Package ‘CGGP’

May 8, 2021

Type Package

Title Composite Grid Gaussian Processes

Version 1.0.3

Description Run computer experiments using the adaptive composite grid algorithm with a Gaussian process model.
The algorithm works best when running an experiment that can evaluate thousands of points from a deterministic computer simulation.
This package is an implementation of a forthcoming paper by Plumlee, Erickson, Ankenman, et al. For a preprint of the paper, contact the maintainer of this package.

License GPL-3

Imports Rcpp (>= 0.12.18)

LinkingTo Rcpp, RcppArmadillo

Suggests testthat, covr, ggplot2, reshape2, plyr, MASS, rmarkdown, knitr

URL https://github.com/CollinErickson/CGGP

BugReports https://github.com/CollinErickson/CGGP/issues

Encoding UTF-8

RooxygenNote 7.1.1

VignetteBuilder knitr

NeedsCompilation yes

Author Collin Erickson [aut, cre],
Matthew Plumlee [aut]

Maintainer Collin Erickson <collinberickson@gmail.com>

Repository CRAN

Date/Publication 2021-05-08 15:10:02 UTC
R topics documented:

CGGP ......................................................... 3
CGGPappend .................................................. 3
CGGPcreate .................................................. 4
CGGPfit ....................................................... 5
CGGPplotblocks .............................................. 6
CGGPplotblockselection .................................... 7
CGGPplotcorr ................................................ 8
CGGPplotheat ............................................... 9
CGGPplottheta .............................................. 10
CGGPplotsamplesneglogpost ................................ 11
CGGPplotslice ............................................... 11
CGGPplottheta .............................................. 13
CGGPplotvariogram ......................................... 13
CGGPvalplot ............................................... 14
CGGPvalstats ............................................... 15
CGGP_internal_calcMSE .................................... 16
CGGP_internal_calcMSEde ................................... 17
CGGP_internal_calcpw ...................................... 17
CGGP_internal_calcpwanddpw ............................... 18
CGGP_internal_CorrMatCauchy ............................. 19
CGGP_internal_CorrMatCauchySQ ........................ 20
CGGP_internal_CorrMatCauchySQT ......................... 21
CGGP_internal_CorrMatGaussian ........................... 22
CGGP_internal_CorrMatMatern32 ........................ 23
CGGP_internal_CorrMatMatern52 ........................ 24
CGGP_internal_CorrMatPowerExp .......................... 25
CGGP_internal_CorrMatWendland0 ......................... 26
CGGP_internal_CorrMatWendland1 ......................... 27
CGGP_internal_CorrMatWendland2 ......................... 28
CGGP_internal_gneglogpost ................................ 29
CGGP_internal_MSEPredcalc ............................... 30
CGGP_internal_neglogpost ................................ 31
CGGP_internal_set_corr ................................... 32
plot.CGGP .................................................. 32
predict.CGGP .............................................. 33
print.CGGP ................................................ 34
rcpp_fastmatclcr .......................................... 35
rcpp_fastmatclcranddclcr ................................. 35
rcpp_gkronDBS ............................................ 36
rcpp_kronDBS ............................................. 36
valplot ..................................................... 37
valstats .................................................... 38

Index ....................................................... 40
**Description**

The CGGP package implements the method presented in Plumlee et al. (2019).

**CGGP functions**

The CGGP functions: CGGPcreate, CGGPfit, CGGPappend, and CGGPpred

---

**CGGPappend**

*Add points to CGGP*

---

**Description**

Add ‘batchsize’ points to ‘SG’ using ‘theta’.

**Usage**

`CGGPappend(CGGP, batchsize, selectionmethod = "MAP")`

**Arguments**

- `CGGP`: Sparse grid object
- `batchsize`: Number of points to add
- `selectionmethod`: How points will be selected: one of ‘UCB’, ‘TS’, ‘MAP’, ‘Oldest’, ‘Random’, or ‘Lowest’. ‘UCB’ uses Upper Confidence Bound estimates for the parameters. ‘TS’ uses Thompson sampling, a random sample from the posterior. ‘MAP’ uses maximum a posteriori parameter estimates. ‘Oldest’ adds the block that has been available the longest. ‘Random’ adds a random block. ‘Lowest’ adds the block with the lowest sum of index levels. ‘UCB’ and ‘TS’ are based on bandit algorithms and account for uncertainty in the parameter estimates, but are the slowest. ‘MAP’ is fast but doesn’t account for parameter uncertainty. The other three are naive methods that are not adaptive and won’t perform well.

**Value**

SG with new points added.

**See Also**

Other CGGP core functions: `CGGPcreate()`, `CGGPfit()`, `predict.CGGP()`
Examples

```r
SG <- CGGPcreate(d=3, batchsize=100)
y <- apply(SG$design, 1, function(x){x[1]+x[2]^2})
SG <- CGGPfit(SG, Y=y)
SG <- CGGPappend(CGGP=SG, batchsize=20, selectionmethod="MAP")
```

---

CGGPcreate

Create sparse grid GP

Description

Create sparse grid GP

Usage

```r
CGGPcreate(
  d,
  batchsize,
  corr = "PowerExponential",
  grid_sizes = c(1, 2, 4, 8, 12, 20, 28, 32),
  Xs = NULL,
  Ys = NULL,
  HandlingSuppData = "Correct",
  supp_args = list()
)
```

Arguments

- `d` Input dimension
- `batchsize` Number added to design each batch for now only on predictions
- `corr` Name of correlation function to use. Must be one of "CauchySQT", "CauchySQ", "Cauchy", "Gaussian", "PowerExp", "Matern32", "Matern52".
- `grid_sizes` Size of grid refinements.
- `Xs` Supplemental X data
- `Ys` Supplemental Y data
- `HandlingSuppData` How should supplementary data be handled? * Correct: full likelihood with grid and supplemental data * Only: only use supplemental data * Ignore: ignore supplemental data
- `supp_args` Arguments used to fit if Xs and Ys are given

Value

CGGP
CGGPfit

See Also
Other CGGP core functions: CGGPappend(), CGGPfit(), predict.CGGP()

Examples

CGGPcreate(d=8,200)

CGGPfit
Update CGGP model given data

Description
This function will update the GP parameters for a CGGP design.

