Package ‘CGGP’

June 15, 2019

Type Package

Title Composite Grid Gaussian Processes

Version 1.0.1

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, knitr

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

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

Encoding UTF-8

RoxygenNote 6.1.1

VignetteBuilder knitr

NeedsCompilation yes

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

Maintainer Collin Erickson <collinberickson@gmail.com>

Repository CRAN

Date/Publication 2019-06-14 22:55:02 UTC
R topics documented:

CGGP ................................................................. 3
CGGPappend .......................................................... 3
CGGPcreate ............................................................ 4
CGGPfit ................................................................. 5
CGGPplotblocks ......................................................... 6
CGGPplotblockselection ............................................... 7
CGGPplotcorr ........................................................... 7
CGGPplotheath .......................................................... 8
CGGPplothist ........................................................... 9
CGGPplotsamplesneglogpost ........................................... 10
CGGPplotslice ........................................................ 11
CGGPplottheta ......................................................... 12
CGGPplotvariogram .................................................. 13
CGGPvalplot .......................................................... 13
CGGPvalstats .......................................................... 14
CGGP_internal_calcMSE .............................................. 15
CGGP_internal_calcMSEde ............................................. 16
CGGP_internal_calcpw ................................................ 16
CGGP_internal_calcpwanddpw ....................................... 17
CGGP_internal_CorrMatCauchy .................................... 18
CGGP_internal_CorrMatCauchySQ .................................. 19
CGGP_internal_CorrMatCauchySQT ................................ 20
CGGP_internal_CorrMatGaussian .................................. 21
CGGP_internal_CorrMatMatern32 .................................. 22
CGGP_internal_CorrMatMatern52 .................................. 23
CGGP_internal_CorrMatPowerExp .................................. 24
CGGP_internal_neglogpost ......................................... 25
CGGP_internal_MSEpredcalc ....................................... 26
CGGP_internal_neglogpost ......................................... 26
CGGP_internal_set_corr ............................................. 27
plot.CGGP ............................................................ 28
predict.CGGP ........................................................ 28
print.CGGP ........................................................... 29
rcpp_fastmatclcr .................................................... 30
rcpp_fastmatclcranddclcr ......................................... 31
rcpp_gkronDBS ...................................................... 31
rcpp_kronDBS ........................................................ 32
valplot ............................................................... 32
valstats .............................................................. 33
**CGGP: A package for running sparse grid computer experiments**

**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 <- CGGPPappend(CGGP=SG, batchsize=20, selectionmethod="MAP")
```

---

**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

**See Also**

Other CGGP core functions: CGGPPappend, CGGPfit, predict.CGGP

**Examples**

```r
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’

Value

Updated CGGP object fit to data given

See Also

Other CGGP core functions: CGGPappend, CGGPcreate, predict.CGGP

Examples

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

CGGP block plot

Description

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

Usage

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

# 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]*x3) + 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(1,1,1,2,2,1,2,2,1,3), 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 selection changes 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)
f <- function(x){x[1]*1.2+x[3]^4*sin(2*pi*x[2]^2*x[3]) + .1*exp(3*x[3])}
y <- apply(gs$design, 1, f)
gs <- CGGPfit(gs, Y=y)
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**

`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

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

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

---

Heatmap of SG design depth

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

Examples

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

# The first and fourth dimensions are most active and will have greater depth
SG <- CGGPcreate(d=5, batchsize=50)
f <- function(x) {cos(2*pi*x[1]*3) + exp(4*x[4])}
for (i in 1:1) {
  SG <- CGGPfit(SG, Y=apply(SG$design, 1, f))
  SG <- CGGappend(CGGP=SG, batchsize=200)
}
# SG <- CGGPfit(SG, Y=apply(SG$design, 1, f))
CGGPplotheat(SG)

CGGPplohist

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

CGGPplohist(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
SG <- CGGPcreate(d=5, batchsize=10)
SG <- CGGPappend(CGGP=SG, batchsize=100)
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

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

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`, `CGGPplottheta`, `CGGPplotsamplesneglogpost`, `CGGPplotvariogram`, `CGGPvalplot`
**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**

```r
d <- 5
f1 <- function(x)(x[1]*x[2]^2 + cos(x[3]^2*2*pi*4) - 3.3)
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),])
```

