ooDACE toolbox

1.4

Generated by Doxygen 1.8.2

Mon Jun 3 2013 11:49:07

Contents

1 The ooDACE toolbox documentation

1.1 Introduction

The **ooDACE** toolbox is a versatile Matlab toolbox that implements the popular Gaussian Process based kriging surrogate models. Kriging is in particular popular for approximating (and optimizing) deterministic computer experiments. Given a dataset the toolbox automatically fits a kriging surrogate model to it. Afterwards the kriging surrogate can be fully exploited instead of the (probably more expensive) simulation code.

The toolbox is aimed for solving complex applications (expensive simulation codes, physical experiments, ...) and for researching new kriging extensions and techniques.

1.2 Download

See `download page`

1.3 Quick start guide

Note

Before the toolbox can be used you have to include the toolbox in Matlab's search path. You can do this manually by running startup, or, if Matlab is started in the root toolbox directory, then startup will be run automatically.

```
startup
```

Now the toolbox is ready to be used. The ooDACE toolbox is designed in an object oriented (OO) fashion. It is strongly recommended to exploit the OO design directly, i.e., use the Kriging and Optimizer matlab classes. However, for convenience wrapper scripts (dacefit, predictor) are provided that emulate the DACE toolbox interface (see wrapper scripts for more information).

Lets define **n** as the number of observations and **d** as the number of input parameters. Then the n-by-d input sample matrix is denoted by **samples** (each row is one observation) and the corresponding output values are stored in the n-by-1 matrix **values**.

The **ooDACE** toolbox provides a script, `oodacefit.m`, that just takes your dataset (a **samples** and **values** matrix) and returns a fitted kriging object, all other parameters are set to some sensible defaults. For instance,

```
% ordinary kriging
k = ooDACEfit( samples, values );
y = k.predict(x);

% or, regression kriging
opts.lambda0 = 0;
opts.lambdaBounds = [-5 ; 5]; % log scale
k = ooDACEfit( samples, values, opts );
y = k.predict(x);
```

For more flexibility the user can utilize the kriging classes directly. **lb** and **ub** are 1-by-d arrays defining the lower bounds and upper bounds, respectively, needed to optimize the **hyperparameters**. In addition, a set of starting values has to be specified, namely, **hyperparameters0** is also an 1-by-d array. Example code to create a kriging model follows:

```
...
 % Generate kriging options structure
 opts = Kriging.getDefaultOptions();
 opts.hpBounds = [lb ; ub]; % hyperparameter optimization bounds

 % configure the optimization algorithm (only one optimizer is included)
 % the Matlab Optimization toolbox is REQUIRED
 optimopts.GradObj = 'on';
 optimopts.DerivativeCheck = 'off';
 optimopts.Diagnostics = 'off';
 optimopts.Algorithm = 'active-set';
 opts.hpOptimizer = MatlabOptimizer( dim, 1, optimopts );

 % create and fit the kriging model
 k = Kriging( opts, hyperparameters0, 'regpoly0', @corrgauss );
 k = k.fit( samples, values );

 % k represents the approximation and can now be used, e.g.,
 [y mse] = k.predict( [1 2] )
...
```

See the included `demo.m` and `oodacefit.m` scripts for more example code on how to use the **ooDACE** toolbox (including more advanced features such as using blind kriging (`BlindKriging`) or how to use regression instead of interpolation). For more information on the classes and their methods please refer to the doxygen documentation and the source files.

1.4 DACE toolbox interface

The **ooDACE** toolbox provides two scripts `dacefit.m` and `predictor.m` that emulate the behavior of the **DACE** toolbox ([1]). Note, that full compatibility between ooDACE and the DACE toolbox is not provided. The scripts merely aim to ease the transition from the DACE toolbox to the ooDACE toolbox.

Example code:

```
krige = dacefit(samples, values, 'regpoly0', 'corrgauss',
      hyperparameters0, lb, ub )
y = predictor([1 2], krige)
```

Obviously, a lot less code is used to copy the setup described above. However, less code means less flexibility (e.g., `blind` kriging and regression kriging are not available using the wrapper scripts). Hence, it is suggested to learn the object oriented interface of **ooDACE** and use it instead.

1.5 Contribute

Suggestions on how to improve the ooDACE toolbox are always welcome. For more information please see the feedback page.

2 Test List

Member demo (var id)

Test case 1: Fits an ordinary kriging model on the branin function Covers: ordinary kriging interpolation, marginalLikelihood

Test case 2: Fits an ordinary kriging model on the bird function Covers: ordinary kriging regression, re-interpolation of variance, pseudoLikelihood, likelihood debug plot

Test case 3: Fits a blind kriging model on the branin function. Covers: blind kriging

Test case 4: Fits a cokriging model on a mathematical 1D function. Covers: cokriging

Test case 5: Fits a stochastic kriging model on the branin function plus some stochastic noise. Covers: stochastic kriging, sigma2 optimization

3 Todo List

Class BasicGaussianProcess

Refactor correlation functions into proper basis function class hierarchy.

solve the correlation matrix vs covariance matrix issue

Member BasicGaussianProcess::imse ()

Implement generic monte carlo integration

Member BasicGaussianProcess::marginalLikelihood (var dpsi, var dsigma2)

Adjoint derivatives work, but are very slow due to naive implementation

Member BasicGaussianProcess::updateRegression (var F, var hp)

Rho is only used by co-kriging, can we abstract this somehow ?

Class CoKriging

Generalize to an arbitrary number of (multi-fidelity) datasets

Member Kriging::getDefaultOptions ()

Apparantly Matlab 2008b doesn't auto forward the (static) call to BasicGaussianProcess::getDefaultOptions()

4 Hierarchical Index

4.1 Class Hierarchy

This inheritance list is sorted roughly, but not completely, alphabetically:

5 Class Index

5.1 Class List

Here are the classes, structs, unions and interfaces with brief descriptions:

6 File Index

6.1 File List

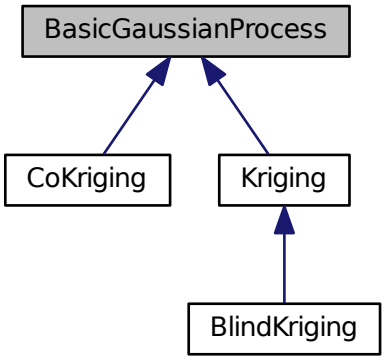
Here is a list of all files with brief descriptions:

7 Class Documentation

7.1 BasicGaussianProcess Class Reference

A kriging surrogate model (also known as a Gaussian Process)

Inheritance diagram for BasicGaussianProcess:



Public Member Functions

- function BasicGaussianProcess (var varargin)
  Class constructor.
- function getHyperparameters ()
  Returns the hyperparameters.
- function getRho ()
  Returns the rho parameter (only valid for `CoKriging`)
- function getProcessVariance ()
  Returns the process variance (sigma2)
- function getCorrelationMatrix ()
  Returns the full extrinsic correlation matrix.
- function getSigma ()
  Returns the intrinsic covariance matrix.
- function getRegressionMatrix ()
  Returns the model matrix (Vandermonde matrix)
- function getSamples ()
  Returns the input sample matrix.
- function getValues ()
  Returns the output value matrix.
- function setOption (var key, var value)
  Sets a value in the options structure.
- function cleanup ()
  Clears some unused variables.
- function display ()
  Returns user-friendly description of the class instance.

- function fit (var samples, var values)

  Fits a gaussian process for a dataset.
- function predict (var points)

  Predict the mean and/or variance for one or more points x.
- function predict_derivatives (var point)

  Predict the derivatives of the mean and/or variance for a points.
- function predict_limit (var points)

  Limit predictor of kriging (EXPERIMENTAL)
- function regressionFunction (var options)

  Returns the regression function.
- function correlationFunction (var options)

  Returns the internal correlation function handle and a symbolic expression of the the correlation part.
- function getExpression (var options)

  Returns the Matlab expression of this Gaussian Process model.
- function cvpe ()

  Calculates the log of the leave-one-out cross validation error (LOO-CV)
- function marginalLikelihood (var dpsi, var dsigma2)

  Marginal likelihood function.
- function pseudoLikelihood (var dpsi, var dsigma2)

  Leave-one-out predictive log probability (pseudo-likelihood)
- function mseTestset (var testx, var testy)

  Calculates error on holdout set.
- function imse ()

  Calculates the log of the integrated mean squared error.
- function rcValues ()

  Quantifies magnification of noise (lower is better)
- function plotVariogram ()

  Variogram plot (EXPERIMENTAL)

Static Public Member Functions

- static function getDefaultOptions ()

  Returns a default options structure.

Public Attributes

- const var RHO

  index of the rho parameter
- const var LAMBDA

  index of the lambda parameter
- const var SIGMA2

  index of the sigma2 parameter
- const var HP

  index of the correlation function parameters

Protected Member Functions

- function setData (var samples, var values)

  Sets samples and values matrix.
- function updateModel (var F, var hp)

  Constructs model.
- function updateRegression (var F, var hp)

  Constructs regression part.
- function updateStochasticProcess (var hp)

  Constructs correlation part.
- function extrinsicCorrelationMatrix (var points1, var points2)

  Constructs extrinsic correlation matrix.
- function intrinsicCovarianceMatrix (var points1, var points2)

  Constructs intrinsic covariance matrix (stochastic kriging/regression kriging)
- function regressionMatrix (var points)

  Constructs regression matrix.
- function tuneParameters (var F)

  Hyperparameter optimization.
- function generateDegrees (var dj)

  Generate degrees matrix from individual ones.