Usage

CGGPfit(
  CGGP,
  Y,
  Xs = NULL,
  Ys = NULL,
  theta0 = pmax(pmin(CGGP$thetaMAP, 0.8), -0.8),
  HandlingSuppData = CGGP$HandlingSuppData,
  separateoutputparameterdimensions = is.matrix(CGGP$thetaMAP),
  set_thetaMAP_to,
  corr,
  Ynew,
)

Arguments

  CGGP     Sparse grid objects
  Y        Output values calculated at CGGP$design
  Xs       Supplemental X matrix
  Ys       Supplemental Y values
  theta0   Initial theta
  HandlingSuppData
          How should supplementary data be handled? * Correct: full likelihood with
grid and supplemental data * Only: only use supplemental data * Ignore: ignore
supplemental data
  separateoutputparameterdimensions
          If multiple output dimensions, should separate parameters be fit to each dimen-
sion?
  set_thetaMAP_to
          Value for thetaMAP to be set to
  corr
          Will update correlation function, if left missing it will be same as last time.
  Ynew
          Values of `CGGP$design_unevaluated`
CGGPplotblocks

**Value**

Updated CGGP object fit to data given

**See Also**

Other CGGP core functions: `CGGPappend()`, `CGGPcreate()`, `predict.CGGP()`

**Examples**

```r
cg <- CGGPcreate(d=3, batchsize=100)
y <- apply(cg$design, 1, function(x){x[1]+x[2]^2})
cg <- CGGPfit(CGGP=cg, Y=y)
```

---

CGGPblock plot

**Description**

Plot the 2D projections of the blocks of an CGGP object.

**Usage**

```r
CGGPplotblocks(CGGP, singleplot = TRUE)
```

**Arguments**

- `CGGP`: CGGP object
- `singleplot`: If only two dimensions, should a single plot be made?

**Value**

ggplot2 plot

**See Also**

Other CGGP plot functions: `CGGPplotcorr()`, `CGGPplotheat()`, `CGGPplothist()`, `CGGPplotsamplesneglogpost()`, `CGGPplotslice()`, `CGGPplottheta()`, `CGGPplotvariogram()`, `CGGPvalplot()`

**Examples**

```r
# The first and fourth dimensions are most active and will have greater depth
ss <- CGGPcreate(d=5, batchsize=50)
f <- function(x) {cos(2*pi*x[1]*3) + x[3]*exp(4*x[4])}
ss <- CGGPfit(ss, Y=apply(ss$design, 1, f))
ss <- CGGPappend(CGGP=ss, batchsize=100)
CGGPplotblocks(ss)
```
mat <- matrix(c(1,1,2,2,1,2,1,3), ncol=2, byrow=TRUE)
CGGPplotblocks(mat)

---

CGGPplotblockselection

*Plot CGGP block selection over time*

---

**Description**

Shows the order in which blocks were selected for each dimension. Gives an idea of how the selections change over time.

**Usage**

`CGGPplotblockselection(CGGP, indims)`

**Arguments**

- **CGGP**: CGGP object
- **indims**: Which input dimensions should be shown?

**Value**

ggplot2 object

**Examples**

```r
gs <- CGGPcreate(d=3, batchsize=100)
# All dimensions will look similar
CGGPplotblockselection(gs)

# You need to append with CGGPappend after fitting to see a difference
f <- function(x){x[1]^1.2}
y <- apply(gs$design, 1, f)
gs <- CGGPfit(gs, Y=y)
gs <- CGGPappend(gs, 100)
# Now you will see higher for X1 from 100 to 200 while others remain low.
CGGPplotblockselection(gs)
```
CGGPplotcorr

**Plot correlation samples**

**Description**

Plot samples for a given correlation function and parameters. Useful for getting an idea of what the correlation parameters mean in terms of smoothness.

**Usage**

```r
CGGPplotcorr(
    Corr = CGGP_internal_CorrMatGaussian,
    theta = NULL,
    numlines = 20,
    outdims = NULL,
    zero = TRUE
)
```

**Arguments**

- **Corr**: Correlation function or CGGP object. If CGGP object, it will make plots for thetaMAP, the max a posteriori theta.
- **theta**: Parameters for Corr
- **numlines**: Number of sample paths to draw
- **outdims**: Which output dimensions should be used?
- **zero**: Should the sample paths start at y=0?

**Value**

Plot

**See Also**

Other CGGP plot functions: `CGGPplotblocks()`, `CGGPplotheat()`, `CGGPplothist()`, `CGGPplotsamplesneglogpost()`, `CGGPplotslice()`, `CGGPplottheta()`, `CGGPplotvariogram()`, `CGGPvalplot()`

**Examples**

```r
CGGPplotcorr()
CGGPplotcorr(theta=c(-2,-1,0,1))

SG <- CGGPcreate(d=3, batchsize=100)
f <- function(x){x[1]^1.2+sin(2*pi*x[2]*3)}
y <- apply(SG$design, 1, f)
SG <- CGGPfit(SG, Y=y)
CGGPplotcorr(SG)
```
Description

The values on the diagonal are largest design depth for that dimension. The off-diagonal values are the largest design depth that both dimensions have been measured at simultaneously. A greater depth means that more points have been measured along that dimension or two-dimensional subspace.

Usage

CGGPplotheat(CGGP)

Arguments

CGGP 
CGGP object

Value

A heat map made from ggplot2

References

https://stackoverflow.com/questions/14290364/heatmap-with-values-ggplot2

See Also

Other CGGP plot functions: CGGPplotblocks(), CGGPplotcorr(), CGGPplothist(), CGGPplotsamplesneglogpost(), CGGPplotslice(), CGGPplottheta(), CGGPplotvariogram(), CGGPvalplot()
CGGPplothist

Histogram of measurements at each design depth of each input dimension

Description
A greater design depth signifies a more important dimension. Thus a larger right tail on the histogram are more important variables.

Usage
CGGPplothist(CGGP, ylog = TRUE)

Arguments
CGGP CGGP object
ylog Should the y axis be put on a log scale?