gs <- CGGPcreate(d=3, batchsize=100)
f <- function(x){x[1]^1.2 + x[3]^2.4*sin(2*pi*x[2]^2*x[3]) + .1*exp(3*x[3])}
y <- apply(gs$design, 1, f)
gs <- CGGPfit(gs, Y=y)
CGGPplottheta(gs)
```
CGGPlotVariogram

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
CGGPlotVariogram(CGGP, facet = 1, outdims = NULL)

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: CGGPlotBlocks, CGGPlotCorr, CGGPlotHeat, CGGPlotHist, CGGPlotSamplesNegLogPost, CGGPlotSlice, CGGPlotTheta, CGGValPlot

Examples
SG <- CGGPCreate(d=3, batchsize=100)
y <- apply(SG$design, 1, f)
SG <- CGGPFit(SG, Y=y)
CGGPlotVariogram(SG)

CGGValPlot
Plot validation prediction errors for CGGP object

Description
Plot validation prediction errors for CGGP object

Usage
CGGValPlot(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

**See Also**

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

---

**CGGPvalstats**

*Calculate stats for CGGP prediction on validation data*

**Description**

Calculate stats for CGGP prediction on validation data

**Usage**

```r
CGGPvalstats(CGGP, Xval, Yval, bydim = TRUE, ...)
```

**Arguments**

- **CGGP**
  - CGGP object
- **Xval**
  - X validation matrix
- **Yval**
  - Y validation data
- **bydim**
  - If multiple outputs, should it be done separately by dimension?
- **...**
  - Passed to `valstats`, such as which stats to calculate.

**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,.5,.9), theta=c(1,2,3),
                      CorrMat=CGGP_internal_CorrMatCauchySQI)
```
CGGP_internal_calcMSEde

*Calculate MSE over blocks*

**Description**

Delta of adding block is product over i=1..d of IMSE(i,j-1) - 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$design, 1, function(x)(x[1]+x[2]^2))
SG <- cGGPfit(SG, y)
MSE_MAP <- outer(1:SG$d, 1:8,
    Vectorize(function(dimlcv, lcv)
        {cGGP_internal_calcMSEde(SG$x[b1:SG$sizeest[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

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

---

### CGGP_internal_calcpwanddpw

*Calculate derivative of pw*

Description

Calculate derivative of pw

Usage

```r
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

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

CGGP_internal_CorrMatCauchy(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 demension
  • HE Log of parameter that controls the hurst effect
return_dCdtheta Should dC/dtheta 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_CorrMatCauchySQx, CGGP_internal_CorrMatGaussian, CGGP_internal_CorrMatMatern32, CGGP_internal_CorrMatMatern52, CGGP_internal_CorrMatPowerExp

Examples
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

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: 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_CorrMatCauchy, CGGP_internal_CorrMatGaussian, CGGP_internal_CorrMatMatern32, CGGP_internal_CorrMatMatern52, CGGP_internal_CorrMatPowerExp

Examples

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

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 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?

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

See Also

Other correlation functions: CGGP_internal_CorrMatCauchySQT, CGGP_internal_CorrMatCauchySQ, CGGP_internal_CorrMatCauchy, CGGP_internal_CorrMatMatern32, CGGP_internal_CorrMatMatern52, CGGP_internal_CorrMatPowerExp

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

See Also

Other correlation functions: CGGP_internal_CorrMatCauchySQT, CGGP_internal_CorrMatCauchySQ, CGGP_internal_CorrMatCauchy, CGGP_internal_CorrMatGaussian, CGGP_internal_CorrMatMatern52, CGGP_internal_CorrMatPowerExp

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** Log of 0.5*parameter that controls the fractal demension
  - **HE** Log of parameter that controls the hurst effect
- **return\_dCdtheta**  
  Should dC/dtheta 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\_CorrMatCauchy, CGGP\_internal\_CorrMatGaussian, CGGP\_internal\_CorrMatMatern32, CGGP\_internal\_CorrMatPowerExp

### 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

See Also
Other correlation functions: CGGP_internal_CorrMatCauchySQT, CGGP_internal_CorrMatCauchySQ, CGGP_internal_CorrMatCauchy, CGGP_internal_CorrMatGaussian, CGGP_internal_CorrMatMatern32, CGGP_internal_CorrMatMatern52

Examples
CGGP_internal_CorrMatPowerExp(c(0,.2,.4),c(.1,.3,.5), theta=c(-.7,.2))
Gradient of negative log likelihood posterior

Description
Gradient of negative log likelihood posterior

Usage
\texttt{CGGP\_internal\_gneglogpost(theta, CGGP, y, \ldots, return\_lik = FALSE,}
\texttt{\hspace{1cm} ys = NULL, xs = NULL, HandlingSuppData = "Correct")}

Arguments
\begin{itemize}
  \item \texttt{theta} \hspace{3cm} Log of correlation parameters
  \item \texttt{CGGP} \hspace{3cm} CGGP object
  \item \texttt{y} \hspace{3cm} CGGP$design measured values
  \item \texttt{\ldots} \hspace{3cm} Forces you to name remaining arguments
  \item \texttt{return\_lik} \hspace{3cm} If yes, it returns a list with lik and glik
  \item \texttt{ys} \hspace{3cm} Supplementary output data
  \item \texttt{xs} \hspace{3cm} Supplementary input data
  \item \texttt{HandlingSuppData} \hspace{3cm} 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
\begin{verbatim}
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)
\end{verbatim}
CGGP_internal_MSEpredcalc

*Calculate MSE prediction along a single dimension*

**Description**

Calculate MSE prediction along a single dimension

**Usage**

`CGGP_internal_MSEpredcalc(xp, x1, theta, CorrMat)`

**Arguments**

- `xp`: Points at which to calculate MSE
- `x1`: Levels along dimension, vector
- `theta`: Correlation parameters
- `CorrMat`: Function that gives correlation matrix for vectors of 1D points.

**Value**

MSE predictions

**Examples**