Protected Attributes

- var options
- var regressionFcn

  degrees matrix (strings are converted)
- var correlationFcn

  string -> function handle
- var hyperparameters0

  initial hp values OR the real ones (if optimization is done outside the class)
- var dist

  sample inter-distance
- var distIdxPsi

  indexing needed to calculate psi from D
- var optimIdx

  logical indices to parameters that are optimized
- var optimNrParameters

  number of optimization parameter (vector; one entry per type of parameter)
- var alpha

  Regression coefficients.
- var gamma

  'Correlation part' coefficients
- var hyperparameters

  correlation parameters
- var rho
- var C

  Choleski Decomposition of extrinsic + intrinsic matrices.
- var sigma2

  process variance of the GP (extrinsic variance)
- var tau2

  intrinsic variance

- var Sigma

  intrinsic covariance matrix (amount of regression of stochastic part)
- var Ft

  decorrelated model matrix
- var R

  from QR decomposition of regression part
- var sigma2_reinterp

  Reinterpolation version of `sigma2`.
- var C_reinterp

  Reinterpolation version of `C`.
- var Ft_reinterp

  Reinterpolation version of `Ft`.
- var R_reinterp

  Reinterpolation version of `R`.

7.1.1 Detailed Description

A kriging surrogate model (also known as a Gaussian Process)

Papers:

- "Design and Analysis of Computer Experiments", J. Sacks, W. Welch, T. Mitchell, H. Wynn, 1989
- "Design and Analysis of Simulation Experiments", J.P.C. Kleijnen, Springer, 2008
- "Engineering Design Via Surrogate Modelling: A Practical Guide", A. Forrester and A. Sobester and A. Keane, Wiley, 2008
- "Gaussian Processes for Machine Learning", C. E. Rasmussen and C. K. I. Williams, MIT Press, 2006

Todo Refactor correlation functions into proper basis function class hierarchy.

solve the correlation matrix vs covariance matrix issue

7.1.2 Constructor & Destructor Documentation

7.1.2.1 function BasicGaussianProcess (var *varargin*) [inline]

Class constructor.

Parameters

options	Options structure
hyperparameters0	Initial theta values
regressionFcn	The type of trend function to use ('regpoly0', 'regpoly1', ...)
correlationFcn	Function handle to the correlation function (@corrgauss, ...)

Returns

instance of the basicGaussianProcess class.

7.1.3 Member Function Documentation

7.1.3.1 function cleanup ( ) [inline]

Clears some unused variables.

7.1.3.2 function correlationFunction (var *options*)

Returns the internal correlation function handle and a symbolic expression of the the correlation part.

Example: [correlationFcn expression] = correlationFunction(this,struct('latex', true, 'includeCoefficients', false))

The expression is based on the scaled data

Parameters

options	Options struct

Return values

correlationFcn	String of correlation function
expression	Symbolic expression

7.1.3.3 function cvpe ( )

Calculates the log of the leave-one-out cross validation error (LOO-CV)

Error function used is the Mean Squared Error (MSE)

Return values

out	log(loo-cv) score

Papers:

- "Blind Kriging: A New Method for Developing Metamodels", V.R. Joseph and Y. Hung and A. Sudjianto, ASME Journal of Mechanical Design, 2008
- "Predictive Approaches for Choosing Hyperparameters in Gaussian Process", S. Sundararajan, S.S. Keerthu, 1999

7.1.3.4 function display ( ) [inline]

Returns user-friendly description of the class instance.

7.1.3.5 function extrinsicCorrelationMatrix (var *points1,* var *points2*) [protected]

Constructs extrinsic correlation matrix.

Generate correlation matrix of points1 vs points2 using the current hyperparameters as followed:

Parameters

points1	input point matrix (optional)
points2	input point matrix (optional)

Return values

psi	Correlation matrix
dpsi	Derivative of correlation matrix w.r.t. the hyperparameters

- extrinsicCorrelationMatrix(this)
  - Assume points1 = points2 = samples
- extrinsicCorrelationMatrix(this, points1)
  - Assume points2 = samples

- extrinsicCorrelationMatrix(this, points1, points2)

NOTE: The first case returns derivatives w.r.t. X The latter two cases returns derivatives w.r.t. the hyperparameters

7.1.3.6 function fit (var *samples,* var *values*)

Fits a gaussian process for a dataset.

Need to be invoked before calling any of the prediction methods.

Parameters

samples	input sample matrix
values	output value matrix

Note

Kriging can't do a cleanup automatically, if needed cleanup() can be called manually.

7.1.3.7 function generateDegrees (var *dj*) [protected]

Generate degrees matrix from individual ones.

Generates a (full) degree matrix based on one or more degree matrices for variable j The option 'regressionMaxLevelInteractions' determines the maximum level of interactions (BasicGaussianProcess::getDefaultOptions)

Parameters

dj	Cell array of degree matrices (for each dimension)

Return values

degrees	Full degrees matrix
idx	Cell array of indices (used by blind kriging)

7.1.3.8 function getCorrelationMatrix ( ) [inline]

Returns the full extrinsic correlation matrix.

7.1.3.9 static function getDefaultOptions ( ) [inline], [static]

Returns a default options structure.

Available options:

```
options = struct( ...
    'generateHyperparameters0', false, ...
    'hpBounds', [], ... % hyperparameter bounds
    'hpOptimizer', [], ... % optimizer class
    'hpLikelihood', @marginalLikelihood, ...
    'sigma20', NaN, ... % initial value for sigma2
    'sigma2Bounds', [0.001 ; 10], ... % sigma2 parameter bounds
    'lambda0' ,-Inf, ... % initial lambda values
    'lambdaBounds', [-10 ; 0], ... % lambda parameter bounds (in log scale)
    'Sigma', [], ... % intrinsic covariance matrix (stochastic kriging)
    'reinterpolation', false, ... % reinterpolate error (replaces standard
     error)
    'lowRankApproximation', false, ... % enable low rank approximation of
     correlation matrix
    'rankTol', 1e-12, ... % tolerance for lowRankApprox.
    'rankMax', Inf, ... % maximum rank to achieve for lowRankApprox.
    'regressionMaxLevelInteractions', 2, ... % consider maximal two-level
     interactions
    'debug', false, ... % enables debug plot of the likelihood function
...  % Cokriging specific
    'rho0', -Inf, ... % initial scaling factor between datasets
    'rhoBounds', [1 ; 5] ... % scaling factor bounds
```

```
);
```

Return values

options	Options structure

7.1.3.10 function getExpression (var *options*)

Returns the Matlab expression of this Gaussian Process model.

Example: expression = getExpression(this)

Parameters

options	Options struct

Return values

expression	Symbolic expression

7.1.3.11 function getHyperparameters ( ) [inline]

Returns the hyperparameters.

7.1.3.12 function getProcessVariance ( ) [inline]

Returns the process variance (sigma2)

7.1.3.13 function getRegressionMatrix ( ) [inline]

Returns the model matrix (Vandermonde matrix)

7.1.3.14 function getRho ( ) [inline]

Returns the rho parameter (only valid for CoKriging)

7.1.3.15 function getSamples ( ) [inline]

Returns the input sample matrix.

7.1.3.16 function getSigma ( ) [inline]

Returns the intrinsic covariance matrix.

7.1.3.17 function getValues ( ) [inline]

Returns the output value matrix.

7.1.3.18 function imse ( )

Calculates the log of the integrated mean squared error.

Only supported for 1D and 2D problems.

Return values

out	log(imse) score

Papers:

- See Sacks 1989

- See pechiny paper...

Todo Implement generic monte carlo integration

7.1.3.19 function intrinsicCovarianceMatrix (var *points1*, var *points2*) [protected]

Constructs intrinsic covariance matrix (stochastic kriging/regression kriging)

Generate covariance matrix matrix of points1 vs points2 using the current hyperparameters as followed:

Parameters

points1	input point matrix (optional)
points2	input point matrix (optional)

Return values

psi	Covariance matrix
dpsi	Derivative of covariance matrix w.r.t. the hyperparameters OR the input points

- intrinsicCovarianceMatrix(this)
  - Assume points1 = points2 = samples
- intrinsicCovarianceMatrix(this, points1)
  - Not used (and not implemented yet)
- intrinsicCovarianceMatrix(this, points1, points2)
  - Not used (and not implemented yet)

7.1.3.20 function marginalLikelihood (var *dpsi*, var *dsigma2*)

Marginal likelihood function.

Used for Maximum Likelihood Estimation (MLE)

Parameters

dpsi	cell array of derivative matrices (optional; for internal use only)

Return values

out	score
dout	Derivatives w.r.t. hyperparameters

Papers:

- "Gaussian Processes for Machine Learning" (Chapter 5), C. E. Rasmussen and C. K. I. Williams, MIT Press, 2006
- "An adjoint for likelihood maximization" D.J.J. Toal, A.I.J. Forrester, N.W. Bressloff, A.J. Keane, C. Holden, Proc. of the Royal Society, 2009

Todo Adjoint derivatives work, but are very slow due to naive implementation

7.1.3.21 function mseTestset (var *testx*, var *testy*)

Calculates error on holdout set.

Error function used is the Mean Squared Error (MSE) function.

Parameters

testx	input samples of the test set
testy	output samples of the test set

Return values

out	mse error on test set

7.1.3.22 function plotVariogram ( )

Variogram plot (EXPERIMENTAL)

Plots the experimental (semi-)variogram (based on the data) as well as the theoretical kriging (semi-)variogram (defined by the correlation function).

Return values

h	Figure handle

Empirical variogram:

- Methods of moments estimator Matheron (Cressie, 1993)
- More robust estimators: Cressie-Hawkins (1980), Genton (1998) In general: use median instead of mean

7.1.3.23 function predict (var *points*)

Predict the mean and/or variance for one or more points x.

Parameters

points	Matrix of input points to be predicted

Return values

values	predicted output values
sigma2	predicted variance of the output values (optional)

Precondition

The kriging object should be fitted using a dataset (BasicGaussianProcess::fit)

7.1.3.24 function predict_derivatives (var *point*)

Predict the derivatives of the mean and/or variance for a points.

