Package ‘demu’

October 13, 2022

Type Package

Title Optimal Design Emulators via Point Processes

Version 0.3.0

Date 2019-12-29

Author Matthew T. Pratola <mpratola@stat.osu.edu> [aut, cre, cph]

Maintainer Matthew T. Pratola <mpratola@stat.osu.edu>

Description Implements the Determinantal point process (DPP) based optimal design emulator described in Pratola, Lin and Craigmile (2018) <arXiv:1804.02089> for Gaussian process regression models. See <http://www.matthewpratola.com/software> for more information and examples.

Depends R (>= 3.2.3)

Imports Rcpp (>= 0.12.12), stats, fields, spam, Matrix, ClusterR

LinkingTo Rcpp, RcppArmadillo

License AGPL-3

LazyData true

NeedsCompilation yes

Repository CRAN

Date/Publication 2020-01-13 09:20:10 UTC

R topics documented:

`demu-package` — `demu` *is an open-source R package implementing a Gaussian process optimal design emulator based on Determinantal point processes.*

Description

`demu` implements a determinantal point process emulator for probabilistically sampling optimal designs for Gaussian process (GP) regression models. Currently, `demu` is a proof of concept implementation that implements basic DPP sampling, conditional DPP sampling for drawing designs of fixed size `n`, sequential DPP sampling to build designs iteratively and a faster C++ implementation of the conditional DPP sampler using sparse matrices. The package supports popular stationary correlation functions commonly used in GP regression models, including the Gaussian and Wendland correlation functions.

Details

The main model fitting functions in the package include `sim.dpp.modal()` for dense correlation matrices and `sim.dpp.modal.fast()` for sparse correlation matrices. These functions use a grid-based approximation to sample from the relevant DPP model.

Author(s)

Matthew T. Pratola <mpratola@stat.osu.edu> [aut, cre, cph]

References

Pratola, Matthew T., Lin, C. Devon, and Craigmile, Peter. (2018) Optimal Design Emulators: A Point Process Approach. *arXiv:1804.02089*.

See Also

`sim.dpp.modal`,`sim.dpp.modal.fast`,`sim.dpp.modal.seq`,`sim.dpp.modal.fast.seq` generalized.wendland — *Calculate the correlation matrix according to the generalized Wendland model.*

Description

generalized.wendland() is a helper function that constructs a correlation matrix according to the generalized Wendland model with lengthscales given by the parameter vector theta. When kap=0 the correlation model corresponds to the Askey correlation model. The design must have been already formated in distlist format using the function makedistlist().

Usage

```
generalized.wendland(l.d,theta, kap)
```

Arguments

l.d	Current design distance matrices in distlist format
theta	A vector of range parameters
kap	A non-negative scalar parameter

Value

A list containing the constructed correlation matrix.

See Also

demu-package rhomat matern32 matern52 wendland1 wendland2

Examples

```
library(demu)

design=matrix(runif(10,0,1),ncol=2,nrow=5)
theta=0.3
kap=3
l.d=makedistlist(design)
R=generalized.wendland(l.d,theta,kap)$R
R
```

getranges	*Get variable ranges from a design matrix.*

Description

getranges() is a helper function to get the lower/upper bounds of variables in a design matrix, used for rescaling the inputs to the $[0, 1]$ hypercube.

Usage

```
getranges(design)
```

Arguments

design	An $n \times p$ matrix of input settings

Value

A $p \times 2$ matrix with the lower and upper bounds (rounded to nearest integer value) of all p variables in the design matrix.

Examples

```
library(demu)

design=matrix(runif(10,1,5),ncol=2,nrow=5)
getranges(design)
```

makedistlist	*Make list of distance matrices for calculating GP correlation matrices.*

Description

makedistlist() is a helper function used to setup the difference matrices that are used by the DPP models.

Usage

```
makedistlist(design)
```

Arguments

design	An $n \times p$ matrix of input settings

Value

A list of p matrices, each of dimension $n \times n$ that contain the outer subtractions of each variable in the design matrix.