Value
Histogram plot made using ggplot2

See Also
Other CGGP plot functions: CGGPplotblocks(), CGGPplotcorr(), CGGPplotheat(), CGGPplotsamplesneglogpost(), CGGPplotslice(), CGGPplottheta(), CGGPplotvariogram(), CGGPvalplot()

Examples

# All dimensions should look similar
d <- 8
SG = CGGPcreate(d,201)
CGGPplothist(SG)
CGGPplothist(SG, ylog=FALSE)

# The first dimension is more active and will have greater depth
f <- function(x) {sin(x[1]^0.6*5)}
SG <- CGGPcreate(d=5, batchsize=100)
SG <- CGGPfit(SG, apply(SG$design, 1, f))
SG <- CGGPappend(CGGP=SG, batchsize=1000)
CGGPplothist(SG)
CGGPplotsamplesneglogpost

Plot negative log posterior likelihood of samples

Description
Plot negative log posterior likelihood of samples

Usage
CGGPplotsamplesneglogpost(CGGP)

Arguments
CGGP
CGGP object

Value
ggplot2 object

See Also
Other CGGP plot functions: CGGPplotblocks(), CGGPplotcorr(), CGGPplotheat(), CGGPplothist(), CGGPplotslice(), CGGPplottheta(), CGGPplotvariogram(), CGGPvalplot()

Examples
gs <- CGGPcreate(d=3, batchsize=100)
f <- function(x){x[1]^1.2+x[3]^0.4*sin(2*pi*x[2]^2)*3 + .1*exp(3*x[3])}
y <- apply(gs$design, 1, f)
gs <- CGGPfit(gs, Y=y)
CGGPplotsamplesneglogpost(gs)

CGGPplotslice

CGGP slice plot

Description
Show prediction plots when varying over only one dimension. Most useful when setting all values to 0.5 because it will have the most points.
Usage

CGGPplotslice(
  CGGP,
  proj = 0.5,
  np = 300,
  color = "pink",
  outdims,
  scales = "free_y",
  facet = "grid"
)

Arguments

CGGP  CGGP object
proj  Point to project onto
np    Number of points to use along each dimension
color Color to make error region
outdims If multiple outputs, which of them should be plotted?
scales Parameter passed to ggplot2::facet_grid()
facet If "grid", will use ggplot2::facet_grid(), if "wrap" will use ggplot2::facet_wrap(). Only applicable for a single output dimension.

Value

ggplot2 object

See Also

Other CGGP plot functions: CGGPplotblocks(), CGGPplotcorr(), CGGPplotheat(), CGGPplothist(), CGGPplotsamplesneglogpost(), CGGPplottheta(), CGGPplotvariogram(), CGGPvalplot()

Examples

d <- 5
s1 <- CGGPcreate(d, 200)
s1 <- CGGPfit(s1, apply(s1$design, 1, f1))
#s1 <- CGGPappend(s1, 200)
#s1 <- CGGPfit(s1, apply(s1$design, 1, f1))
CGGPplotslice(s1)
CGGPplotslice(s1, 0.)
CGGPplotslice(s1, s1$design[nrow(s1$design),])
CGGPplottheta  

Plot theta samples

Description

Plot theta samples

Usage

CGGPplottheta(CGGP)

Arguments

CGGP  
CGGP object

Value

ggplot2 object

See Also

Other CGGP plot functions: CGGPplotblocks(), CGGPplotcorr(), CGGPplotheat(), CGGPplothist(), CGGPplotsamplesneglogpost(), CGGPplotslice(), CGGPplotvariogram(), CGGPvalplot()

Examples

gs <- CGGPcreate(d=3, batchsize=100)
y <- apply(gs$design, 1, f)  
gs <- CGGPfit(gs, Y=y)  
CGGPplottheta(gs)

CGGPplotvariogram  

Plot something similar to a semivariogram

Description

It’s not actually a variogram or semivariogram. It shows how the correlation function falls off as distance increases.

Usage

CGGPplotvariogram(CGGP, facet = 1, outdims = NULL)
**CGGPvalplot**

---

**Description**

Plot validation prediction errors for CGGP object

**Usage**

```
CGGPvalplot(CGGP, Xval, Yval, d = NULL)
```

**Arguments**

- **CGGP**: CGGP object that has been fitted
- **Xval**: X validation data
- **Yval**: Y validation data
- **d**: If output is multivariate, which column to use. Will do all if left as NULL.

**Value**

None, makes a plot

---

**Arguments**

- **CGGP**: CGGP object
- **facet**: How should the plots be faceted? If 1, in a row, if 2, in a column, if 3, wrapped around.
- **outdims**: Which output dimensions should be shown.

**Value**

ggplot2 object

**See Also**

Other CGGP plot functions: `CGGPplotblocks()`, `CGGPplotcorr()`, `CGGPplotheat()`, `CGGPplothist()`, `CGGPplotsamplesneglogpost()`, `CGGPplotslice()`, `CGGPplottheta()`, `CGGPvalplot()`

---

**Examples**

```r
SG <- CGGPcreate(d=3, batchsize=100)
f <- function(x){x[1]^{1.2}+x[3]^{.4}*sin(2*pi*x[2]^2*3) + .1*exp(3*x[3])}
y <- apply(SG$design, 1, f)
SG <- CGGPfit(SG, Y=y)
CGGPplotvariogram(SG)
```
See Also

Other CGGP plot functions: \texttt{CGGPplotblocks()}, \texttt{CGGPplotcorr()}, \texttt{CGGPplotheat()}, \texttt{CGGPplothist()}, \texttt{CGGPplotsamplesneglogpost()}, \texttt{CGGPplotslice()}, \texttt{CGGPplottheta()}, \texttt{CGGPplotvariogram()}

Examples

```r
SG <- CGGPcreate(d=3, batchsize=100)
f1 <- function(x){x[1]+x[2]^2}
y <- apply(SG$design, 1, f1)
SG <- CGGPfit(SG, y)
Xval <- matrix(runif(3*100), ncol=3)
Yval <- apply(Xval, 1, f1)
CGGPvalplot(CGGP=SG, Xval=Xval, Yval=Yval)
```

\texttt{CGGPvalstats} \hspace{1cm} \textit{Calculate stats for CGGP prediction on validation data}

Description

Calculate stats for CGGP prediction on validation data

Usage

\texttt{CGGPvalstats(CGGP, Xval, Yval, bydim = TRUE, \ldots)}

Arguments

\begin{itemize}
  \item \texttt{CGGP} \hspace{1cm} CGGP object
  \item \texttt{Xval} \hspace{1cm} X validation matrix
  \item \texttt{Yval} \hspace{1cm} Y validation data
  \item \texttt{bydim} \hspace{1cm} If multiple outputs, should it be done separately by dimension?
  \item \texttt{\ldots} \hspace{1cm} Passed to valstats, such as which stats to calculate.
\end{itemize}

Value

data frame

Examples

```r
SG <- CGGPcreate(d=3, batchsize=100)
f1 <- function(x){x[1]+x[2]^2}
y <- apply(SG$design, 1, f1)
SG <- CGGPfit(SG, y)
Xval <- matrix(runif(3*100), ncol=3)
Yval <- apply(Xval, 1, f1)
CGGPvalstats(CGGP=SG, Xval=Xval, Yval=Yval)
```
# Multiple outputs
SG <- CGGPcreate(d=3, batchsize=100)
f1 <- function(x){x[1]+x[2]^2}
f2 <- function(x){x[1]^1.3+.4*sin(6*x[2])+10}
y1 <- apply(SG$design, 1, f1)+rnorm(1,0,.01)
y2 <- apply(SG$design, 1, f2)+rnorm(1,0,.01)
y <- cbind(y1, y2)
SG <- CGGPfit(SG, Y=y)
Xval <- matrix(runif(3*100), ncol=3)
Yval <- cbind(apply(Xval, 1, f1),
apply(Xval, 1, f2))
CGGPvalstats(SG, Xval, Yval)
CGGPvalstats(SG, Xval, Yval, bydim=FALSE)

---

CGGP_internal_calcMSE  Calculate MSE over single dimension

**Description**

Calculated using grid of integration points. Can be calculated exactly, but not much reason in 1D.