NOTE:

Parameters

point	input point to calculate the derivative of

Return values

dvalues	Derivative w.r.t. the output
sigma2	Derivative w.r.t. the output variance (optional)

- limited to one point at a time (x is a vector)

Precondition

The kriging object should be fitted using a dataset (BasicGaussianProcess::fit)

7.1.3.25 function predict_limit (var *points*)

Limit predictor of kriging (EXPERIMENTAL)

Parameters

points	Matrix of input points to be predicted

Return values

values	predicted output values
sigma2	predicted variance of the output values (optional)

7.1.3.26 function pseudoLikelihood (var *dpsi,* var *dsigma2*)

Leave-one-out predictive log probability (pseudo-likelihood)

Calculates the leave-one-out predictive log probability.

Parameters

dpsi	cell array of derivative matrices (optional; for internal use only)

Return values

out	score
dout	Derivatives w.r.t. hyperparameters

Papers:

- "Predictive Approaches for Choosing Hyperparameters in Gaussian Process", S. Sundararajan, S.S. Keerthu, 1999

7.1.3.27 function rcValues ( )

Quantifies magnification of noise (lower is better)

Robustness-criterion (In theory useful only for ordinary kriging).

Return values

rc	robustness-criterion

Returns a 2xn matrix:

- the first row contains the absolute robustness
- the second row contains the relative robustness Papers:
- "Kriging models that are robust w.r.t. simulation errors" A.Y.D. Siem, D. den Hertog (tech report)

7.1.3.28 function regressionFunction (var *options*)

Returns the regression function.

Example: [regressionFcn expression terms] = regressionFunction(this, struct('latex', true, 'precision', '%.5g'))

The expression is based on the scaled data

Parameters

options	Options struct

Return values

regressionFcn	Degree matrix representing the regression function
expression	Symbolic expression
terms	Cell array of the individual terms

7.1.3.29 function regressionMatrix (var *points*) [protected]

Constructs regression matrix.

Regression matrix (model matrix, Vandermonde matrix, ...) for a set of points Based on this.regressionFcn.

Parameters

points	input point matrix (optional)

Return values

F	Model matrix
dF	Derivative of model matrix w.r.t. the hyperparameters OR the input points

7.1.3.30 function setData (var *samples,* var *values*) [inline], [protected]

Sets samples and values matrix.

Parameters

samples	input sample matrix
values	output value matrix

7.1.3.31 function setOption (var *key,* var *value*) [inline]

Sets a value in the options structure.

Parameters

key	option name
value	option value

7.1.3.32 function tuneParameters (var *F*) [protected]

Hyperparameter optimization.

Setups and invokes the optimizer.

Parameters

F	model matrix

Return values

optimHp	optimized hyperparameters
perf	Performance score (likelihood score)

7.1.3.33 function updateModel (var *F*, var *hp*) [protected]

Constructs model.

Full update of the model (regression + correlation part)

Parameters

F	model matrix
hp	new hyperparameters

7.1.3.34 function updateRegression (var *F*, var *hp*) [protected]

Constructs regression part.

Updates regression part of the model.

Parameters

F	model matrix
hp	new hyperparameters

Return values

err	error string (if any)

Todo Rho is only used by co-kriging, can we abstract this somehow ?

7.1.3.35 function updateStochasticProcess (var *hp*) [protected]

Constructs correlation part.

Updates correlation part of the model.

Parameters

hp	hyperparameters

Return values

err	error string (if any)
dpsi	Derivative of correlation matrix w.r.t. the hyperparameters

7.1.4 Member Data Documentation

7.1.4.1 var alpha [protected]

Regression coefficients.

7.1.4.2 var C [protected]

Choleski Decomposition of extrinsic + intrinsic matrices.

7.1.4.3 var C_reinterp [protected]

Reinterpolation version of C.

7.1.4.4 var correlationFcn [protected]

string -> function handle

7.1.4.5 var dist [protected]

sample inter-distance

7.1.4.6 var distIdxPsi [protected]

indexing needed to calculate psi from D

7.1.4.7 var Ft [protected]

decorrelated model matrix

7.1.4.8 var Ft_reinterp [protected]

Reinterpolation version of Ft.

7.1.4.9 var gamma [protected]

'Correlation part' coefficients

7.1.4.10 const var HP

index of the correlation function parameters

7.1.4.11 var hyperparameters [protected]

correlation parameters

7.1.4.12 var hyperparameters0 [protected]

initial hp values OR the real ones (if optimization is done outside the class)

7.1.4.13 const var LAMBDA

index of the lambda parameter

7.1.4.14 var optimIdx [protected]

logical indices to parameters that are optimized

7.1.4.15 var optimNrParameters [protected]

number of optimization parameter (vector; one entry per type of parameter)

7.1.4.16 var options [protected]

7.1.4.17 var R [protected]

from QR decomposition of regression part

7.1.4.18 var R_reinterp [protected]

Reinterpolation version of R.

7.1.4.19 var regressionFcn [protected]

degrees matrix (strings are converted)

7.1.4.20 const var RHO

index of the rho parameter

7.1.4.21 var rho [protected]

7.1.4.22 var Sigma [protected]

intrinsic covariance matrix (amount of regression of stochastic part)

7.1.4.23 const var SIGMA2

index of the sigma2 parameter

7.1.4.24 var sigma2 [protected]

process variance of the GP (extrinsic variance)

7.1.4.25 var sigma2_reinterp [protected]

Reinterpolation version of `sigma2`.

Note

reinterpolation: might be nicer to just construct and keep a sub-GP... takes more space, some calculations are done twice but performance shouldn't take a very big hit...

7.1.4.26 var tau2 [protected]

intrinsic variance

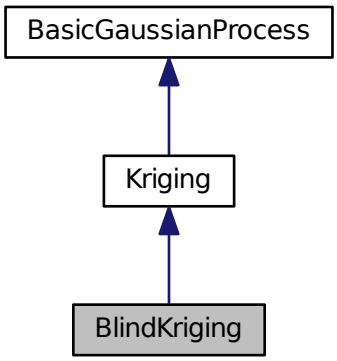
The documentation for this class was generated from the following files:

- /home/ilm/projecten/meta/scripts/ooDACE/@BasicGaussianProcess/BasicGaussianProcess.m
- /home/ilm/projecten/meta/scripts/ooDACE/@BasicGaussianProcess/correlationFunction.m
- /home/ilm/projecten/meta/scripts/ooDACE/@BasicGaussianProcess/cvpe.m
- /home/ilm/projecten/meta/scripts/ooDACE/@BasicGaussianProcess/extrinsicCorrelationMatrix.m
- /home/ilm/projecten/meta/scripts/ooDACE/@BasicGaussianProcess/fit.m
- /home/ilm/projecten/meta/scripts/ooDACE/@BasicGaussianProcess/generateDegrees.m
- /home/ilm/projecten/meta/scripts/ooDACE/@BasicGaussianProcess/getExpression.m
- /home/ilm/projecten/meta/scripts/ooDACE/@BasicGaussianProcess/imse.m
- /home/ilm/projecten/meta/scripts/ooDACE/@BasicGaussianProcess/intrinsicCovarianceMatrix.m
- /home/ilm/projecten/meta/scripts/ooDACE/@BasicGaussianProcess/likelihood.m
- /home/ilm/projecten/meta/scripts/ooDACE/@BasicGaussianProcess/marginalLikelihood.m
- /home/ilm/projecten/meta/scripts/ooDACE/@BasicGaussianProcess/mseTestset.m
- /home/ilm/projecten/meta/scripts/ooDACE/@BasicGaussianProcess/plotLikelihood.m
- /home/ilm/projecten/meta/scripts/ooDACE/@BasicGaussianProcess/plotVariogram.m
- /home/ilm/projecten/meta/scripts/ooDACE/@BasicGaussianProcess/predict.m
- /home/ilm/projecten/meta/scripts/ooDACE/@BasicGaussianProcess/predict_derivatives.m
- /home/ilm/projecten/meta/scripts/ooDACE/@BasicGaussianProcess/pseudoLikelihood.m
- /home/ilm/projecten/meta/scripts/ooDACE/@BasicGaussianProcess/rcValues.m
- /home/ilm/projecten/meta/scripts/ooDACE/@BasicGaussianProcess/regressionFunction.m
- /home/ilm/projecten/meta/scripts/ooDACE/@BasicGaussianProcess/regressionMatrix.m
- /home/ilm/projecten/meta/scripts/ooDACE/@BasicGaussianProcess/tuneParameters.m
- /home/ilm/projecten/meta/scripts/ooDACE/@BasicGaussianProcess/updateModel.m
- /home/ilm/projecten/meta/scripts/ooDACE/@BasicGaussianProcess/updateRegression.m
- /home/ilm/projecten/meta/scripts/ooDACE/@BasicGaussianProcess/updateStochasticProcess.m

7.2 BlindKriging Class Reference

A blind kriging surrogate model.