See Also

getranges scaledesign

Examples

```
library(demu)

design=matrix(runif(10,1,5),ncol=2,nrow=5)
r=getranges(design)
design=scaledesign(design,r)
l.v=makedistlist(design)
```

`matern32` *Calculate the correlation matrix according to the Matern model with* $\nu = 3/2$.

Description

`matern32()` is a helper function that constructs a correlation matrix according to the Matern model with parameter $\nu = 3/2$ and lengthscales given by the parameter vector `theta`. The design must have been already formated in distlist format using the function `makedistlist()`.

Usage

```
matern32(l.d,theta)
```

Arguments

`l.d`	Current design distance matrices in distlist format
`theta`	A vector of range parameters

Value

A list containing the constructed correlation matrix.

See Also

demu-package rhomat matern52 wendland1 wendland2 generalized.wendland

Examples

```
library(demu)

design=matrix(runif(10,0,1),ncol=2,nrow=5)
theta=rep(0.2,2)
l.d=makedistlist(design)
R=matern32(l.d,theta)$R
R
```

`matern52` *Calculate the correlation matrix according to the Matern model with $\nu = 5/2$.*

Description

`matern52()` is a helper function that constructs a correlation matrix according to the Matern model with parameter $\nu = 5/2$ and lengthscales given by the parameter vector `theta`. The design must have been already formated in distlist format using the function `makedistlist()`.

Usage

```
matern52(l.d,theta)
```

Arguments

`l.d`	Current design distance matrices in distlist format
`theta`	A vector of range parameters

Value

A list containing the constructed correlation matrix.

See Also

demu-package rhomat matern32 wendland1 wendland2 generalized.wendland

Examples

```
library(demu)

design=matrix(runif(10,0,1),ncol=2,nrow=5)
theta=rep(0.2,2)
l.d=makedistlist(design)
R=matern52(l.d,theta)$R
R
```

`remove.projections` *Identify candidate points making up all marginal subprojections of an existing design.*

Description

`remove.projections()` is a helper function to identify all lower-dimensional marginal projection points of the existing design points indexed by `curpts`. This function can be used to remove a subset of points from the *candidate set* in order to enforce non-collapsingness of when sequentially adding design points.

Usage

```
remove.projections(curpts,X)
```

Arguments

`curpts`	Indices of points currently in the design
`X`	An `n x p` matrix of all candidate points

Value

A list containing the vector `curpts`, the vector `projpts` which contains the identified projection points of the current design, and `allpts`.

See Also

`demu-package` `sim.dpp.modal.seq`

Examples

```
library(demu)

n1=3
n2=3
n3=3
rho=rep(1e-10,2)
ngrid=10

x=seq(0,1,length=ngrid)
X=as.matrix(expand.grid(x,x))
l.d=makedistlist(X)

# Initial design
R=rhomat(l.d,rho)$R
pts.1=sim.dpp.modal(R,n1)
pts.1.proj=remove.projections(pts.1,X)

# Plot - design points in black, design+projection points in grey.
```

```
#plot(X,xlim=c(0,1),ylim=c(0,1))
#points(X[pts.1.proj$projpts,],pch=20,cex=2,col="grey")
#points(X[pts.1,],pch=20,cex=2)
```

---

rhomat *Calculate the correlation matrix according to the squared exponential family of models.*

Description

`rhomat()` is a helper function that constructs a correlation matrix according to the squared exponential model with parameterized by correlation parameters `rho` taking values in [0,1) and the exponent parameter `alpha`. The default of `alpha=2` results in the Gaussian correlation while selecting `alpha=1` corresponds to the Exponential correlation model. The design must have been already formated in distlist format using the function `makedistlist()`.

Usage

```
rhomat(l.d,rho,alpha=2)
```

Arguments

`l.d`	Current design distance matrices in distlist format
`rho`	A vector of correlation parameters taking on values in [0,1)
`alpha`	Exponent parameter

Value

A list containing the constructed correlation matrix.

See Also

`demu-package` `matern52` `wendland1` `wendland2` `generalized.wendland`

Examples

```
library(demu)

design=matrix(runif(10,0,1),ncol=2,nrow=5)
rho=rep(0.01,2)
l.d=makedistlist(design)
R=rhomat(l.d,rho)$R
R
```

scaledesign — *Rescale a design matrix to the [0,1] hypercube.*

Description

`scaledesign()` is a helper function to rescale a design to the $[0, 1]$ hypercube using variable ranges previously extracted by a call to `getranges()`.