**Usage**

CGGP_internal_calcMSE(xl, theta, CorrMat)

**Arguments**

- **xl**: Vector of points in 1D
- **theta**: Correlation parameters
- **CorrMat**: Function that gives correlation matrix for vectors of 1D points.

**Value**

MSE value

**Examples**

CGGP_internal_calcMSE(xl=c(0,0.5,.9), theta=c(1,2,3),
CorrMat=CGGP_internal_CorrMatCauchySQT)
**CGGP_internal_calcMSEde**  
*Calculate MSE over blocks*

**Description**

Delta of adding block is product over \(i=1..d\) of \(\text{IMSE}(i,j-1) - \text{IMSE}(i,j)\)

**Usage**

`CGGP_internal_calcMSEde(valsinds, MSE_MAP)`

**Arguments**

- **valsinds**  
  Block levels to calculate MSEs for
- **MSE_MAP**  
  Matrix of MSE values

**Value**

All MSE values

**Examples**

```r
SG <- CGGPcreate(d=3, batchsize=100)
y <- apply(SG$design, 1, function(x){x[1]+x[2]^2})
SG <- CGGPfit(SG, Y=y)
MSE_MAP <- outer(1:SG$d, 1:8,
  Vectorize(function(dimlcv, lcv1) {
    CGGP_internal_calcMSE(SG$xb[1:SG$sizest[dimlcv]],
    theta=SG$thetaMAP[(dimlcv-1)*SG$numpara+1:SG$numpara],
    CorrMat=SG$CorrMat)
  })),
CGGP_internal_calcMSEde(SG$po[1:SG$poCOUNT, ], MSE_MAP)
```

**CGGP_internal_calcpw**  
*Calculate predictive weights for CGGP*

**Description**

Predictive weights are \(\Sigma^{-1}y\) in standard GP. This calculation is much faster since we don’t need to solve the full system of equations.

**Usage**

`CGGP_internal_calcpw(CGGP, y, theta, return_lS = FALSE)`
Arguments

CGGP  CGGP object
y     Measured values for CGGP$design
theta  Correlation parameters
return_Ls  Should Ls be returned?

Value

Vector with predictive weights

Examples

cggp <- CGGPcreate(d=3, batchsize=100)
y <- apply(cggp$design, 1, function(x){x[1]+x[2]^2+rnorm(1,0,.01}))
CGGP_internal_calcpw(CGGP=cggp, y=y, theta=cggp$thetaMAP)

CGGP_internal_calcpwanddpw

Calculate derivative of pw

Description

Calculate derivative of pw

Usage

CGGP_internal_calcpwanddpw(CGGP, y, theta, return_Ls = FALSE)

Arguments

CGGP  CGGP object
y     Measured values for CGGP$design
theta  Correlation parameters
return_Ls  Should Ls and dLs be returned?

Value

derivative matrix of pw with respect to logtheta

Examples

cggp <- CGGPcreate(d=3, batchsize=100)
y <- apply(cggp$design, 1, function(x){x[1]+x[2]^2+rnorm(1,0,.01)})
CGGP_internal_calcpwanddpw(CGGP=cggp, y=y, theta=cggp$thetaMAP)
CGGP\_internal\_CorrMatCauchy

Cauchy correlation function

Description

Calculate correlation matrix for two sets of points in one dimension. Note that this is not the correlation between two vectors.

Usage

```r
CGGP\_internal\_CorrMatCauchy(
  x1,
  x2,
  theta,
  return\_dCdtheta = FALSE,
  return\_numpara = FALSE,
  return\_logs = FALSE
)
```

Arguments

- `x1`: Vector of coordinates from same dimension
- `x2`: Vector of coordinates from same dimension
- `theta`: Correlation parameters:
  - LS Log of parameter that controls lengthscale
  - FD Logit of 0.5*parameter that controls the fractal dimension
  - HE Log of parameter that controls the hurst effect
- `return\_dCdtheta`: Should dCdtheta be returned?
- `return\_numpara`: Should it just return the number of parameters?
- `return\_logs`: Should log of correlation be returned?

Value

Matrix of correlation values between x1 and x2

See Also

Other correlation functions: `CGGP\_internal\_CorrMatCauchySQT()`, `CGGP\_internal\_CorrMatCauchySQ()`, `CGGP\_internal\_CorrMatGaussian()`, `CGGP\_internal\_CorrMatMatern32()`, `CGGP\_internal\_CorrMatMatern52()`, `CGGP\_internal\_CorrMatPowerExp()`, `CGGP\_internal\_CorrMatWendland0()`, `CGGP\_internal\_CorrMatWendland1()`, `CGGP\_internal\_CorrMatWendland2()`

Examples

```r
CGGP\_internal\_CorrMatCauchy(c(0,.2,.4),c(.1,.3,.5), theta=c(-1,.9,.1))
```
CGGP_internal_CorrMatCauchySQ

CauchySQ correlation function

Description

Calculate correlation matrix for two sets of points in one dimension. Note that this is not the correlation between two vectors.

Usage

```r
CGGP_internal_CorrMatCauchySQ(
  x1,
  x2,
  theta,
  return_dCdtheta = FALSE,
  return_numpara = FALSE,
  returnlogs = FALSE
)
```

Arguments

- **x1**: Vector of coordinates from same dimension
- **x2**: Vector of coordinates from same dimension
- **theta**: Correlation parameters:
  - LS Log of parameter that controls lengthscale
  - FD Log of 0.5*parameter that controls the fractal dimension
  - HE Log of parameter that controls the hurst effect
- **return_dCdtheta**: Should dCdtheta be returned?
- **return_numpara**: Should it just return the number of parameters?
- **returnlogs**: Should log of correlation be returned?