Inheritance diagram for BlindKriging:



Public Member Functions

- function BlindKriging (var varargin)
  Class constructor.
- function getStatistics ()
  Returns some useful statistics.
- function fit (var samples, var values)
  Fits a blind kriging model.
- function regressionFunction (var varargin)
  Returns the regression function.
- function getProcessVariance ()
  Returns the process variance.
- function getSamples ()
  Returns the input sample matrix.
- function getValues ()
  Returns the output value matrix.
- function predict (var points)
  Predict the mean and/or variance for one or more points x.
- function predict_derivatives (var point)
  Predict the derivatives of the mean and/or variance for a points x.
- function getExpression (var outputIndex)
  Returns the Matlab expression of this Gaussian Process model.
- function cvpe ()
  Calculates cross validated prediction error (cvpe)
- function getHyperparameters ()
  Returns the hyperparameters.
- function getRho ()
  Returns the rho parameter (only valid for `CoKriging`)

- function getCorrelationMatrix ()

  Returns the full extrinsic correlation matrix.
- function getSigma ()

  Returns the intrinsic covariance matrix.
- function getRegressionMatrix ()

  Returns the model matrix (Vandermonde matrix)
- function setOption (var key, var value)

  Sets a value in the options structure.
- function cleanup ()

  Clears some unused variables.
- function display ()

  Returns user-friendly description of the class instance.
- function predict_limit (var points)

  Limit predictor of kriging (EXPERIMENTAL)
- function correlationFunction (var options)

  Returns the internal correlation function handle and a symbolic expression of the the correlation part.
- function marginalLikelihood (var dpsi, var dsigma2)

  Marginal likelihood function.
- function pseudoLikelihood (var dpsi, var dsigma2)

  Leave-one-out predictive log probability (pseudo-likelihood)
- function mseTestset (var testx, var testy)

  Calculates error on holdout set.
- function imse ()

  Calculates the log of the integrated mean squared error.
- function rcValues ()

  Quantifies magnification of noise (lower is better)
- function plotVariogram ()

  Variogram plot (EXPERIMENTAL)

Static Public Member Functions

- static function getDefaultOptions ()

  Returns a default options structure.

Public Attributes

- const var RHO

  index of the rho parameter
- const var LAMBDA

  index of the lambda parameter
- const var SIGMA2

  index of the sigma2 parameter
- const var HP

  index of the correlation function parameters

Protected Member Functions

- function regressionMatrix (var points)

  Constructs regression matrix.
- function setData (var samples, var values)

  Sets samples and values matrix.
- function updateModel (var F, var hp)

  Constructs model.
- function updateRegression (var F, var hp)

  Constructs regression part.
- function updateStochasticProcess (var hp)

  Constructs correlation part.
- function extrinsicCorrelationMatrix (var points1, var points2)

  Constructs extrinsic correlation matrix.
- function intrinsicCovarianceMatrix (var points1, var points2)

  Constructs intrinsic covariance matrix (stochastic kriging/regression kriging)
- function tuneParameters (var F)

  Hyperparameter optimization.
- function generateDegrees (var dj)

  Generate degrees matrix from individual ones.

Protected Attributes

- var options
- var regressionFcn

  degrees matrix (strings are converted)
- var correlationFcn

  string -> function handle
- var hyperparameters0

  initial hp values OR the real ones (if optimization is done outside the class)
- var dist

  sample inter-distance
- var distIdxPsi

  indexing needed to calculate psi from D
- var optimIdx

  logical indices to parameters that are optimized
- var optimNrParameters

  number of optimization parameter (vector; one entry per type of parameter)
- var alpha

  Regression coefficients.
- var gamma

  'Correlation part' coefficients
- var hyperparameters

  correlation parameters
- var rho
- var C

  Choleski Decomposition of extrinsic + intrinsic matrices.
- var sigma2

  process variance of the GP (extrinsic variance)
- var tau2

  intrinsic variance

- var Sigma

  intrinsic covariance matrix (amount of regression of stochastic part)
- var Ft

  decorrelated model matrix
- var R

  from QR decomposition of regression part
- var sigma2_reinterp

  Reinterpolation version of `sigma2`.
- var C_reinterp

  Reinterpolation version of `C`.
- var Ft_reinterp

  Reinterpolation version of `Ft`.
- var R_reinterp

  Reinterpolation version of `R`.

7.2.1 Detailed Description

A blind kriging surrogate model.

Papers:

- "Blind Kriging: A New Method for Developing Metamodels", V.R. Joseph and Y. Hung and A. Sudjianto, ASME Journal of Mechanical Design, 2008
- "Functionally Induced Priors for the Analysis of Experiments", V.R. Joseph and J.D. Delaney, Technometrics, 2007

Limitations:

- quartic is the highest order supported

7.2.2 Constructor & Destructor Documentation

7.2.2.1 function BlindKriging (var *varargin*) [inline]

Class constructor.

Initializes the Blind kriging model. Takes the same parameters as `Kriging`

Returns

instance of the blind kriging class

7.2.3 Member Function Documentation

7.2.3.1 function cleanup ( ) [inline], [inherited]

Clears some unused variables.

7.2.3.2 function correlationFunction (var *options*) [inherited]

Returns the internal correlation function handle and a symbolic expression of the the correlation part.

Example: [correlationFcn expression] = correlationFunction(this,struct('latex', true, 'includeCoefficients', false))

The expression is based on the scaled data

Parameters

options	Options struct

Return values

correlationFcn	String of correlation function
expression	Symbolic expression

7.2.3.3 function cvpe ( ) [inherited]

Calculates cross validated prediction error (cvpe)

Error function used is the Mean Squared Error (MSE)

Return values

out	cvpe score

7.2.3.4 function display ( ) [inline], [inherited]

Returns user-friendly description of the class instance.

7.2.3.5 function extrinsicCorrelationMatrix (var *points1*, var *points2*) [protected], [inherited]

Constructs extrinsic correlation matrix.

Generate correlation matrix of points1 vs points2 using the current hyperparameters as followed:

Parameters

points1	input point matrix (optional)
points2	input point matrix (optional)

Return values

psi	Correlation matrix
dpsi	Derivative of correlation matrix w.r.t. the hyperparameters

- extrinsicCorrelationMatrix(this)
  - Assume points1 = points2 = samples
- extrinsicCorrelationMatrix(this, points1)
  - Assume points2 = samples
- extrinsicCorrelationMatrix(this, points1, points2)

NOTE: The first case returns derivatives w.r.t. X The latter two cases returns derivatives w.r.t. the hyperparameters

7.2.3.6 function fit (var *samples*, var *values*)

Fits a blind kriging model.

Need to be invoked before calling any of the prediction methods.

Parameters

samples	input sample matrix
values	output value matrix

Return values

IK	The initial kriging model

7.2.3.7 function generateDegrees (var *dj*) [protected],[inherited]

Generate degrees matrix from individual ones.

Generates a (full) degree matrix based on one or more degree matrices for variable j The option 'regressionMaxLevelInteractions' determines the maximum level of interactions (BasicGaussianProcess::getDefaultOptions)

Parameters

dj	Cell array of degree matrices (for each dimension)

Return values

degrees	Full degrees matrix
idx	Cell array of indices (used by blind kriging)

7.2.3.8 function getCorrelationMatrix ( ) [inline],[inherited]

Returns the full extrinsic correlation matrix.

7.2.3.9 static function getDefaultOptions ( ) [inline],[static]

Returns a default options structure.

Blind kriging specific options:

```
options.regressionMetric = 'cvpe'; % metric to guide the feature
      selection phase
options.retuneParameters = false; % retune parameters after every BK
      step
options.RmatrixThreshold = 250; % when to use full matrix R or when to
      approximate by using sparse diagonal of R
options.regressionMaxOrder = 2; % maximum order of candidate feature to
      consider (quadratic)
```

Return values

options	Options structure

7.2.3.10 function getExpression (var *outputIndex*) [inherited]

Returns the Matlab expression of this Gaussian Process model.

Example: expression = getExpression(this)

Parameters

options	Options struct

Return values

expression	Symbolic expression

7.2.3.11 function getHyperparameters ( ) [inline],[inherited]

Returns the hyperparameters.

7.2.3.12 function getProcessVariance ( ) [inline], [inherited]

Returns the process variance.

Return values

sigma2	process variance

7.2.3.13 function getRegressionMatrix ( ) [inline], [inherited]

Returns the model matrix (Vandermonde matrix)

7.2.3.14 function getRho ( ) [inline], [inherited]

Returns the rho parameter (only valid for CoKriging)

7.2.3.15 function getSamples ( ) [inline], [inherited]

Returns the input sample matrix.

Return values

samples	unscaled samples (original)
scaledSamples	scaled samples

7.2.3.16 function getSigma ( ) [inline], [inherited]

Returns the intrinsic covariance matrix.

7.2.3.17 function getStatistics ( ) [inline]

Returns some useful statistics.

Return values

stats	Statistics structure

7.2.3.18 function getValues ( ) [inline], [inherited]

Returns the output value matrix.

Return values

values	unscaled values (original)
scaledValues	scaled values

7.2.3.19 function imse ( ) [inherited]

Calculates the log of the integrated mean squared error.

Only supported for 1D and 2D problems.

Return values

out	log(imse) score

Papers:

- See Sacks 1989

- See pechiny paper...

Todo Implement generic monte carlo integration

7.2.3.20 function intrinsicCovarianceMatrix (var *points1*, var *points2*) [protected],[inherited]

Constructs intrinsic covariance matrix (stochastic kriging/regression kriging)

Generate covariance matrix matrix of points1 vs points2 using the current hyperparameters as followed:

Parameters

points1	input point matrix (optional)
points2	input point matrix (optional)

Return values

psi	Covariance matrix
dpsi	Derivative of covariance matrix w.r.t. the hyperparameters OR the input points

- intrinsicCovarianceMatrix(this)
  - Assume points1 = points2 = samples
- intrinsicCovarianceMatrix(this, points1)
  - Not used (and not implemented yet)
- intrinsicCovarianceMatrix(this, points1, points2)
  - Not used (and not implemented yet)

7.2.3.21 function marginalLikelihood (var *dpsi*, var *dsigma2*) [inherited]

Marginal likelihood function.

Used for Maximum Likelihood Estimation (MLE)

Parameters

dpsi	cell array of derivative matrices (optional; for internal use only)

Return values

out	score
dout	Derivatives w.r.t. hyperparameters

Papers:

- "Gaussian Processes for Machine Learning" (Chapter 5), C. E. Rasmussen and C. K. I. Williams, MIT Press, 2006
- "An adjoint for likelihood maximization" D.J.J. Toal, A.I.J. Forrester, N.W. Bressloff, A.J. Keane, C. Holden, Proc. of the Royal Society, 2009

Todo Adjoint derivatives work, but are very slow due to naive implementation

7.2.3.22 function mseTestset (var *testx*, var *testy*) [inherited]

Calculates error on holdout set.

