Package ‘vdg’

February 20, 2015

Type Package
Title Variance Dispersion Graphs and Fraction of Design Space Plots
Version 1.0.1
Date 2015-02-13
Author Pieter Schoonees [aut, cre, cph]
Maintainer Pieter Schoonees <schoonees@gmail.com>
Depends R (>= 3.1.2), parallel, ggplot2, quantreg
Imports proxy, splines
Suggests rsm, Vdgraph
Description Facilities for constructing variance dispersion graphs, fraction-of-design-space plots and similar graphics for exploring the properties of experimental designs. The design region is explored via random sampling, which allows for more flexibility than traditional variance dispersion graphs. A formula interface is leveraged to provide access to complex model formulae. Graphics can be constructed simultaneously for multiple experimental designs and/or multiple model formulae. Instead of using pointwise optimization to find the minimum and maximum scaled prediction variance curves, which can be inaccurate and time consuming, this package uses quantile regression as an alternative.
License GPL (>= 2)
LazyData yes
NeedsCompilation yes
Repository CRAN
Date/Publication 2015-02-13 11:03:49

R topics documented:

vdg-package .................................................. 2
GI54 ......................................................... 2
LHS ......................................................... 3
meanspv ..................................................... 4
plot.spv .................................................... 5
print.spv ................................................... 7
vdg-package

Variance Dispersion Graphs (VDGs) and Fraction of Design Space (FDS) Plots

Description

Produces VDG, FDS and related plots for assessing experimental designs.

Author(s)

Pieter C. Schoonees <schoonees@gmail.com>

GJ54

Design from Goos & Jones

Description

This data frame contains the design of Table 5.4 in Goos & Jones (2011).

Usage

GJ54

Format

a data frame of 15 runs in two variables: Time (seconds) and Temperature (Kelvin)

References

Description

Different versions of latin hypercube sampling (LHS): ordinary LHS, midpoint LHS, symmetric LHS or randomized symmetric LHS. LHS is a method for constructing space-filling designs. They can be more efficient for hypercuboidal design regions than other sampling methods.

Usage

\[ \text{LHS}(n, m = 3, \text{lim} = c(-1, 1)) \]
\[ \text{MLHS}(n, m = 3, \text{lim} = c(-1, 1)) \]
\[ \text{SLHS}(n, m = 3, \text{lim} = c(-1, 1)) \]
\[ \text{RSLHS}(n, m = 3, \text{lim} = c(-1, 1)) \]

Arguments

- \( n \)   number of design points to generate
- \( m \)   number of design factors
- \( \text{lim} \)   limits of the coordinates in all dimensions

Value

Matrix with samples as rows.

Author(s)

Pieter C. Schoonees

Examples

```r
set.seed(1234)
pts <- seq(-1, 1, length = 11)

# Ordinary LHS
samp <- LHS(n = 10, m = 2)
plot(samp, main = "LHS")
abline(h = pts, v = pts, col = "lightgrey")

# Midpoint LHS
samp <- MLHS(n = 10, m = 2)
plot(samp, main = "MLHS")
abline(h = pts, v = pts, col = "lightgrey")
```
# Symmetric LHS
samp <- SLHS(n = 10, m = 2)
plot(samp, main = "SLHS")
abline(h = pts, v = pts, col = "lightgrey")

# Randomized Symmetric LHS
samp <- RSLHS(n = 10, m = 2)
plot(samp, main = "RSLHS")
abline(h = pts, v = pts, col = "lightgrey")

## meanspv

### Compute Mean Spherical SPV

#### Description
Computes the matrix of spherical region moments for a given model formula and a vector of radii, and uses this to calculate the mean spherical SPV for each of the radii. The function expmat calculates the matrix containing the exponents of each model factor within each model term as columns. Only simple formulae are allowed for. Only products of terms should be included in calls to 1. The power operator ^ should be used instead of sqrt. Models should contain only monomial terms.

#### Usage

```r
meanspv(formula, radii, FtF.inv, n)
```

```r
expmat(formula)
```

#### Arguments

- `formula`: model formula
- `radii`: numeric vector or radii at which to calculate the matrix of spherical region moments
- `FtF.inv`: inverse of F'F, where F is the design matrix
- `n`: integer giving the number of design runs

#### Examples

```r
f1 <- formula(~ x1*x2)
expmat(f1)
f2 <- update(f1, ~ . + I(x1^2) + I(x2^2))
expmat(f2)
f3 <- update(f2, ~ . + I(x2^0.4))
expmat(f3)
f4 <- update(f3, ~ . + I(x1^2):I(x2^2))
expmat(f4)
f5 <- update(f4, ~ . + I(x1^3*x2^0.5))
expmat(f5)
```
Description

Produce Variance Dispersion Graphs and/or Fraction of Design Space plots for experimental designs. There are methods for the S3 classes `spv`, `spvlist`, `spvforlist` and `spvlistforlist` – see `spv`.