Value

Matrix of correlation values between x1 and x2

See Also

Other correlation functions: `CGGP_internal_CorrMatCauchySQ()`, `CGGP_internal_CorrMatCauchy()`, `CGGP_internal_CorrMatGaussian()`, `CGGP_internal_CorrMatMatern32()`, `CGGP_internal_CorrMatMatern52()`, `CGGP_internal_CorrMatPowerExp()`, `CGGP_internal_CorrMatWendland0()`, `CGGP_internal_CorrMatWendland1()`, `CGGP_internal_CorrMatWendland2()`

Examples

```r
CGGP_internal_CorrMatCauchySQ(c(0,.2,.4),c(.1,.3,.5), theta=c(-.7,-.5))
```
**CGGP_internal_CorrMatCauchySQT**

*CauchySQT correlation function*

---

**Description**

Calculate correlation matrix for two sets of points in one dimension. Note that this is not the correlation between two vectors.

**Usage**

```r
CGGP_internal_CorrMatCauchySQT(
  x1,  
  x2,  
  theta,  
  return_dCdtheta = FALSE,  
  return_numpara = FALSE,  
  returnlogs = FALSE  
)
```

**Arguments**

- **x1** Vector of coordinates from same dimension
- **x2** Vector of coordinates from same dimension
- **theta** Correlation parameters:
  - LS Log of parameter that controls lengthscale
  - FD Logit of 0.5*parameter that controls the fractal dimension
  - HE Log of parameter that controls the hurst effect
- **return_dCdtheta** Should dCdtheta be returned?
- **return_numpara** Should it just return the number of parameters?
- **returnlogs** Should log of correlation be returned?

**Value**

Matrix of correlation values between x1 and x2

**See Also**

Other correlation functions: `CGGP_internal_CorrMatCauchySQ()`, `CGGP_internal_CorrMatCauchy()`, `CGGP_internal_CorrMatGaussian()`, `CGGP_internal_CorrMatMatern32()`, `CGGP_internal_CorrMatMatern52()`, `CGGP_internal_CorrMatPowerExp()`, `CGGP_internal_CorrMatWendland0()`, `CGGP_internal_CorrMatWendland1()`, `CGGP_internal_CorrMatWendland2()`

**Examples**

```r
CGGP_internal_CorrMatCauchySQT(c(0,.2,.4),c(.1,.3,.5), theta=c(-.1,.3,-.7))
```
CGGP_internal_CorrMatGaussian

Gaussian correlation function

Description

Calculate correlation matrix for two sets of points in one dimension. Note that this is not the correlation between two vectors.

Usage

CGGP_internal_CorrMatGaussian(
  x1,
  x2,
  theta,
  return_dCdtheta = FALSE,
  return_numpara = FALSE,
  returnlogs = FALSE
)

Arguments

x1 Vector of coordinates from same dimension
x2 Vector of coordinates from same dimension
theta Correlation parameters:
  • LS Log of parameter that controls lengthscale
  • FD Logit of 0.5*parameter that controls the fractal dimension
  • HE Log of parameter that controls the hurst effect
return_dCdtheta Should dCdtheta be returned?
return_numpara Should it just return the number of parameters?
returnlogs Should log of correlation be returned?

Details

WE HIGHLY ADVISE NOT USING THIS CORRELATION FUNCTION. Try Power Exponential, CauchySQT, Cauchy, or Matern 3/2 instead.

Value

Matrix of correlation values between x1 and x2
CGGP_internal_CorrMatMatern32

See Also
Other correlation functions: CGGP_internal_CorrMatCauchySQT(), CGGP_internal_CorrMatCauchySQ(), CGGP_internal_CorrMatCauchy(), CGGP_internal_CorrMatMatern32(), CGGP_internal_CorrMatMatern52(), CGGP_internal_CorrMatPowerExp(), CGGP_internal_CorrMatWendland0(), CGGP_internal_CorrMatWendland1(), CGGP_internal_CorrMatWendland2()

Examples
CGGP_internal_CorrMatGaussian(c(0, .2, .4), c(.1, .3, .5), theta=c(-.7))

CGGP_internal_CorrMatMatern32

Matern 3/2 correlation function

Description
Calculate correlation matrix for two sets of points in one dimension. Note that this is not the correlation between two vectors.

Usage
CGGP_internal_CorrMatMatern32(
  x1,
  x2,
  theta,
  return_dCdtheta = FALSE,
  return_numpara = FALSE,
  returnlogs = FALSE
)

Arguments
x1 Vector of coordinates from same dimension
x2 Vector of coordinates from same dimension
theta Correlation parameters:
  • LS Log of parameter that controls lengthscale
  • FD Logit of 0.5*parameter that controls the fractal dimension
  • HE Log of parameter that controls the hurst effect
return_dCdtheta Should dCdtheta be returned?
return_numpara Should it just return the number of parameters?
returnlogs Should log of correlation be returned?

Value
Matrix of correlation values between x1 and x2
CGGP_internal_CorrMatMatern52

See Also

Other correlation functions: CGGP_internal_CorrMatCauchySQT(), CGGP_internal_CorrMatCauchySQ(), CGGP_internal_CorrMatCauchy(), CGGP_internal_CorrMatGaussian(), CGGP_internal_CorrMatMatern52(), CGGP_internal_CorrMatPowerExp(), CGGP_internal_CorrMatWendland0(), CGGP_internal_CorrMatWendland1(), CGGP_internal_CorrMatWendland2()

Examples

CGGP_internal_CorrMatMatern32(c(0,.2,.4),c(.1,.3,.5), theta=c(-.7))

CGGP_internal_CorrMatMatern52

Matern 5/2 correlation function

Description

Calculate correlation matrix for two sets of points in one dimension. Note that this is not the correlation between two vectors.

Usage

CGGP_internal_CorrMatMatern52(
  x1,
  x2,
  theta,
  return_dCdtheta = FALSE,
  return_numpara = FALSE,
  returnlogs = FALSE
)

Arguments

x1 Vector of coordinates from same dimension
x2 Vector of coordinates from same dimension
theta Correlation parameters:
  • LS Log of parameter that controls lengthscale
  • FD Logit of 0.5*parameter that controls the fractal dimension
  • HE Log of parameter that controls the hurst effect
return_dCdtheta Should dCdtheta be returned?
return_numpara Should it just return the number of parameters?
returnlogs Should log of correlation be returned?