Error function used is the Mean Squared Error (MSE) function.

Parameters

testx	input samples of the test set
testy	output samples of the test set

Return values

out	mse error on test set

7.2.3.23 function plotVariogram ( ) [inherited]

Variogram plot (EXPERIMENTAL)

Plots the experimental (semi-)variogram (based on the data) as well as the theoretical kriging (semi-)variogram (defined by the correlation function).

Return values

h	Figure handle

Empirical variogram:

- Methods of moments estimator Matheron (Cressie, 1993)
- More robust estimators: Cressie-Hawkins (1980), Genton (1998) In general: use median instead of mean

7.2.3.24 function predict (var *points*) [inherited]

Predict the mean and/or variance for one or more points x.

Parameters

points	Matrix of input points to be predicted

Return values

values	predicted output values
sigma2	predicted variance of the output values (optional)

Precondition

The kriging object should be fitted using a dataset (`fit`)

7.2.3.25 function predict_derivatives (var *point*) [inherited]

Predict the derivatives of the mean and/or variance for a points x.

NOTE:

Parameters

point	input point to calculate the derivative of

Return values

dvalues	Derivative w.r.t. the output
sigma2	Derivative w.r.t. the output variance (optional)

- limited to one point at a time (x is a row vector)

7.2.3.26 function predict_limit (var *points*) [inherited]

Limit predictor of kriging (EXPERIMENTAL)

Parameters

points	Matrix of input points to be predicted

Return values

values	predicted output values
sigma2	predicted variance of the output values (optional)

7.2.3.27 function pseudoLikelihood (var *dpsi,* var *dsigma2*) [inherited]

Leave-one-out predictive log probability (pseudo-likelihood)

Calculates the leave-one-out predictive log probability.

Parameters

dpsi	cell array of derivative matrices (optional; for internal use only)

Return values

out	score
dout	Derivatives w.r.t. hyperparameters

Papers:

- "Predictive Approaches for Choosing Hyperparameters in Gaussian Process", S. Sundararajan, S.S. Keerthu, 1999

7.2.3.28 function rcValues ( ) [inherited]

Quantifies magnification of noise (lower is better)

Robustness-criterion (In theory useful only for ordinary kriging).

Return values

rc	robustness-criterion

Returns a 2xn matrix:

- the first row contains the absolute robustness
- the second row contains the relative robustness Papers:
- "Kriging models that are robust w.r.t. simulation errors" A.Y.D. Siem, D. den Hertog (tech report)

7.2.3.29 function regressionFunction (var *varargin*)

Returns the regression function.

Example: [regressionFcn expression terms] = regressionFunction(this, struct('latex', true, 'precision', '%.5g'))

Parameters

varargin	Options

Return values

regressionFcn	Degree matrix representing the regression function
expression	Symbolic expression
terms	Cell array of the individual terms

Note

Symbolic expression does not support maxorder greater than two

7.2.3.30 function regressionMatrix (var *points*) [protected]

Constructs regression matrix.

Regression matrix (model matrix, Vandermonde matrix, ...) for a set of points Based on this.regressionFcn.

Parameters

points	points matrix (optional)

Return values

F	model matrix
dF	derivative of the model matrix w.r.t. points or the hyperparameters

Uses coded sample matrix!

7.2.3.31 function setData (var *samples,* var *values*) [protected], [inherited]

Sets samples and values matrix.

Scales samples and values, passing the scaled dataset to the underlying base method.

Parameters

samples	input sample matrix
values	output value matrix

7.2.3.32 function setOption (var *key,* var *value*) [inline], [inherited]

Sets a value in the options structure.

Parameters

key	option name
value	option value

7.2.3.33 function tuneParameters (var *F*) [protected], [inherited]

Hyperparameter optimization.

Setups and invokes the optimizer.

Parameters

F	model matrix

Return values

optimHp	optimized hyperparameters
perf	Performance score (likelihood score)

7.2.3.34 function updateModel (var *F*, var *hp*) [protected], [inherited]

Constructs model.

Full update of the model (regression + correlation part)

Parameters

F	model matrix
hp	new hyperparameters

7.2.3.35 function updateRegression (var *F*, var *hp*) [protected], [inherited]

Constructs regression part.

Updates regression part of the model.

Parameters

F	model matrix
hp	new hyperparameters

Return values

err	error string (if any)

Todo Rho is only used by co-kriging, can we abstract this somehow ?

7.2.3.36 function updateStochasticProcess (var *hp*) [protected], [inherited]

Constructs correlation part.

Updates correlation part of the model.

Parameters

hp	hyperparameters

Return values

err	error string (if any)
dpsi	Derivative of correlation matrix w.r.t. the hyperparameters

7.2.4 Member Data Documentation

7.2.4.1 var alpha [protected], [inherited]

Regression coefficients.

7.2.4.2 var C [protected], [inherited]

Choleski Decomposition of extrinsic + intrinsic matrices.

7.2.4.3 var C_reinterp [protected], [inherited]

Reinterpolation version of `C`.

7.2.4.4 var correlationFcn [protected], [inherited]

string -> function handle

7.2.4.5 var dist [protected], [inherited]

sample inter-distance

7.2.4.6 var distIdxPsi [protected], [inherited]

indexing needed to calculate psi from D

7.2.4.7 var Ft [protected], [inherited]

decorrelated model matrix

7.2.4.8 var Ft_reinterp [protected], [inherited]

Reinterpolation version of `Ft`.

7.2.4.9 var gamma [protected], [inherited]

'Correlation part' coefficients

7.2.4.10 const var HP [inherited]

index of the correlation function parameters

7.2.4.11 var hyperparameters [protected], [inherited]

correlation parameters

7.2.4.12 var hyperparameters0 [protected], [inherited]

initial hp values OR the real ones (if optimization is done outside the class)

7.2.4.13 const var LAMBDA [inherited]

index of the lambda parameter

7.2.4.14 var optimIdx [protected], [inherited]

logical indices to parameters that are optimized

7.2.4.15 var optimNrParameters [protected], [inherited]

number of optimization parameter (vector; one entry per type of parameter)

7.2.4.16 var options [protected], [inherited]

7.2.4.17 var R [protected], [inherited]

from QR decomposition of regression part

7.2.4.18 var R_reinterp [protected],[inherited]

Reinterpolation version of R.

7.2.4.19 var regressionFcn [protected],[inherited]

degrees matrix (strings are converted)

7.2.4.20 const var RHO [inherited]

index of the rho parameter

7.2.4.21 var rho [protected],[inherited]

7.2.4.22 var Sigma [protected],[inherited]

intrinsic covariance matrix (amount of regression of stochastic part)

7.2.4.23 const var SIGMA2 [inherited]

index of the sigma2 parameter

7.2.4.24 var sigma2 [protected],[inherited]

process variance of the GP (extrinsic variance)

7.2.4.25 var sigma2_reinterp [protected],[inherited]

Reinterpolation version of sigma2.

Note

reinterpolation: might be nicer to just construct and keep a sub-GP... takes more space, some calculations are done twice but performance shouldn't take a very big hit...

7.2.4.26 var tau2 [protected],[inherited]

intrinsic variance

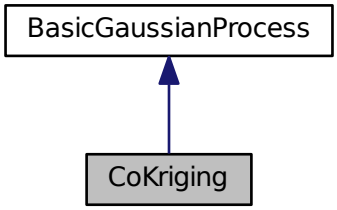
The documentation for this class was generated from the following files:

- /home/ilm/projecten/meta/scripts/ooDACE/@BlindKriging/BlindKriging.m
- /home/ilm/projecten/meta/scripts/ooDACE/@BlindKriging/fit.m
- /home/ilm/projecten/meta/scripts/ooDACE/@BlindKriging/polynomialCoding.m
- /home/ilm/projecten/meta/scripts/ooDACE/@BlindKriging/posteriorBeta.m
- /home/ilm/projecten/meta/scripts/ooDACE/@BlindKriging/regressionFunction.m
- /home/ilm/projecten/meta/scripts/ooDACE/@BlindKriging/regressionMatrix.m
- /home/ilm/projecten/meta/scripts/ooDACE/@BlindKriging/Rmatrix.m

7.3 CoKriging Class Reference

A cokriging surrogate model.

Inheritance diagram for CoKriging:



Public Member Functions

- function CoKriging (var varargin)
- function getSamplesIdx (var t)

  Returns samples of dataset t.
- function getValuesIdx (var t)

  Returns values of dataset t.
- function fit (var samples, var values)

  Fits a gaussian process for multi-fidelity datasets.
- function regressionFunction (var varargin)

  Returns the regression function.
- function correlationFunction (var varargin)

  Returns the internal correlation function handle.
- function getHyperparameters ()

  Returns the hyperparameters.
- function getRho ()

  Returns the rho parameter (only valid for `CoKriging`)
- function getProcessVariance ()

  Returns the process variance (sigma2)
- function getCorrelationMatrix ()

  Returns the full extrinsic correlation matrix.
- function getSigma ()

  Returns the intrinsic covariance matrix.
- function getRegressionMatrix ()

  Returns the model matrix (Vandermonde matrix)
- function getSamples ()

  Returns the input sample matrix.
- function getValues ()

  Returns the output value matrix.
- function setOption (var key, var value)

  Sets a value in the options structure.
- function cleanup ()

  Clears some unused variables.
- function display ()

  Returns user-friendly description of the class instance.
- function predict (var points)

Predict the mean and/or variance for one or more points x.