Usage

```
scaledesign(design,r)
```

Arguments

`design`	An $n \times p$ matrix of input settings
`r`	An $p \times 2$ matrix of variable ranges extracted from `getranges()`

Value

A $n \times p$ design matrix rescaled to the $[0, 1]$ hypercube.

See Also

`unscalemat`

Examples

```
library(demu)

design=matrix(runif(10,1,5),ncol=2,nrow=5)
r=getranges(design)
scaledesign(design,r)
```

sim.dpp.modal — *Draw samples from the conditional DPP design emulator.*

Description

`sim.dpp.modal()` uses the DPP-based design emulator of Pratola et al. (2018) to draw a sample of the `n`-run optimal design for a Gaussian process regression model with stationary correlation function $r(x, x')$, where the entries of `R` are formed by evaluating $r(x, x')$ over a grid of candidate locations.

Usage

```
sim.dpp.modal(R,n=0,eigs=NULL)
```

Arguments

R	A correlation matrix evaluated over a grid of candidate design sites.
n	Size of the design to sample.
eigs	One can alternatively pass the pre-computed eigendecomposition of the correlation matrix instead of R.

Details

For more details on the method, see Pratola et al. (2018). Detailed examples demonstrating the method are available at http://www.matthewpratola.com/software.

Value

A vector of indices to the sampled design sites.

References

Pratola, Matthew T., Lin, C. Devon, and Craigmile, Peter. (2018) Optimal Design Emulators: A Point Process Approach. *arXiv:1804.02089*.

See Also

demu-package sim.dpp.modal.fast sim.dpp.modal.seq

Examples

```
library(demu)

# candidate grid
ngrid=20
x=seq(0,1,length=ngrid)
X=as.matrix(expand.grid(x,x))
l.d=makedistlist(X)

# draw design from DPP mode
n=21
rho=0.01
R=rhomat(l.d,rep(rho,2))$R
pts=sim.dpp.modal(R,n)

# Could plot the result:
# plot(X,xlim=c(0,1),ylim=c(0,1))
# points(X[pts,],pch=20)
```

`sim.dpp.modal.fast` *Draw samples from the conditional DPP design emulator.*

Description

`sim.dpp.modal.fast()` is similar to `sim.dpp.modal` but is a C++ codepath that makes use of SPAM's sparse matrices to enable faster computation. It implements the DPP-based design emulator of Pratola et al. (2018) to draw a sample of the `n`-run optimal design for a Gaussian process regression model with compact correlation function $r(x, x')$, where the entries of `R` are formed by evaluating $r(x, x')$ over a grid of candidate locations.

Usage

```
sim.dpp.modal.fast(R,n)
```

Arguments

`R`	A sparse correlation matrix evaluated over a grid of candidate design sites. The sparse matrix should be of type `dgCMatrix` (see package spam).
`n`	Size of the design to sample.

Details

For more details on the method, see Pratola et al. (2018). Detailed examples demonstrating the method are available at `http://www.matthewpratola.com/software`.

Value

A vector of indices to the sampled design sites.

References

Pratola, Matthew T., Lin, C. Devon, and Craigmile, Peter. (2018) Optimal Design Emulators: A Point Process Approach. *arXiv:1804.02089*.

See Also

`demu-package` `sim.dpp.modal` `sim.dpp.modal.seq`

Examples

```
library(demu)
library(fields)
library(spam)
library(Matrix)
library(Rcpp)

# candidate grid
```

```
ngrid=20
x=seq(0,1,length=ngrid)
X=as.matrix(expand.grid(x,x))

# draw design from DPP mode
n=21
theta=0.39
R.spam=wendland.cov(X,X,theta=theta,k=3)
R=as.dgCMatrix.spam(R.spam)
rm(R.spam)
pts=sim.dpp.modal.fast(R,n)

# Could plot the result:
# plot(X,xlim=c(0,1),ylim=c(0,1))
# points(X[pts,],pch=20)
```