Value

Matrix of correlation values between x1 and x2
See Also

Other correlation functions: CGGP_internal_CorrMatCauchySQT(), CGGP_internal_CorrMatCauchySQ(), CGGP_internal_CorrMatCauchy(), CGGP_internal_CorrMatGaussian(), CGGP_internal_CorrMatMatern32(), CGGP_internal_CorrMatPowerExp(), CGGP_internal_CorrMatWendland0(), CGGP_internal_CorrMatWendland1(), CGGP_internal_CorrMatWendland2()

Examples

CGGP_internal_CorrMatMatern52(c(0,.2,.4),c(.1,.3,.5), theta=c(-.7))

CGGP_internal_CorrMatPowerExp

*Power exponential correlation function*

Description

Calculate correlation matrix for two sets of points in one dimension. Note that this is not the correlation between two vectors.

Usage

CGGP_internal_CorrMatPowerExp(
  x1,
  x2,
  theta,
  return_dCdtheta = FALSE,
  return_numpara = FALSE,
  returnlogs = FALSE
)

Arguments

x1 Vector of coordinates from same dimension
x2 Vector of coordinates from same dimension
theta Correlation parameters:
  • LS Log of parameter that controls lengthscale
  • FD Logit of 0.5*parameter that controls the fractal dimension
  • HE Log of parameter that controls the hurst effect
return_dCdtheta Should dCdtheta be returned?
return_numpara Should it just return the number of parameters?
returnlogs Should log of correlation be returned?

Value

Matrix of correlation values between x1 and x2
CGGP_internal_CorrMatWendland0

See Also

Other correlation functions: CGGP_internal_CorrMatCauchySQT(), CGGP_internal_CorrMatCauchySQ(), CGGP_internal_CorrMatCauchy(), CGGP_internal_CorrMatGaussian(), CGGP_internal_CorrMatMatern32(), CGGP_internal_CorrMatMatern52(), CGGP_internal_CorrMatWendland0(), CGGP_internal_CorrMatWendland1(), CGGP_internal_CorrMatWendland2()

Examples

CGGP_internal_CorrMatPowerExp(c(0,.2,.4),c(.1,.3,.5), theta=c(-.7,.2))

CGGP_internal_CorrMatWendland0

Wendland0 (Triangle) correlation function

Description

Calculate correlation matrix for two sets of points in one dimension. Note that this is not the correlation between two vectors.

Usage

CGGP_internal_CorrMatWendland0(
  x1,
  x2,
  theta,
  return_dCdtheta = FALSE,
  return_numpara = FALSE,
  returnlogs = FALSE
)

Arguments

x1  Vector of coordinates from same dimension
x2  Vector of coordinates from same dimension
theta  Correlation parameters:
• LS Log of parameter that controls lengthscale
• FD Logit of 0.5*parameter that controls the fractal dimension
• HE Log of parameter that controls the hurst effect
return_dCdtheta  Should dCdtheta be returned?
return_numpara  Should it just return the number of parameters?
returnlogs  Should log of correlation be returned?

Value

Matrix of correlation values between x1 and x2
See Also

Other correlation functions: `CGGP_internal_CorrMatCauchySQ()`, `CGGP_internal_CorrMatCauchySQ()`, `CGGP_internal_CorrMatCauchy()`, `CGGP_internal_CorrMatGaussian()`, `CGGP_internal_CorrMatMatern32()`, `CGGP_internal_CorrMatMatern52()`, `CGGP_internal_CorrMatPowerExp()`, `CGGP_internal_CorrMatWendland1()`, `CGGP_internal_CorrMatWendland2()`

Examples

```r
CGGP_internal_CorrMatWendland1(c(0,.2,.4),c(.1,.3,.5), theta=-.7)
```

---

**Description**

Calculate correlation matrix for two sets of points in one dimension. Note that this is not the correlation between two vectors.

**Usage**

```r
CGGP_internal_CorrMatWendland1(
  x1,
  x2,
  theta,
  return_dCdtheta = FALSE,
  return_numpara = FALSE,
  returnlogs = FALSE
)
```

**Arguments**

- `x1`: Vector of coordinates from same dimension
- `x2`: Vector of coordinates from same dimension
- `theta`: Correlation parameters:
  - LS Log of parameter that controls lengthscale
  - FD Logit of 0.5*parameter that controls the fractal dimension
  - HE Log of parameter that controls the hurst effect
- `return_dCdtheta`: Should dCdtheta be returned?
- `return_numpara`: Should it just return the number of parameters?
- `returnlogs`: Should log of correlation be returned?

**Value**

Matrix of correlation values between x1 and x2
See Also

Other correlation functions: `CGGP_internal_CorrMatCauchySQT()`, `CGGP_internal_CorrMatCauchySQ()`,
`CGGP_internal_CorrMatCauchy()`, `CGGP_internal_CorrMatGaussian()`, `CGGP_internal_CorrMatMatern32()`,
`CGGP_internal_CorrMatMatern52()`, `CGGP_internal_CorrMatPowerExp()`, `CGGP_internal_CorrMatWendland0()`,
`CGGP_internal_CorrMatWendland2()`.

Examples

```r
CGGP_internal_CorrMatWendland1(c(0,.2,.4),c(.1,.3,.5), theta=-.7)
```

**Description**

Calculate correlation matrix for two sets of points in one dimension. Note that this is not the correlation between two vectors.

**Usage**

```r
CGGP_internal_CorrMatWendland2(
  x1, x2,
  theta,
  return_dCdtheta = FALSE,
  return_numpara = FALSE,
  returnlogs = FALSE
)
```

**Arguments**

- `x1`: Vector of coordinates from same dimension
- `x2`: Vector of coordinates from same dimension
- `theta`: Correlation parameters:
  - LS Log of parameter that controls lengthscale
  - FD Log of 0.5*parameter that controls the fractal dimension
  - HE Log of parameter that controls the hurst effect
- `return_dCdtheta`: Should dCdtheta be returned?
- `return_numpara`: Should it just return the number of parameters?
- `returnlogs`: Should log of correlation be returned?