- function predict_derivatives (var point)

  Predict the derivatives of the mean and/or variance for a points.
- function predict_limit (var points)

  Limit predictor of kriging (EXPERIMENTAL)
- function getExpression (var options)

  Returns the Matlab expression of this Gaussian Process model.
- function cvpe ()

  Calculates the log of the leave-one-out cross validation error (LOO-CV)
- function marginalLikelihood (var dpsi, var dsigma2)

  Marginal likelihood function.
- function pseudoLikelihood (var dpsi, var dsigma2)

  Leave-one-out predictive log probability (pseudo-likelihood)
- function mseTestset (var testx, var testy)

  Calculates error on holdout set.
- function imse ()

  Calculates the log of the integrated mean squared error.
- function rcValues ()

  Quantifies magnification of noise (lower is better)
- function plotVariogram ()

  Variogram plot (EXPERIMENTAL)

Static Public Member Functions

- static function getDefaultOptions ()

  Returns a default options structure.

Public Attributes

- const var RHO

  index of the rho parameter
- const var LAMBDA

  index of the lambda parameter
- const var SIGMA2

  index of the sigma2 parameter
- const var HP

  index of the correlation function parameters

Protected Member Functions

- function setData (var samples, var values)

  Sets samples and values matrix.
- function extrinsicCorrelationMatrix (var points1, var points2)

  Constructs extrinsic correlation matrix.
- function intrinsicCovarianceMatrix (var points1, var points2)

  Constructs intrinsic covariance matrix (stochastic kriging/regression kriging)
- function regressionMatrix (var points)

  Constructs regression matrix.
- function updateModel (var F, var hp)

  Constructs model.

- function updateRegression (var F, var hp)

  Constructs regression part.
- function updateStochasticProcess (var hp)

  Constructs correlation part.
- function tuneParameters (var F)

  Hyperparameter optimization.
- function generateDegrees (var dj)

  Generate degrees matrix from individual ones.

Protected Attributes

- var options
- var regressionFcn

  degrees matrix (strings are converted)
- var correlationFcn

  string -> function handle
- var hyperparameters0

  initial hp values OR the real ones (if optimization is done outside the class)
- var dist

  sample inter-distance
- var distIdxPsi

  indexing needed to calculate psi from D
- var optimIdx

  logical indices to parameters that are optimized
- var optimNrParameters

  number of optimization parameter (vector; one entry per type of parameter)
- var alpha

  Regression coefficients.
- var gamma

  'Correlation part' coefficients
- var hyperparameters

  correlation parameters
- var rho
- var C

  Choleski Decomposition of extrinsic + intrinsic matrices.
- var sigma2

  process variance of the GP (extrinsic variance)
- var tau2

  intrinsic variance
- var Sigma

  intrinsic covariance matrix (amount of regression of stochastic part)
- var Ft

  decorrelated model matrix
- var R

  from QR decomposition of regression part
- var sigma2_reinterp

  Reinterpolation version of `sigma2`.
- var C_reinterp

  Reinterpolation version of `C`.
- var Ft_reinterp

  Reinterpolation version of `Ft`.
- var R_reinterp

  Reinterpolation version of `R`.

7.3.1 Detailed Description

A cokriging surrogate model.

Papers:

- "Bayesian Analysis of Computer Code Outputs" M. Kennedy, A. O'Hagan, N. Higgins (2001)
- "Multi-fidelity optimization via surrogate modelling" A.I.J Forrester, A. Sobester, A.J. Keane (2007)

Limitations:

- Supports 2 fidelity datasets
- Likelihood function doesn't support gradients

Todo Generalize to an arbitrary number of (multi-fidelity) datasets

7.3.2 Constructor & Destructor Documentation

7.3.2.1 function CoKriging (var *varargin*) [inline]

7.3.3 Member Function Documentation

7.3.3.1 function cleanup ( ) [inline],[inherited]

Clears some unused variables.

7.3.3.2 function correlationFunction (var *varargin*)

Returns the internal correlation function handle.

Symbolic expression not supported for cokriging.

Parameters

varargin	Options

Return values

correlationFcn	String of correlation function

7.3.3.3 function cvpe ( ) [inherited]

Calculates the log of the leave-one-out cross validation error (LOO-CV)

Error function used is the Mean Squared Error (MSE)

Return values

out	log(loo-cv) score

Papers:

- "Blind Kriging: A New Method for Developing Metamodels", V.R. Joseph and Y. Hung and A. Sudjianto, ASME Journal of Mechanical Design, 2008
- "Predictive Approaches for Choosing Hyperparameters in Gaussian Process", S. Sundararajan, S.S. Keerthu, 1999

7.3.3.4 function display ( ) [inline],[inherited]

Returns user-friendly description of the class instance.

7.3.3.5 function extrinsicCorrelationMatrix (var *points1*, var *points2*) [protected]

Constructs extrinsic correlation matrix.

Generate correlation matrix of points1 vs points2 using the current hyperparameters as followed:

Parameters

points1	input point matrix (optional)
points2	input point matrix (optional)

Return values

psi	Correlation matrix
dpsi	Derivative of correlation matrix w.r.t. the hyperparameters

- extrinsicCorrelationMatrix(this)
  - Assume points1 = points2 = samples
- extrinsicCorrelationMatrix(this, points1)
  - Assume points2 = samples
- extrinsicCorrelationMatrix(this, points1, points2)

NOTE: The first case returns derivatives w.r.t. X The latter two cases returns derivatives w.r.t. the hyperparameters

Issues:

- Actually returns a covariance matrix instead of a correlation matrix shouldn't be a problem as long as an extrinsicCorrelationMatrix method doesn't mix both types...

7.3.3.6 function fit (var *samples*, var *values*)

Fits a gaussian process for multi-fidelity datasets.

Need to be invoked before calling any of the prediction methods.

Parameters

samples	input sample matrix (cell array)
values	output value matrix (cell array)

samples/values are (columnwise!) cell arrays. {1} is cheap data ... more expensive ... {end} is most expensive NOTE: though only 2 datasets are supported atm (length of samples/values must be 2)

Note

Allow to update the CoKriging model with new data:

- this means no scaling as the scaling changes when new data arrives (we inherit from BasicGaussianProcess)
- no refitting with the same hyperparameters (no xval calculation)

7.3.3.7 function generateDegrees (var *dj*) [protected],[inherited]

Generate degrees matrix from individual ones.

Generates a (full) degree matrix based on one or more degree matrices for variable j The option 'regressionMaxLevelInteractions' determines the maximum level of interactions (BasicGaussianProcess::getDefaultOptions)

Parameters

dj	Cell array of degree matrices (for each dimension)

Return values

degrees	Full degrees matrix
idx	Cell array of indices (used by blind kriging)

7.3.3.8 function getCorrelationMatrix ( ) [inline], [inherited]

Returns the full extrinsic correlation matrix.

7.3.3.9 static function getDefaultOptions ( ) [inline], [static]

Returns a default options structure.

Return values

options	Options structure

7.3.3.10 function getExpression (var *options*) [inherited]

Returns the Matlab expression of this Gaussian Process model.

Example: expression = getExpression(this)

Parameters

options	Options struct

Return values

expression	Symbolic expression

7.3.3.11 function getHyperparameters ( ) [inline], [inherited]

Returns the hyperparameters.

7.3.3.12 function getProcessVariance ( ) [inline], [inherited]

Returns the process variance (sigma2)

7.3.3.13 function getRegressionMatrix ( ) [inline], [inherited]

Returns the model matrix (Vandermonde matrix)

7.3.3.14 function getRho ( ) [inline], [inherited]

Returns the rho parameter (only valid for CoKriging)

7.3.3.15 function getSamples ( ) [inline], [inherited]

Returns the input sample matrix.

7.3.3.16 function getSamplesIdx (var *t*) [inline]

Returns samples of dataset t.

Parameters

t	index of dataset to retrieve

Return values

samples	samples of dataset t

7.3.3.17 function getSigma ( ) [inline], [inherited]

Returns the intrinsic covariance matrix.

7.3.3.18 function getValues ( ) [inline], [inherited]

Returns the output value matrix.

7.3.3.19 function getValuesIdx (var *t*) [inline]

Returns values of dataset t.

Parameters

t	index of dataset to retrieve

Return values

values	values of dataset t

7.3.3.20 function imse ( ) [inherited]

Calculates the log of the integrated mean squared error.

Only supported for 1D and 2D problems.

Return values

out	log(imse) score

Papers:

- See Sacks 1989
- See pechiny paper...

Todo Implement generic monte carlo integration

7.3.3.21 function intrinsicCovarianceMatrix (var *points1*, var *points2*) [protected]

Constructs intrinsic covariance matrix (stochastic kriging/regression kriging)

Generate covariance matrix matrix of points1 vs points2 using the current hyperparameters as followed:

Parameters

points1	input point matrix (optional)
points2	input point matrix (optional)

Return values

psi	Covariance matrix
dpsi	Derivative of covariance matrix w.r.t. the hyperparameters OR the input points

- intrinsicCovarianceMatrix(this)
  - Assume points1 = points2 = samples
- intrinsicCovarianceMatrix(this, points1)
  - Not used (and not implemented yet)
- intrinsicCovarianceMatrix(this, points1, points2)
  - Not used (and not implemented yet)

7.3.3.22 function marginalLikelihood (var *dpsi*, var *dsigma2*) [inherited]

Marginal likelihood function.

Used for Maximum Likelihood Estimation (MLE)

Parameters

dpsi	cell array of derivative matrices (optional; for internal use only)

Return values

out	score
dout	Derivatives w.r.t. hyperparameters

Papers:

- "Gaussian Processes for Machine Learning" (Chapter 5), C. E. Rasmussen and C. K. I. Williams, MIT Press, 2006
- "An adjoint for likelihood maximization" D.J.J. Toal, A.I.J. Forrester, N.W. Bressloff, A.J. Keane, C. Holden, Proc. of the Royal Society, 2009

Todo Adjoint derivatives work, but are very slow due to naive implementation

7.3.3.23 function mseTestset (var *testx*, var *testy*) [inherited]

Calculates error on holdout set.