Usage

```r
## S3 method for class 'spv'
plot(x, which = 1L:5L, np = 50,
     alpha = 7/sqrt(length(x$spv)), points.colour = "#39BE1",
     points.size = 2, tau = c(0.05, 0.95), radii = 21, hexbin = FALSE,
     bins = 80, df = 10, lines.size = 1, origin = rep(0, ncol(x$sample)),
     method, ...)

## S3 method for class 'spvforlist'
plot(x, which = 1L:5L, np = 50,
     alpha = 7/sqrt(length(x[[1]]$spv)), points.colour = "#39BE1",
     points.size = 2, tau = c(0.05, 0.95), radii = 21, hexbin = FALSE,
     bins = 80, df = 10, lines.size = 1, origin = rep(0,
     ncol(x[[1]]$sample)), method, ...)

## S3 method for class 'spvlist'
plot(x, which = 1L:5L, np = 50,
     alpha = 7/sqrt(length(x[[1]]$spv)), points.colour = "#39BE1",
     points.size = 2, tau = c(0.05, 0.95), radii = 21, hexbin = FALSE,
     bins = 80, VRFDS = FALSE, df = 10, lines.size = 1, origin = rep(0,
     ncol(x[[1]]$sample)), method, ...)

## S3 method for class 'spvlistforlist'
plot(x, which = 1L:5L, np = 50,
     alpha = 7/sqrt(length(x[[1]][[1]]$spv)), points.colour = "#39BE1",
     points.size = 2, tau = c(0.05, 0.95), radii = 21, hexbin = FALSE,
     bins = 80, df = 10, lines.size = 1, origin = rep(0,
     ncol(x[[1]][[1]]$sample)), method, ...)
```

Arguments

- `x`: an object of type `spv` for a single experimental design or an object of type `spvlist` for multiple designs.
- `which`: a vector of integers indicating which plots to produce. A subset of `1L:4L` for `spv` or a subset of `1L:5L` for `spvlist`.
- `np`: scalar; the number of points to use for calculating the fraction of design space criterion.
alpha the alpha transparency coefficient for the plots
points.colour colour for points in scatterplot of SPV against the radius
points.size size for points in scatterplot of SPV against the radius
tau the tau parameter for rq (quantile regression)
radii either a numeric vector containing the radii to use for calculating the mean spherical SPV over the spherical design space, or an integer (length one vector) giving the number of radii to use for calculating the mean spherical SPV. If missing, the mean spherical SPV is not used.
hexbin logical indicating whether hexagonal binning should be used to display density instead of colour transparency
bins argument passed to stat_binhex to determine the number of hexagons used for binning.
df degrees-of-freedom parameter passed to bs
lines.size line size passed to geom_line
origin numeric vector specifying the origin of the design space
method optional; passed to dist to overwrite defaults of "Euclidean" for spherical regions or "supremum" for cubiodal regions
VRFDS logical indicating whether to construct a variance ratio FDS plot or not (only for class spvlist). The first design is used as reference design in case of VRFDS is TRUE
... additional arguments passed to dist

Value

Returns a list of ggplot2 graphical objects (or grobs) with names plot1, plot2 etc. These can be manipulated by changing plot aesthetics and theme elements.

Author(s)

Pieter C. Schoonees

Examples

# Single design (class 'spv')
# Larger n should be used in actual cases
library(rsm)
bbd3 <- as.data.frame(bbd(3)[,3:5])
colnames(bbd3) <- paste0("x", 1:3)
quad.3f <- formula(~ x1*x2*x3 - x1:x2:x3 + I(x1^2) + I(x2^2) + I(x3^2))
set.seed(1234)
out <- spv(n = 1000, design = bbd3, type = "spherical", formula = quad.3f)
out.plot(out)