**Value**

Matrix of correlation values between x1 and x2
CGGP\_internal\_gneglogpost

See Also

Other correlation functions: CGGP\_internal\_CorrMatCauchySQT(), CGGP\_internal\_CorrMatCauchySQ(), CGGP\_internal\_CorrMatCauchy(), CGGP\_internal\_CorrMatGaussian(), CGGP\_internal\_CorrMatMatern32(), CGGP\_internal\_CorrMatMatern52(), CGGP\_internal\_CorrMatPowerExp(), CGGP\_internal\_CorrMatWendland0(), CGGP\_internal\_CorrMatWendland1()

Examples

CGGP\_internal\_CorrMatWendland2(c(0,.2,.4),c(.1,.3,.5), theta=-.7)

\[ CGGP\_internal\_gneglogpost \]

*Gradient of negative log likelihood posterior*

Description

Gradient of negative log likelihood posterior

Usage

CGGP\_internal\_gneglogpost(
  theta,
  CGGP,
  y,
  ...,  
  return\_lik = FALSE,
  ys = NULL,
  Xs = NULL,
  HandlingSuppData = "Correct"
)

Arguments

\begin{itemize}
  \item \textbf{theta} \hspace{1cm} Log of correlation parameters
  \item \textbf{CGGP} \hspace{1cm} CGGP object
  \item \textbf{y} \hspace{1cm} CGGP\$design measured values
  \item \textbf{...} \hspace{1cm} Forces you to name remaining arguments
  \item \textbf{return\_lik} \hspace{1cm} If yes, it returns a list with lik and glik
  \item \textbf{ys} \hspace{1cm} Supplementary output data
  \item \textbf{Xs} \hspace{1cm} Supplementary input data
  \item \textbf{HandlingSuppData} \hspace{1cm} How should supplementary data be handled? * Correct: full likelihood with grid and supplemental data * Only: only use supplemental data * Ignore: ignore supplemental data
\end{itemize}
Value

Vector for gradient of likelihood w.r.t. x (theta)

Examples

cg <- CGGPcreate(d=3, batchsize=20)
Y <- apply(cg$design, 1, function(x){x[1]+x[2]^2})
cg <- CGGPfit(cg, Y)
CGGP_internal_gneglogpost(cg$thetaMAP, CGGP=cg, y=cg$y)

CGGP_internal_MSEpredcalc

Calculate MSE prediction along a single dimension

Description

Calculate MSE prediction along a single dimension

Usage

CGGP_internal_MSEpredcalc(xp, xl, theta, CorrMat)

Arguments

xp Points at which to calculate MSE
xl Levels along dimension, vector??
theta Correlation parameters
CorrMat Function that gives correlation matrix for vectors of 1D points.

Value

MSE predictions

Examples

CGGP_internal_MSEpredcalc(c(.4,.52), c(0,.25,.5,.75,1), theta=c(.1,.2), 
CorrMat=CGGP_internal_CorrMatCauchySQ)
Calculate negative log posterior

Usage

CGGP_internal_neglogpost(
  theta,
  CGGP,
  y,
  ...,
  ys = NULL,
  Xs = NULL,
  HandlingSuppData = "Correct"
)

Arguments

theta  Correlation parameters
CGGP   CGGP object
y      Measured values of CGGP$design
...    Forces you to name remaining arguments
ys     Supplementary output data
Xs     Supplementary input data
HandlingSuppData

How should supplementary data be handled?
  • Correct: full likelihood with grid and supplemental data
  • Only: only use supplemental data
  • Ignore: ignore supplemental data

Value

Likelihood

Examples

```r
  cg <- CGGPcreate(d=3, batchsize=20)
  Y <- apply(cg$design, 1, function(x)(x[1]+x[2]^2))
  cg <- CGGPfit(cg, Y)
  CGGP_internal_neglogpost(cg$thetaMAP, CGGP=cg, y=cg$Y)
```
CGGP_internal_set_corr

Set correlation function of CGGP object

Description

Set correlation function of CGGP object

Usage

CGGP_internal_set_corr(CGGP, corr)

Arguments

CGGP  CGGP object
corr   Correlation function

Value

CGGP object

Examples

obj <- CGGPcreate(3, 20, corr="matern52")
CGGP_internal_set_corr(obj, "gaussian")

plot.CGGP

S3 plot method for CGGP

Description


Usage

## S3 method for class 'CGGP'
plot(x, y, ...)

Arguments

x         CGGP object
y         Don’t use
...       Passed to CGGPplotblocks
predict.CGGP

Value

Either makes plot or returns plot object

Examples

SG = CGGPcreate(3,100)
plot(SG)

---

predict.CGGP  \hspace{1cm} S3 predict method for CGGP

Description

Passes to CGGPpred

Predict using SG with y values at xp? Shouldn’t y values already be stored in SG?

Usage

## S3 method for class 'CGGP'
predict(object, xp, ...)

CGGPpred(CGGP, xp, theta = NULL, outdims = NULL)

Arguments

- **object**: CGGP object
- **xp**: x value to predict at
- **...**: Other arguments passed to ‘CGGPpred’
- **CGGP**: SG object
- **theta**: Leave as NULL unless you want to use a value other than thetaMAP. Much slower.
- **outdims**: If multiple outputs fit without PCA and with separate parameters, you can predict just for certain dimensions to speed it up. Will leave other columns in the output, but they will be wrong.

Value

Predicted mean values

See Also

Other CGGP core functions: CGGPappend(), CGGPcreate(), CGGPfit()
Examples

SG <- CGGPcreate(d=3, batchsize=100)
y <- apply(SG$design, 1, function(x){x[1]+x[2]^2+rnorm(1,0,.01)})
SG <- CGGPfit(SG, Y=y)
CGGPpred(SG, matrix(c(.1,.1,.1),1,3))
cbind(CGGPpred(SG, SG$design)$mean, y) # Should be near equal

print.CGGP

Print CGGP object

Description

Default print as a list is bad since there’s a lot of elements.

Usage

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

Arguments

x          CGGP object
...
... Passed to print

Value

String to be printed

Examples

SG = CGGPcreate(3,21)
print(SG)
f <- function(x) {x[1]+exp(x[2]) + log(x[3]+4)}
y <- apply(SG$design, 1, f)
SG <- CGGPfit(SG, y)
print(SG)
rcpp_fastmatclcr

rcpp_fastmatclcr

Description

rcpp_fastmatclcr

Usage

rcpp_fastmatclcr(I, w, MSEmat, S, maxlevel)

Arguments

- **I**: Matrix
- **w**: vector
- **MSEmat**: Matrix
- **S**: Vector
- **maxlevel**: Integer

Value

Nothing, void

rcpp_fastmatclcranddclcr

rcpp_fastmatclcranddclcr

Description

rcpp_fastmatclcranddclcr

Usage

rcpp_fastmatclcranddclcr(I, w, MSEmat, dMSEmat, S, dS, maxlevel, numpara)