```r
CGGP_internal_MSEpredcalc(c(-.4,.52), c(0,.25,.5,.75,1), theta=c(.1,.2),
CorrMat=CGGP_internal_CorrMatCauchySQ)
```

---

CGGP_internal_neglogpost

*Calculate negative log posterior*

**Description**

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

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

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

---

**plot.CGGP**

*S3 plot method for CGGP*

**Description**

There are a few different plot functions for CGGP objects: ‘CGGPblockplot’, ‘CGGPcorrplot’, ‘CGGPprojectionplot’, ‘CGGPvalplot’, ‘CGGPheat’, and ‘CGGPhist’. Currently ‘CGGPblockplot’ is the default plot object.

**Usage**

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

**Arguments**

- `x`: CGGP object
- `y`: Don’t use
- `...`: Passed to CGGPblockplot

**Value**

Either makes plot or returns plot object

**Examples**

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

---

**predict.CGGP**

*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

```r
## 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

```r
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

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

x  CGGP object

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)

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_fastmatclrcranddclcr**

**Description**

rcpp_fastmatclrcranddclcr

**Usage**

```r
ccpp_fastmatclrcranddclcr(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

**rcpp_gkronDBS**

**Description**

rcpp_gkronDBS

**Usage**

```r
ccpp_gkronDBS(A, dA, B, p)
```

**Arguments**

- **A**: Vector
- **dA**: Vector
- **B**: Vector
- **p**: Vector
**Value**

kronDBS calculation

**Examples**

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

---

**Description**

rcpp_kronDBS

**Usage**

```r
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**

```r
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.
valstats

**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$fitted, mod.pred$se.fit^2, Yval=Yval)
```

**valstats** Calculate stats for prediction on validation data

**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.

Value  
data frame

References  

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

CGGP, 3
CGGP-package (CGGP), 3
CGGP_internal_calcMSE, 15
CGGP_internal_calcMSEde, 16
CGGP_internal_calcpw, 16
CGGP_internal_calcpwanddpw, 17
CGGP_internal_CorrMatCauchy, 18, 19–24
CGGP_internal_CorrMatCauchySQ, 18, 19, 20–24
CGGP_internal_CorrMatCauchySQT, 18, 19, 20, 21–24
CGGP_internal_CorrMatGaussian, 18–20, 21, 22–24
CGGP_internal_CorrMatMatern32, 18–21, 22, 23, 24
CGGP_internal_CorrMatMatern52, 18–22, 23, 24
CGGP_internal_CorrMatPowerExp, 18–23, 24
CGGP_internal_gneglogpost, 25
CGGP_internal_MSEpredcalc, 26
CGGP_internal_neglogpost, 26
CGGP_append, 3, 4, 5, 29
CGGP_create, 3, 4, 5, 29
CGGP_fit, 3, 4, 5, 29
CGGP_plotblocks, 6, 8–14
CGGP_plotblockselection, 7
CGGP_plotcorr, 6, 7, 9–14
CGGP_plotheat, 6, 8, 10–14
CGGP_plothist, 6, 8, 9, 10–14
CGGP_plotsamplesneglogpost, 6, 8–10, 10, 11–14
CGGP_plotslice, 6, 8–10, 11, 12–14
CGGP_plottheta, 6, 8–11, 12, 13, 14
CGGP_plotvariogram, 6, 8–12, 13, 14
CGGP_pred (predict.CGGP), 28
CGGP_valplot, 6, 8–13, 13
CGGP_valstats, 14

plot.CGGP, 28
predict.CGGP, 3–5, 28
print.CGGP, 29
rcpp_fastmatclcr, 30
rcpp_fastmatclcranddclcr, 31
rcpp_gkronDBS, 31
rcpp_kronDBS, 32
valplot, 32
valstats, 33