description

This is an .rda file design and response.

Usage

```
data(SneeMq)
```

Format

An 16 x 9 data frame

Source

source

References

Snee, D. D. and Marquardt D. W. (1976) Screening Concepts and designs for experiments with mixtures, *Technometrics*, Vol. 18, pp 19-29.

`Vertcen` *Loads compiled fortran in shared file cnvrt*

Description

This function loads and runs the compiled fortran code cnvrt. cnvrt is Pieple's 1988 JQT fortran code for extreme vertices designs.

Usage

```
Vertcen(ndm,nvrr,ncon2,rtheta2)
```

Arguments

`ndm`	This is the order of centroids desired (0=none, 1=edge centroids, 2=face centroids etc.)
`nvrr`	This is the number of mixture variables (maximum is 12)
`ncon2`	This is the number of constraints (maximum is 45)
`rtheta2`	This is the constraint matrix stored as a vector of columns.

Value

`rxvt`	This is the matrix of vertices and centroids stored as a vector of columns.

Note

This function is called by the function crtave.

Author(s)

John S. Lawson <lawson@byu.edu>

References

1. Piepel, G. F. "Programs for Generating Extreme Vertices and Centroids of Linearly Consrtained Experimental Regions" *Journal of Quality Technology*, Vol 20, No. 2, pp. 125-139, 1988.

`Xvert` *This function creates an extreme vertices design in a constrained mixture space.*

Description

This function calls the function crvtave to create an extreme vertices design in a constrained mixture space. If there are only three factors the function DesignPoints is called to plot the results.

Usage

```
Xvert(nfac,x1,x2,x3,x4,x5,x6,x7,x8,x9,x10,x11,x12,nlc=0,lb=c(0,0),ub=c(0,0),
coef=c(0,0),ndm)
```

Arguments

`nfac`	an integer representing the number of mixture variables in the design
`x1`	a vector containing lower and upper constraints on x1
`x2`	a vector containing lower and upper constraints on x2
`x3`	a vector containing lower and upper constraints on x3
`x4`	a vector containing lower and upper constraints on x4
`x5`	a vector containing lower and upper constraints on x5
`x6`	a vector containing lower and upper constraints on x6
`x7`	a vector containing lower and upper constraints on x7
`x8`	a vector containing lower and upper constraints on x8
`x9`	a vector containing lower and upper constraints on x9
`x10`	a vector containing lower and upper constraints on x10
`x11`	a vector containing lower and upper constraints on x11
`x12`	a vector containing lower and upper constraints on x12
`nlc`	the number of linear constraints
`lb`	a vector of length nlc containing the lower bounds for the linear constraints
`ub`	a vector of length nlc containing the upper bounds for the linear constraints
`coef`	a vector of length nlc*nfac containing the coefficients of the components of the linear constraints
`ndm`	an integer representing the highest order of centroids requested. An overall centroid is always included, 0 indicates no other centroids will be created, 1 indicates edge centroids are requested, etc.

Note

This function calls crvtave. If the number of factors is 3, the function DesignPoints is called to graph the results.

Author(s)

John S. Lawson <lawson@byu.edu>

References

1. Piepel, G. F. "Programs for Generating Extreme Vertices and Centroids of Linearly Consrtained Experimental Regions" *Journal of Quality Technology*, Vol 20, No. 2, pp. 125-139, 1988.

Examples

```
# Polvoron Example from Lawson
des<-Xvert(3,x1=c(0,.8),x2=c(.10,.95),x3=c(.05,.50),ndm=2)

# Exercise 11.3 DAE with SAS
Xvert(3,x1=c(.2,.8),x2=c(.1,.4),x3=c(.1,.5))

#Snee Marquardt(1976) example
Xvert(8,x1=c(.1,.45),x2=c(.05,.50),x3=c(0,.10),x4=c(0,.4),x5=c(.1,.6),x6=c(.05,.2),
      x7=c(0,.05),x8=c(0,.05),ndm=0)

# Example page 465
exvert<-Xvert(4,x1=c(.124,.188),x2=c(.064,.128),x3=c(.374,.438),x4=c(.374,.438),ndm=2)

# Example from Piepel 1988
des<-Xvert(nfac=3, x1=c(.1,.5), x2=c(.1,.7), x3=c(0,.7), nlc=2, lb=c(.9,.4),
ub=c(.95,0), coef=c(.85,.90,1,.70,0,1), ndm=1 )
```

Index