`sim.dpp.modal.fast.seq` *Draw sequential samples from the conditional DPP given previously sampled points already in the design.*

Description

`sim.dpp.modal.fast.seq()` is similar to `sim.dpp.modal.fast` but sequentially selects `n` additional points to add to the design given that the points in `curpts` are alread in the design from previous sequential iterations. It uses the C++ codepath that makes use of SPAM's sparse matrices to enable faster computation. It implements the DPP-based design emulator of Pratola et al. (2018) to draw a sequential sample of the `n`-run additional optimal design points for a Gaussian process regression model with compact correlation function $r(x, x')$, where the entries of `R` are formed by evaluating $r(x, x')$ over a grid of candidate locations.

Usage

```
sim.dpp.modal.fast.seq(curpts, R,n)
```

Arguments

`curpts`	A vector of indices to the candidate points that already appear in the design.
`R`	A sparse correlation matrix evaluated over a grid of candidate design sites. The sparse matrix should be of type `dgCMatrix` (see package spam).
`n`	Size of the design to sample.

Details

For more details on the method, see Pratola et al. (2018). Detailed examples demonstrating the method are available at `http://www.matthewpratola.com/software`.

Value

A vector of indices to the sampled design sites.

References

Pratola, Matthew T., Lin, C. Devon, and Craigmile, Peter. (2018) Optimal Design Emulators: A Point Process Approach. *arXiv:1804.02089*.

See Also

demu-package sim.dpp.modal.fast sim.dpp.modal

Examples

```
library(demu)
library(fields)
library(spam)
library(Matrix)

n1=3
n2=3
n3=3
rho=0.2
ngrid=10

x=seq(0,1,length=ngrid)
X=as.matrix(expand.grid(x,x))
l.d=makedistlist(X)

# Initial design
R.spam=wendland.cov(X,X,theta=rho,k=3)
R=as.dgCMatrix.spam(R.spam)
pts.1=sim.dpp.modal.fast(R,n1)
pts.1.proj=remove.projections(pts.1,X)

# Next sequential step, removing projections
pts.2=sim.dpp.modal.fast.seq(pts.1.proj$allpts,R,n2)
design=c(pts.1,pts.2$pts.new)
pts.2.proj=remove.projections(design,X)

# Next sequential step, removing projections
pts.3=sim.dpp.modal.fast.seq(pts.2.proj$allpts,R,n3)
design=c(design,pts.3$pts.new)


# Or, starting with the initial design, don't remove projections
pts.2=sim.dpp.modal.fast.seq(pts.1,R,n2)
designB=c(pts.1,pts.2$pts.new)

pts.3=sim.dpp.modal.fast.seq(designB,R,n3)
designB=c(designB,pts.3$pts.new)
```

```
# Plot the result:
#par(mfrow=c(1,3))
#plot(X,xlim=c(0,1),ylim=c(0,1),main="Initial Design")
#points(X[pts.1,],pch=20,cex=2)
#
#plot(X,xlim=c(0,1),ylim=c(0,1),main="+3x2 remove projections")
#points(X[design,],pch=20,cex=2)
#
#plot(X,xlim=c(0,1),ylim=c(0,1),main="+3x2 not removing projections")
#points(X[designB,],pch=20,cex=2)
```

`sim.dpp.modal.np` *Draw samples from the conditional DPP design emulator using a kmeans-based Nystrom approximation.*

Description

`sim.dpp.modal.np()` uses `sim.dpp.modal.nystrom.kmeans()` to draw a design of `n` points in `p` dimensions using the kmeans-based Nystrom approximation of Zhang and Kwok (2010) and the DPP-based design emulator of Pratola et al. (2018). The design constructed assumes a Gaussian process regression model with stationary correlation function $r(x, x')$, where the entries of `R` are formed by evaluating $r(x, x')$ over a set of landmarks chosen by the kmeans algorithm, and the resulting eigenvectors are projected into the higher dimensional space using the Nystrom approximation. Additional options for `sim.dpp.modal.nystrom.kmeans()` can be passed to alter the construction of the landmark set.