Arguments

- **I**: Matrix
- **w**: vector
- **MSEmat**: Matrix
- **dMSEmat**: Matrix
- **S**: Vector
- **dS**: Matrix
- **maxlevel**: Integer
- **numpara**: Integer
Value

Nothing, void

Description

rcpp_kronDBS

Usage

rcpp_gkronDBS(A, dA, B, p)

Arguments

A  Vector
dA Vector
B  Vector
p  Vector

Value

kronDBS calculation

Examples

rcpp_gkronDBS(c(1, 1), c(0, 0), c(.75), c(1, 1))

Description

rcpp_kronDBS

Usage

rcpp_kronDBS(A, B, p)

Arguments

A  Vector
B  Vector
p  Vector
Value

kronDBS calculation

---

valplot  

Plot validation prediction errors

Description

Plot validation prediction errors

Usage

valplot(predmean, predvar, Yval, d = NULL)

Arguments

- **predmean**: Predicted mean
- **predvar**: Predicted variance
- **Yval**: Y validation data
- **d**: If output is multivariate, which column to use. Will do all if left as NULL.

Value

None, makes a plot

Examples

```r
x <- matrix(runif(100*3), ncol=3)
f1 <- function(x){x[1]+x[2]^2}
y <- apply(x, 1, f1)
# Create a linear model on the data
mod <- lm(y ~ ., data.frame(x))
# Predict at validation data
Xval <- matrix(runif(3*100), ncol=3)
mod.pred <- predict.lm(mod, data.frame(Xval), se.fit=TRUE)
# Compare to true results
Yval <- apply(Xval, 1, f1)
valplot(mod.pred$fit, mod.pred$se.fit^2, Yval=Yval)
```
**Description**

Calculate stats for prediction on validation data

**Usage**

```r
valstats(
predmean,
predvar,
Yval,
bydim = TRUE,
RMSE = TRUE,
score = TRUE,
CRPscore = TRUE,
coverage = TRUE,
corr = TRUE,
R2 = TRUE,
MAE = FALSE,
MIS90 = FALSE,
metrics,
min_var = .Machine$double.eps
)
```

**Arguments**

- `predmean`: Predicted mean
- `predvar`: Predicted variance
- `Yval`: Y validation data
- `bydim`: If multiple outputs, should it be done separately by dimension?
- `RMSE`: Should root mean squared error (RMSE) be included?
- `score`: Should score be included?
- `CRPscore`: Should CRP score be included?
- `coverage`: Should coverage be included?
- `corr`: Should correlation between predicted and true mean be included?
- `R2`: Should R^2 be included?
- `MAE`: Should mean absolute error (MAE) be included?
- `MIS90`: Should mean interval score for 90% confidence be included? See Gneiting and Raftery (2007).
- `metrics`: Optional additional metrics to be calculated. Should have same first three parameters as this function.
- `min_var`: Minimum value of the predicted variance. Negative or zero variances can cause errors.


Valstats

Value
data frame

References


Examples

valstats(c(0,1,2), c(.01,.01,.01), c(0,1.1,1.9))
valstats(cbind(c(0,1,2), c(1,2,3)),
    cbind(c(.01,.01,.01),c(.1,.1,.1)),
    cbind(c(0,1.1,1.9),c(1,2,3)))
valstats(cbind(c(0,1,2), c(8,12,34)),
    cbind(c(.01,.01,.01),c(1.1,.81,1.1)),
    cbind(c(0,1.1,1.9),c(10,20,30)), bydim=FALSE)
valstats(cbind(c(.8,1.2,3.4), c(8,12,34)),
    cbind(c(.01,.01,.01),c(1.1,.81,1.1)),
    cbind(c(1,2,3),c(10,20,30)), bydim=FALSE)
Index

* CGGP core functions
  CGGPappend, 3
  CGGPcreate, 4
  CGGPfit, 5
  predict.CGGP, 33
* CGGP plot functions
  CGGPplotblocks, 6
  CGGPplotcorr, 8
  CGGPplotheat, 9
  CGGPplothist, 10
  CGGPplotsamplesneglogpost, 11
  CGGPplotslice, 11
  CGGPplottheta, 13
  CGGPplotvariogram, 13
  CGGPvalplot, 14
* correlation functions
  CGGP_internal_CorrMatCauchy, 19
  CGGP_internal_CorrMatCauchySQ, 20
  CGGP_internal_CorrMatCauchySQT, 21
  CGGP_internal_CorrMatGaussian, 22
  CGGP_internal_CorrMatMatern32, 23
  CGGP_internal_CorrMatMatern52, 24
  CGGP_internal_CorrMatPowerExp, 25
  CGGP_internal_CorrMatWendland0, 26
  CGGP_internal_CorrMatWendland1, 27
  CGGP_internal_CorrMatWendland2, 28
  CGGP_internal_CorrMatMatern32, 19–21, 23, 25–29
  CGGP_internal_CorrMatMatern52, 19–21, 23, 24, 26–29
  CGGP_internal_CorrMatPowerExp, 19–21, 23–25, 25, 27–29
  CGGP_internal_CorrMatWendland0, 19–21, 23–26, 26, 28, 29
  CGGP_internal_CorrMatWendland1, 19–21, 23–27, 27, 29
  CGGP_internal_CorrMatWendland2, 19–21, 23–28, 28
  CGGP_internal_gneglogpost, 29
  CGGP_internal_MSEpredcalc, 30
  CGGP_internal_neglogpost, 31
  CGGP_internal_set_corr, 32
  CGGPappend, 3, 5, 6, 33
  CGGPcreate, 3, 4, 6, 33
  CGGPfit, 3, 5, 6, 33
  CGGPplotblocks, 6, 8–15
  CGGPplotblockselection, 7
  CGGPplotcorr, 6, 8, 9–15
  CGGPplotheat, 6, 8, 9, 10–15
  CGGPplothist, 6, 8, 9, 10, 11–15
  CGGPplotsamplesneglogpost, 6, 8–10, 11, 12–15
  CGGPplotslice, 6, 8–11, 11, 13–15
  CGGPplottheta, 6, 8–12, 13, 14, 15
  CGGPplotvariogram, 6, 8–13, 13, 15
  CGGPpred (predict.CGGP), 33
  CGGPvalplot, 6, 8–14, 14
  CGGPvalstats, 15
  plot.CGGP, 32
  predict.CGGP, 3, 5, 6, 33
  print.CGGP, 34
  rcpp_fastmatclcr, 35
  rcpp_fastmatclcranddclcr, 35
  rcpp_gkronDBS, 36
rcpp_kronDBS, 36
valplot, 37
valstats, 38