# List of designs (class 'spvlist')
## Not run:
library(Vdgraph)
print.spv

```
data(SCDH5); data(SCDDL5)
des.list <- list(SCDH5 = SCDH5, SCDDL5 = SCDDL5)
quad.5f <- formula(~ x1 + x2 + x3 + x4 + x5 + x1:x2 + x1:x3 + x1:x4 + x1:x5
                  + x2:x3 + x2:x4 + x2:x5 + x3:x4 + x3:x5 + x4:x5
                  + I(x1^2) + I(x2^2) + I(x3^2) + I(x4^2) + I(x5^2))
out2 <- spv(n = 500, design = des.list, type = "spherical", formula = quad.5f)
out2
plot(out2)

## End(Not run)

# List of formulae (class 'spvforlist')
## Not run:
fact3 <- expand.grid(x1 = c(-1,1), x2 = c(-1, 1), x3 = c(-1,1))
lin.3f <- formula(~ x1 + x2 + x3)
int.3f <- formula(~ (x1+x2+x3)^2)
set.seed(4312)
out3 <- spv(n = 500, design = fact3, type = "cuboidal",
          formula = list(linear = lin.3f, interaction = int.3f))
out3
plot(out3)

## End(Not run)

# List of formulae and designs (class 'spvlistforlist')
## Not run:
fact3.n <- rbind(fact3, 0, 0, 0)
set.seed(4312)
out4 <- spv(n = 200, design = list(factorial = fact3, factorial.with.cntr = fact3.n),
          type = "cuboidal", formula = list(linear = lin.3f, interaction = int.3f))
out4
plot(out4)

## End(Not run)
```

print.spv  

**Print Method for S3 spv classes**

**Description**

Simple print methods for S3 classes spv, spvlist, spvforlist and spvlistforlist. See `plot.spv` for examples.

**Usage**

```
## S3 method for class 'spv'
print(x, ...)

## S3 method for class 'spvforlist'
print(x, ...)
```
## S3 method for class 'spvlist'
print(x, ...)

## S3 method for class 'spvlistforlist'
print(x, ...)

### Arguments

- **x**: Object of class `spv` or `spvlist`
- **...**: Unimplemented

### Author(s)

Pieter C. Schoonees

---

**runif_cube**

*Sampling for hyperspheres/hypercubes*

## Description

Sample uniformly in or on a hyperspheres or hypercubes.

## Usage

```r
runif_cube(n, m = 2, max.dist = 1, at = FALSE, nr.dist = 21)
runif_sphere(n, m = 2, max.radius = sqrt(m), at = FALSE, nr.rad = 21)
```

### Arguments

- **n**: number of points to sample
- **m**: number of design factors
- **max.dist**: maximum distance from origin (L-infinity norm/supremum distance) for the hypercuboidal design region (enveloping hypercube)
- **at**: logical indicating whether to sample on concentric hyperspheres/hypercubes or not. With this option `n` is distributed proportionally across radii / supremum distances so that the density of samples on each concentric hypercube / hypersphere are uniform across the different hyperspheres / hypercubes.
- **nr.dist**: the number of concentric hypercubes to use in case `at` is TRUE
- **max.radius**: maximum radius of the hyperspherical design region (enveloping hypersphere)
- **nr.rad**: number of concentric hyperspherical to sample on in case of `at` being TRUE

### Author(s)

Pieter C. Schoonees
sampler

Examples

```r
set.seed(1234)
sampler(n = 10)
set.seed(1234)
samp <- runif_sphere(n = 500, at = TRUE)
plot(samp, asp = 1)
```

sampler | Sampler Function

Description

This is a wrapper for the sampling functions of the `vdg` package. It extracts design properties from the design passed to it to take appropriate samples.

Usage

```r
sampler(n, design, type = "spherical", at = FALSE, ...)
```

Arguments

<table>
<thead>
<tr>
<th>n</th>
<th>number of points to sample</th>
</tr>
</thead>
<tbody>
<tr>
<td>design</td>
<td>design for which the sample is required (either a matrix or data frame)</td>
</tr>
<tr>
<td>type</td>
<td>type of design region/sampling method. One of &quot;spherical&quot;, &quot;cuboidal&quot;, &quot;lhs&quot;, &quot;mlhs&quot;, &quot;slhs&quot;, or &quot;rslhs&quot;</td>
</tr>
<tr>
<td>at</td>
<td>logical; should sampling be done on the surface of hyperspheres or hypercubes? Not used for LHS methods.</td>
</tr>
<tr>
<td>...</td>
<td>other arguments passed to the underlying sampling functions.</td>
</tr>
</tbody>
</table>

Value

Matrix with samples as rows, with S3 class `smpl`

Author(s)

Pieter C. Schoonees

See Also

`runif_sphere`, `runif_cube`, `LHS`, `MLHS`, `SLHS`, `RSLHS`

Examples

```r
set.seed(1896)
sampler(n = 10, design = expand.grid(x = -1:1, y = -1:1))
```
Calculate the Scaled Prediction Variance (or SPV)

Description
Calculates the SPV for a sample of points in a design region of specified type. Sampling is done by calling \texttt{sampler}.