Usage

```
sim.dpp.modal.np(n,p,N,rho,m=max(ceiling(N*0.1),n),...)
```

Arguments

`n`	Size of the desired design.
`p`	Dimension of the desired design.
`N`	Number of kernel approximation points drawn uniformly from the p-dimensional design space.
`rho`	The $p \times 1$ parameter vector for the Gaussian correlation model.
`m`	Number of landmark points to use in constructing the kmeans-based Nystrom approximation.
`...`	Additional options to pass to `sim.dpp.modal.nystrom.kmeans()` for drawing the design.

Details

For more details on the method, see Pratola et al. (2018). Detailed examples demonstrating the method are available at http://www.matthewpratola.com/software.

Value

A list containing a matrix which is the union of the $N \times p$ uniformly sampled kernel approximation points and the `m` selected landmark sites, and the indices into this matrix of the selected design sites.

References

Pratola, Matthew T., Lin, C. Devon, and Craigmile, Peter. (2018) Optimal Design Emulators: A Point Process Approach. *arXiv:1804.02089*.

Zhang, Kai and Kwok, James T. (2010) Clustered Nystrom method for large scale manifold learning and dimension reduction. *IEEE Transactions on Neural Networks*, **21.10**, 1576–1587. doi: 10.1109/TNN.2010.2064786

See Also

`demu-package` `sim.dpp.modal` `sim.dpp.modal.nystrom.kmeans`

Examples

```
library(demu)

n=50
p=5
N=500
rho=rep(0.01,5)
samp=sim.dpp.modal.np(n,p,N,rho)

# Could plot the result:
# pchvec=rep(1,nrow(samp$X))
# pchvec[samp$pts]=20
# cexvec=rep(0.1,nrow(samp$X))
# cexvec[samp$pts]=1
# colvec=rep("black",nrow(samp$X))
# colvec[samp$pts]="red"
# pairs(samp$X,pch=pchvec,cex=cexvec,col=colvec,xlim=c(0,1),ylim=c(0,1))
```

`sim.dpp.modal.nystrom` *Draw samples from the conditional DPP design emulator using grid-based Nystrom approximation.*

Description

`sim.dpp.modal.nystrom()` uses the DPP-based design emulator of Pratola et al. (2018) to draw a sample of the `n`-run optimal design for a Gaussian process regression model with stationary correlation function $r(x, x')$, where the entries of `R` are formed by evaluating $r(x, x')$ over a grid of candidate locations. This function uses a grid-based Nystrom approximation based on the passed matrix `X` to avoid constructing a large correlation matrix if dimension `ngrid^p` and its subsequent eigendecomposition.

Usage

```
sim.dpp.modal.nystrom(Xin,rho,n=0,ngrid=NULL,method="Nystrom")
```

Arguments

`Xin`	A initial $n \times p$ matrix of points.
`rho`	The $p \times 1$ parameter vector for the Gaussian correlation model.
`n`	Size of the design to sample from the candidate grid.
`ngrid`	Size of the candidate grid will be `ngrid^p`.
`method`	Type of approximation to use. Currently only supports “Nystrom”.

Details

For more details on the method, see Pratola et al. (2018). Detailed examples demonstrating the method are available at http://www.matthewpratola.com/software.

Value

A list containing the candidate points constructed and the points selected as the design sites from this candidate set as well as their indices.

References

Pratola, Matthew T., Lin, C. Devon, and Craigmile, Peter. (2018) Optimal Design Emulators: A Point Process Approach. *arXiv:1804.02089*.

See Also

`demu-package` `sim.dpp.modal` `sim.dpp.modal.nystrom.kmeans`

Examples

```
library(demu)

# Starting design
X=matrix(runif(10*2),ncol=2)
rho=rep(0.01,2)
n=10
ngrid=11
samp=sim.dpp.modal.nystrom(X,rho,n,ngrid)
samp$design

# Could plot the result:
# plot(samp$X,xlim=c(0,1),ylim=c(0,1))
# points(samp$X[samp$pts,],pch=20)
```

sim.dpp.modal.nystrom.kmeans

Subsample an observational dataset using the conditional DPP design emulator with a kmeans-based Nystrom approximation.