Error function used is the Mean Squared Error (MSE) function.

Parameters

testx	input samples of the test set
testy	output samples of the test set

Return values

out	mse error on test set

7.3.3.24 function plotVariogram ( ) [inherited]

Variogram plot (EXPERIMENTAL)

Plots the experimental (semi-)variogram (based on the data) as well as the theoretical kriging (semi-)variogram (defined by the correlation function).

Return values

h	Figure handle

Empirical variogram:

- Methods of moments estimator Matheron (Cressie, 1993)
- More robust estimators: Cressie-Hawkins (1980), Genton (1998) In general: use median instead of mean

7.3.3.25 function predict (var *points*) [inherited]

Predict the mean and/or variance for one or more points x.

Parameters

points	Matrix of input points to be predicted

Return values

values	predicted output values
sigma2	predicted variance of the output values (optional)

Precondition

The kriging object should be fitted using a dataset (BasicGaussianProcess::fit)

7.3.3.26 function predict_derivatives (var *point*) [inherited]

Predict the derivatives of the mean and/or variance for a points.

NOTE:

Parameters

point	input point to calculate the derivative of

Return values

dvalues	Derivative w.r.t. the output
sigma2	Derivative w.r.t. the output variance (optional)

- limited to one point at a time (x is a vector)

Precondition

The kriging object should be fitted using a dataset (BasicGaussianProcess::fit)

7.3.3.27 function predict_limit (var *points*) [inherited]

Limit predictor of kriging (EXPERIMENTAL)

Parameters

points	Matrix of input points to be predicted

Return values

values	predicted output values
sigma2	predicted variance of the output values (optional)

7.3.3.28 function pseudoLikelihood (var *dpsi,* var *dsigma2*) [inherited]

Leave-one-out predictive log probability (pseudo-likelihood)

Calculates the leave-one-out predictive log probability.

Parameters

dpsi	cell array of derivative matrices (optional; for internal use only)

Return values

out	score
dout	Derivatives w.r.t. hyperparameters

Papers:

- "Predictive Approaches for Choosing Hyperparameters in Gaussian Process", S. Sundararajan, S.S. Keerthu, 1999

7.3.3.29 function rcValues ( ) [inherited]

Quantifies magnification of noise (lower is better)

Robustness-criterion (In theory useful only for ordinary kriging).

Return values

rc	robustness-criterion

Returns a 2xn matrix:

- the first row contains the absolute robustness
- the second row contains the relative robustness Papers:
- "Kriging models that are robust w.r.t. simulation errors" A.Y.D. Siem, D. den Hertog (tech report)

7.3.3.30 function regressionFunction (var *varargin*)

Returns the regression function.

Symbolic expression and terms not supported for cokriging.

Parameters

varargin	Options

Return values

regressionFcn	Degree matrix representing the regression function
expression	Symbolic expression
terms	Cell array of the individual terms

7.3.3.31 function regressionMatrix (var *points*) [protected]

Constructs regression matrix.

Regression matrix (model matrix, Vandermonde matrix, ...) for a set of points Based on this.regressionFcn.

Parameters

points	input point matrix (optional)

Return values

F	Model matrix
dF	Derivative of model matrix w.r.t. the hyperparameters OR the input points

7.3.3.32 function setData (var *samples,* var *values*) [protected]

Sets samples and values matrix.

Concatenate sample/values cell array to numeric array passing the resulting dataset to the underlying base method.

Parameters

samples	input sample matrix (cell array)
values	output value matrix (cell array)

7.3.3.33 function setOption (var *key,* var *value*) [inline], [inherited]

Sets a value in the options structure.

Parameters

key	option name
value	option value

7.3.3.34 function tuneParameters (var *F*) [protected], [inherited]

Hyperparameter optimization.

Setups and invokes the optimizer.

Parameters

F	model matrix

Return values

optimHp	optimized hyperparameters
perf	Performance score (likelihood score)

7.3.3.35 function updateModel (var *F,* var *hp*) [protected], [inherited]

Constructs model.

Full update of the model (regression + correlation part)

Parameters

F	model matrix
hp	new hyperparameters

7.3.3.36 function updateRegression (var *F*, var *hp*) [protected], [inherited]

Constructs regression part.

Updates regression part of the model.

Parameters

F	model matrix
hp	new hyperparameters

Return values

err	error string (if any)

Todo Rho is only used by co-kriging, can we abstract this somehow ?

7.3.3.37 function updateStochasticProcess (var *hp*) [protected], [inherited]

Constructs correlation part.

Updates correlation part of the model.

Parameters

hp	hyperparameters

Return values

err	error string (if any)
dpsi	Derivative of correlation matrix w.r.t. the hyperparameters

7.3.4 Member Data Documentation

7.3.4.1 var alpha [protected], [inherited]

Regression coefficients.

7.3.4.2 var C [protected], [inherited]

Choleski Decomposition of extrinsic + intrinsic matrices.

7.3.4.3 var C_reinterp [protected], [inherited]

Reinterpolation version of C.

7.3.4.4 var correlationFcn [protected], [inherited]

string -> function handle

7.3.4.5 var dist [protected], [inherited]

sample inter-distance

7.3.4.6 var distIdxPsi [protected], [inherited]

indexing needed to calculate psi from D

7.3.4.7 var Ft [protected],[inherited]

decorrelated model matrix

7.3.4.8 var Ft_reinterp [protected],[inherited]

Reinterpolation version of `Ft`.

7.3.4.9 var gamma [protected],[inherited]

'Correlation part' coefficients

7.3.4.10 const var HP [inherited]

index of the correlation function parameters

7.3.4.11 var hyperparameters [protected],[inherited]

correlation parameters

7.3.4.12 var hyperparameters0 [protected],[inherited]

initial hp values OR the real ones (if optimization is done outside the class)

7.3.4.13 const var LAMBDA [inherited]

index of the lambda parameter

7.3.4.14 var optimIdx [protected],[inherited]

logical indices to parameters that are optimized

7.3.4.15 var optimNrParameters [protected],[inherited]

number of optimization parameter (vector; one entry per type of parameter)

7.3.4.16 var options [protected],[inherited]

7.3.4.17 var R [protected],[inherited]

from QR decomposition of regression part

7.3.4.18 var R_reinterp [protected],[inherited]

Reinterpolation version of `R`.

7.3.4.19 var regressionFcn [protected],[inherited]

degrees matrix (strings are converted)

7.3.4.20 const var RHO [inherited]

index of the rho parameter

7.3.4.21 var rho [protected],[inherited]

7.3.4.22 var Sigma [protected],[inherited]

intrinsic covariance matrix (amount of regression of stochastic part)

7.3.4.23 const var SIGMA2 [inherited]

index of the sigma2 parameter

7.3.4.24 var sigma2 [protected],[inherited]

process variance of the GP (extrinsic variance)

7.3.4.25 var sigma2_reinterp [protected],[inherited]

Reinterpolation version of `sigma2`.

Note

reinterpolation: might be nicer to just construct and keep a sub-GP... takes more space, some calculations are done twice but performance shouldn't take a very big hit...

7.3.4.26 var tau2 [protected],[inherited]

intrinsic variance

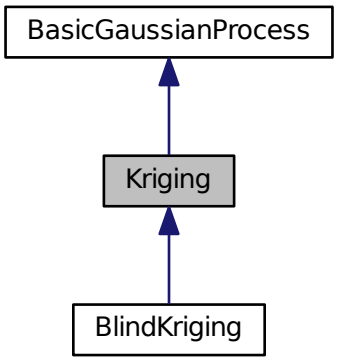
The documentation for this class was generated from the following files:

- /home/ilm/projecten/meta/scripts/ooDACE/@CoKriging/CoKriging.m
- /home/ilm/projecten/meta/scripts/ooDACE/@CoKriging/correlationFunction.m
- /home/ilm/projecten/meta/scripts/ooDACE/@CoKriging/extrinsicCorrelationMatrix.m
- /home/ilm/projecten/meta/scripts/ooDACE/@CoKriging/fit.m
- /home/ilm/projecten/meta/scripts/ooDACE/@CoKriging/intrinsicCovarianceMatrix.m
- /home/ilm/projecten/meta/scripts/ooDACE/@CoKriging/regressionFunction.m
- /home/ilm/projecten/meta/scripts/ooDACE/@CoKriging/regressionMatrix.m
- /home/ilm/projecten/meta/scripts/ooDACE/@CoKriging/setData.m

7.4 Kriging Class Reference

A kriging surrogate model.