Usage

\begin{verbatim}
spv(n, design, type = "spherical", formula, at = FALSE, keepfun, sample,
  unscaled = FALSE, ...)

## S3 method for class 'data.frame'
spv(n, design, type = "spherical", formula, at = FALSE,
  keepfun, sample, unscaled = FALSE, ...)

## S3 method for class 'list'
spv(n, design, type = "spherical", formula, at = FALSE,
  keepfun, sample, unscaled = FALSE, ...)

## S3 method for class 'matrix'
spv(n, design, type = "spherical", formula, at = FALSE,
  keepfun, sample, unscaled = FALSE, ...)
\end{verbatim}

Arguments

\begin{itemize}
  \item \texttt{n} \hspace{1cm} \text{number of samples to take}
  \item \texttt{design} \hspace{1cm} \text{a design or list of designs. Each design must be either a matrix or a data.frame or coercible to a data.frame.}
  \item \texttt{type} \hspace{1cm} \text{type of sampling passed to \texttt{sampler}}
  \item \texttt{formula} \hspace{1cm} \text{either a single model formula of a list of formulae}
  \item \texttt{at} \hspace{1cm} \text{only used when type is 'spherical' or 'cuboidal'}
  \item \texttt{keepfun} \hspace{1cm} \text{optional; function operating on the columns of a matrix with the same number of columns as design which return a logical value for including a specific point in the sample or not. Useful for rejection sampling for nonstandard design regions.}
  \item \texttt{sample} \hspace{1cm} \text{optional; if not missing it should contain a matrix or data.frame containing points sampled over the required design region. If it is not missing, no further sampling will be done: the SPV is simply evaluated at these points.}
  \item \texttt{unscaled} \hspace{1cm} \text{logical indicating whether to use the unscaled prediction variance (UPV) instead of the scale prediction variance (SPV)}
  \item ... \hspace{1cm} \text{additional arguments passed to \texttt{sampler}}
\end{itemize}
Value

Object of class ‘spv’, ‘spvlist’, ‘spvforlist’ or ‘spvlistforlist’, depending on whether single designs/formulas are passed or lists of these.

Author(s)

Pieter C. Schoonees

See Also

plot.spv for more examples

Examples

# Single design (class 'spv')
library(rsm)
bbd3 <- as.data.frame(bbd(3)[,3:5])
colnames(bbd3) <- paste0("x", 1:3)
quad.3f <- formula(~(x1 + x2 + x3)^2 - x1:x2:x3)
out <- spv(n = 1000, design = bbd3, type = "spherical", formula = quad.3f)
out

stdrange

Standardize or Unstandardize the Column Range

Description

Simple functions for rescaling a data matrix to a coded design and back. stdrange converts the design in actual measurements into a coded design, while ustdrange reverses the process (if the correct arguments are given).

Usage

stdrange(x, mins = apply(x, 2, min), maxs = apply(x, 2, max))
ustdrange(x, mins, maxs)

Arguments

x
matrix containing the design, or an object coercible to a matrix.
mins
vector of original values, one for each column, which should be recoded to the value -1; or which have already been recoded to -1. This and the next argument are both recycled if not of the correct length.
maxs
vector of original values which should be recoded as 1, or which have already been recoded to 1/

Author(s)

Pieter C. Schoonees
Index

*Topic design
  LHS, 3
*Topic hplot
  plot.spv, 5
*Topic multivariate
  spv, 10
*Topic package
  vdg-package, 2
*Topic print
  print.spv, 7
  ^, 4
bs, 6
dist, 6
expmat (meanspv), 4
geom_line, 6
ggplot2, 6
GJ54, 2
I, 4
LHS, 3, 9
meanspv, 4
MLHS, 9
MLHS (LHS), 3
plot.spv, 5, 7, 11
plot.spvforlist (plot.spv), 5
plot.spvlist (plot.spv), 5
plot.spvlistforlist (plot.spv), 5
print.spv, 7
print.spvforlist (print.spv), 7
print.spvlist (print.spv), 7
print.spvlistforlist (print.spv), 7
rq, 6
RSLHS, 9

RSLHS (LHS), 3
runif_cube, 8, 9
runif_sphere, 9
runif_sphere (runif_cube), 8
sampler, 9, 10
SLHS, 9
SLHS (LHS), 3
spv, 5, 10
 sqrt, 4
stat_binhex, 6
stdrange, 11
ustdrange (stdrange), 11
vdg (vdg-package), 2
vdg-package, 2