Description

sim.dpp.modal.nystrom.kmeans() uses the kmeans-based Nystrom approximation of Zhang and Kwok (2010) to select n design sites from the observational dataset Xin using the DPP-based design emulator of Pratola et al. (2018). The design constructed assumes a Gaussian process regression model with stationary correlation function $r(x,x')$, where the entries of R are formed by evaluating $r(x,x')$ over a set of landmarks chosen by the kmeans algorithm, and the resulting eigenvectors are projected into the higher dimensional space using the Nystrom approximation. Additional options for the MiniBatchKmeans() algorithm from package ClusterR can be passed to alter the construction of the landmark set.

Usage

```
sim.dpp.modal.nystrom.kmeans(Xin,rho=rep(0.01,ncol(Xin)),
  n,m=max(ceiling(nrow(Xin)*0.1),n),method="KmeansNystrom",
  initializer="kmeans++",...)
```

Arguments

Xin	An $n \times p$ dataset of observations from which we want to draw subsamples.
n	Size of the designed subsample to draw from Xall.
rho	The $p \times 1$ parameter vector for the Gaussian correlation model.
m	Number of landmark points to use in constructing the kmeans-based Nystrom approximation.
method	Type of approximation to use. Currently only supports “KmeansNystrom”.
initializer	Initialization to use in the Kmeans algorithm, default is “kmeans++”.
...	Additional options to pass to MiniBatchKmeans() for selecting the landmark points.

Details

For more details on the method, see Pratola et al. (2018). Detailed examples demonstrating the method are available at http://www.matthewpratola.com/software.

Value

A list containing a matrix which is the union of the observation matrix Xin and selected landmark sites, the indices into this matrix of the selected design sites as well as matrix of the design sites.

References

Pratola, Matthew T., Lin, C. Devon, and Craigmile, Peter. (2018) Optimal Design Emulators: A Point Process Approach. *arXiv:1804.02089*.

Zhang, Kai and Kwok, James T. (2010) Clustered Nystrom method for large scale manifold learning and dimension reduction. *IEEE Transactions on Neural Networks*, **21.10**, 1576–1587. doi: 10.1109/TNN.2010.2064786

See Also

demu-package sim.dpp.modal sim.dpp.modal.nystrom

Examples

```
library(demu)

# Fake dataset in 5 dimensions
X=matrix(runif(500*5),ncol=5)
rho=rep(0.01,5)
n=50
samp=sim.dpp.modal.nystrom.kmeans(X,rho,n)
samp$design

# Could plot the result:
# pchvec=rep(1,nrow(samp$X))
# pchvec[samp$pts]=20
# cexvec=rep(0.1,nrow(samp$X))
# cexvec[samp$pts]=1
# colvec=rep("black",nrow(samp$X))
# colvec[samp$pts]="red"
# pairs(samp$X,pch=pchvec,cex=cexvec,col=colvec,xlim=c(0,1),ylim=c(0,1))
```

---

sim.dpp.modal.seq — *Draw sequential samples from the conditional DPP given previously sampled points already in the design.*

Description

`sim.dpp.modal.seq()` is similar to `sim.dpp.modal` but sequentially selects `n` additional points to add to the design given that the points in `curpts` are alread in the design from previous sequential iterations. It implements the DPP-based design emulator of Pratola et al. (2018) to draw a sequential sample of `n`-run additional optimal design points for a Gaussian process regression model with correlation function $r(x, x')$, where the entries of `R` are formed by evaluating $r(x, x')$ over a grid of candidate locations. As is typical, `R` is formed based on *all* of the candidate grid points.

Usage

```
sim.dpp.modal.seq(curpts, R, n)
```

Arguments

curpts	A vector of indices to the candidate points that already appear in the design.
R	A correlation matrix evaluated over a grid of candidate design sites.
n	Size of the design to sample.

Details

For more details on the method, see Pratola et al. (2018). Detailed examples demonstrating the method are available at http://www.matthewpratola.com/software.

Value

A vector of indices to add to the existing design sites.

References

Pratola, Matthew T., Lin, C. Devon, and Craigmile, Peter. (2018) Optimal Design Emulators: A Point Process Approach. *arXiv:1804.02089*.

See Also

demu-package sim.dpp.modal sim.dpp.modal.fast

Examples

```
library(demu)

n1=3
n2=3
n3=3
rho=rep(1e-10,2)
ngrid=10

x=seq(0,1,length=ngrid)
X=as.matrix(expand.grid(x,x))
l.d=makedistlist(X)

# Initial design
R=rhomat(l.d,rho)$R
pts.1=sim.dpp.modal(R,n1)
pts.1.proj=remove.projections(pts.1,X)

# Next sequential step, removing projections
pts.2=sim.dpp.modal.seq(pts.1.proj$allpts,R,n2)
design=c(pts.1,pts.2$pts.new)
pts.2.proj=remove.projections(design,X)

# Next sequential step, removing projections
pts.3=sim.dpp.modal.seq(pts.2.proj$allpts,R,n3)
design=c(design,pts.3$pts.new)
```

```
# Or, starting with the initial design, don't remove projections
pts.2=sim.dpp.modal.seq(pts.1,R,n2)
designB=c(pts.1,pts.2$pts.new)

pts.3=sim.dpp.modal.seq(designB,R,n3)
designB=c(designB,pts.3$pts.new)


# Plot the result:
#par(mfrow=c(1,3))
#plot(X,xlim=c(0,1),ylim=c(0,1),main="Initial Design")
#points(X[pts.1,],pch=20,cex=2)
#
#plot(X,xlim=c(0,1),ylim=c(0,1),main="+3x2 remove projections")
#points(X[design,],pch=20,cex=2)
#
#plot(X,xlim=c(0,1),ylim=c(0,1),main="+3x2 not removing projections")
#points(X[designB,],pch=20,cex=2)
```

---

unscalemat *Unscale a matrix back to its original ranges.*

Description

unscalemat() is a helper function to rescale a matrix back to its original ranges. Typically this is used to rescale the posterior samples of the parameters back to their original scale.

Usage

```
unscalemat(mat,r)
```

Arguments

mat	An $n \times p$ matrix of numbers scaled to the $[0,1]$ hypercube
r	An $p \times 2$ matrix of the original ranges of the variables

Value

A $n \times p$ matrix with variables rescaled back to their original ranges, as specified by ranges.

See Also

getranges scaledesign

Examples

```
library(demu)

design=matrix(runif(10,1,5),ncol=2,nrow=5)
r=getranges(design)
design=scaledesign(design,r)
unscalemat(design,r)
```

---

wendland1 *Calculate the correlation matrix according to the Wendland1 model.*

Description

`wendland1()` is a helper function that constructs a correlation matrix according to the Wendland 1 model with lengthscales given by the parameter vector `theta`. The design must have been already formated in distlist format using the function `makedistlist()`.

Usage

```
wendland1(l.d,theta)
```

Arguments

`l.d`	Current design distance matrices in distlist format
`theta`	A vector of range parameters

Value

A list containing the constructed correlation matrix.

See Also

demu-package rhomat matern32 matern52 wendland2 generalized.wendland

Examples

```
library(demu)

design=matrix(runif(10,0,1),ncol=2,nrow=5)
theta=rep(0.3,2)
l.d=makedistlist(design)
R=wendland1(l.d,theta)$R
R
```

wendland2 *Calculate the correlation matrix according to the Wendland2 model.*

Description

wendland2() is a helper function that constructs a correlation matrix according to the Wendland 2 model with lengthscales given by the parameter vector theta. The design must have been already formated in distlist format using the function makedistlist().

Usage

```
wendland2(l.d,theta)
```

Arguments

l.d	Current design distance matrices in distlist format
theta	A vector of range parameters

Value

A list containing the constructed correlation matrix.

See Also

demu-package rhomat matern32 matern52 wendland1 generalized.wendland

Examples

```
library(demu)

design=matrix(runif(10,0,1),ncol=2,nrow=5)
theta=rep(0.3,2)
l.d=makedistlist(design)
R=wendland2(l.d,theta)$R
R
```

Index