Inheritance diagram for Kriging:



Public Member Functions

- function Kriging (var varargin)
  Class constructor.
- function getProcessVariance ()
  Returns the process variance.
- function getSamples ()
  Returns the input sample matrix.
- function getValues ()
  Returns the output value matrix.
- function predict (var points)
  Predict the mean and/or variance for one or more points x.
- function predict_derivatives (var point)
  Predict the derivatives of the mean and/or variance for a points x.
- function getExpression (var outputIndex)
  Returns the Matlab expression of this Gaussian Process model.
- function cvpe ()
  Calculates cross validated prediction error (cvpe)
- function getHyperparameters ()
  Returns the hyperparameters.
- function getRho ()
  Returns the rho parameter (only valid for `CoKriging`)
- function getCorrelationMatrix ()
  Returns the full extrinsic correlation matrix.
- function getSigma ()
  Returns the intrinsic covariance matrix.
- function getRegressionMatrix ()
  Returns the model matrix (Vandermonde matrix)
- function setOption (var key, var value)
  Sets a value in the options structure.
- function cleanup ()
  Clears some unused variables.
- function display ()
  Returns user-friendly description of the class instance.
- function fit (var samples, var values)
  Fits a gaussian process for a dataset.
- function predict_limit (var points)
  Limit predictor of kriging (EXPERIMENTAL)
- function regressionFunction (var options)
  Returns the regression function.
- function correlationFunction (var options)
  Returns the internal correlation function handle and a symbolic expression of the the correlation part.
- function marginalLikelihood (var dpsi, var dsigma2)
  Marginal likelihood function.
- function pseudoLikelihood (var dpsi, var dsigma2)
  Leave-one-out predictive log probability (pseudo-likelihood)
- function mseTestset (var testx, var testy)
  Calculates error on holdout set.
- function imse ()
  Calculates the log of the integrated mean squared error.
- function rcValues ()
  Quantifies magnification of noise (lower is better)
- function plotVariogram ()
  Variogram plot (EXPERIMENTAL)

Static Public Member Functions

- static function getDefaultOptions ()

  Returns a default options structure.

Public Attributes

- const var RHO

  index of the rho parameter
- const var LAMBDA

  index of the lambda parameter
- const var SIGMA2

  index of the sigma2 parameter
- const var HP

  index of the correlation function parameters

Protected Member Functions

- function setData (var samples, var values)

  Sets samples and values matrix.
- function updateModel (var F, var hp)

  Constructs model.
- function updateRegression (var F, var hp)

  Constructs regression part.
- function updateStochasticProcess (var hp)

  Constructs correlation part.
- function extrinsicCorrelationMatrix (var points1, var points2)

  Constructs extrinsic correlation matrix.
- function intrinsicCovarianceMatrix (var points1, var points2)

  Constructs intrinsic covariance matrix (stochastic kriging/regression kriging)
- function regressionMatrix (var points)

  Constructs regression matrix.
- function tuneParameters (var F)

  Hyperparameter optimization.
- function generateDegrees (var dj)

  Generate degrees matrix from individual ones.

Protected Attributes

- var options
- var regressionFcn

  degrees matrix (strings are converted)
- var correlationFcn

  string -> function handle
- var hyperparameters0

  initial hp values OR the real ones (if optimization is done outside the class)
- var dist

  sample inter-distance
- var distIdxPsi

  indexing needed to calculate psi from D
- var optimIdx

logical indices to parameters that are optimized

- var optimNrParameters

  number of optimization parameter (vector; one entry per type of parameter)
- var alpha

  Regression coefficients.
- var gamma

  'Correlation part' coefficients
- var hyperparameters

  correlation parameters
- var rho
- var C

  Choleski Decomposition of extrinsic + intrinsic matrices.
- var sigma2

  process variance of the GP (extrinsic variance)
- var tau2

  intrinsic variance
- var Sigma

  intrinsic covariance matrix (amount of regression of stochastic part)
- var Ft

  decorrelated model matrix
- var R

  from QR decomposition of regression part
- var sigma2_reinterp

  Reinterpolation version of `sigma2`.
- var C_reinterp

  Reinterpolation version of `C`.
- var Ft_reinterp

  Reinterpolation version of `Ft`.
- var R_reinterp

  Reinterpolation version of `R`.

7.4.1 Detailed Description

A kriging surrogate model.

Is basically the same as the base class BasicGaussianProcess besides the scaling of the samples/values

This class is derived by KrigingModel, a class in the SUMO-Toolbox that works like an interface between SUMO and the actual kriging implementation

7.4.2 Constructor & Destructor Documentation

7.4.2.1 function Kriging (var *varargin*) [inline]

Class constructor.

Initializes the kriging model. Takes the same parameters as `BasicGaussianProcess`

Returns

instance of the kriging class

7.4.3 Member Function Documentation

7.4.3.1 function cleanup ( ) [inline],[inherited]

Clears some unused variables.

7.4.3.2 function correlationFunction (var *options*) [inherited]

Returns the internal correlation function handle and a symbolic expression of the the correlation part.

Example: [correlationFcn expression] = correlationFunction(this,struct('latex', true, 'includeCoefficients', false))

The expression is based on the scaled data

Parameters

options	Options struct

Return values

correlationFcn	String of correlation function
expression	Symbolic expression

7.4.3.3 function cvpe ( )

Calculates cross validated prediction error (cvpe)

Error function used is the Mean Squared Error (MSE)

Return values

out	cvpe score

7.4.3.4 function display ( ) [inline],[inherited]

Returns user-friendly description of the class instance.

7.4.3.5 function extrinsicCorrelationMatrix (var *points1,* var *points2*) [protected],[inherited]

Constructs extrinsic correlation matrix.

Generate correlation matrix of points1 vs points2 using the current hyperparameters as followed:

Parameters

points1	input point matrix (optional)
points2	input point matrix (optional)

Return values

psi	Correlation matrix
dpsi	Derivative of correlation matrix w.r.t. the hyperparameters

- extrinsicCorrelationMatrix(this)
  - Assume points1 = points2 = samples
- extrinsicCorrelationMatrix(this, points1)
  - Assume points2 = samples
- extrinsicCorrelationMatrix(this, points1, points2)

NOTE: The first case returns derivatives w.r.t. X The latter two cases returns derivatives w.r.t. the hyperparameters

7.4.3.6 function fit (var *samples*, var *values*) [inherited]

Fits a gaussian process for a dataset.

Need to be invoked before calling any of the prediction methods.

Parameters

samples	input sample matrix
values	output value matrix

Note

Kriging can't do a cleanup automatically, if needed cleanup() can be called manually.

7.4.3.7 function generateDegrees (var *dj*) [protected], [inherited]

Generate degrees matrix from individual ones.

Generates a (full) degree matrix based on one or more degree matrices for variable j The option 'regressionMaxLevelInteractions' determines the maximum level of interactions (BasicGaussianProcess::getDefaultOptions)

Parameters

dj	Cell array of degree matrices (for each dimension)

Return values

degrees	Full degrees matrix
idx	Cell array of indices (used by blind kriging)

7.4.3.8 function getCorrelationMatrix ( ) [inline], [inherited]

Returns the full extrinsic correlation matrix.

7.4.3.9 static function getDefaultOptions ( ) [inline], [static]

Returns a default options structure.

Return values

options	Options structure

Todo Apparantly Matlab 2008b doesn't auto forward the (static) call to BasicGaussianProcess::getDefaultOptions()

7.4.3.10 function getExpression (var *outputIndex*)

Returns the Matlab expression of this Gaussian Process model.

Example: expression = getExpression(this)

Parameters

options	Options struct

Return values

expression	Symbolic expression

7.4.3.11 function getHyperparameters ( ) [inline], [inherited]

Returns the hyperparameters.

7.4.3.12 function getProcessVariance ( ) [inline]

Returns the process variance.

Return values

sigma2	process variance

7.4.3.13 function getRegressionMatrix ( ) [inline], [inherited]

Returns the model matrix (Vandermonde matrix)

7.4.3.14 function getRho ( ) [inline], [inherited]

Returns the rho parameter (only valid for CoKriging)

7.4.3.15 function getSamples ( ) [inline]

Returns the input sample matrix.

Return values

samples	unscaled samples (original)
scaledSamples	scaled samples

7.4.3.16 function getSigma ( ) [inline], [inherited]

Returns the intrinsic covariance matrix.

7.4.3.17 function getValues ( ) [inline]

Returns the output value matrix.

Return values

values	unscaled values (original)
scaledValues	scaled values

7.4.3.18 function imse ( ) [inherited]

Calculates the log of the integrated mean squared error.

Only supported for 1D and 2D problems.

Return values

out	log(imse) score

Papers:

- See Sacks 1989
- See pechiny paper...

Todo Implement generic monte carlo integration

7.4.3.19 function intrinsicCovarianceMatrix (var *points1,* var *points2*) [protected],[inherited]

Constructs intrinsic covariance matrix (stochastic kriging/regression kriging)

Generate covariance matrix matrix of points1 vs points2 using the current hyperparameters as followed:

Parameters

points1	input point matrix (optional)
points2	input point matrix (optional)

Return values

psi	Covariance matrix
dpsi	Derivative of covariance matrix w.r.t. the hyperparameters OR the input points

- intrinsicCovarianceMatrix(this)
  - Assume points1 = points2 = samples
- intrinsicCovarianceMatrix(this, points1)
  - Not used (and not implemented yet)
- intrinsicCovarianceMatrix(this, points1, points2)
  - Not used (and not implemented yet)

7.4.3.20 function marginalLikelihood (var *dpsi,* var *dsigma2*) [inherited]

Marginal likelihood function.

Used for Maximum Likelihood Estimation (MLE)

Parameters

dpsi	cell array of derivative matrices (optional; for internal use only)

Return values

out	score
dout	Derivatives w.r.t. hyperparameters

Papers:

- "Gaussian Processes for Machine Learning" (Chapter 5), C. E. Rasmussen and C. K. I. Williams, MIT Press, 2006
- "An adjoint for likelihood maximization" D.J.J. Toal, A.I.J. Forrester, N.W. Bressloff, A.J. Keane, C. Holden, Proc. of the Royal Society, 2009

Todo Adjoint derivatives work, but are very slow due to naive implementation

7.4.3.21 function mseTestset (var *testx,* var *testy*) [inherited]

Calculates error on holdout set.

Error function used is the Mean Squared Error (MSE) function.

Parameters

testx	input samples of the test set
testy	output samples of the test set

Return values

out	mse error on test set

7.4.3.22 function plotVariogram ( ) [inherited]

Variogram plot (EXPERIMENTAL)

Plots the experimental (semi-)variogram (based on the data) as well as the theoretical kriging (semi-)variogram (defined by the correlation function).

Return values

h	Figure handle

Empirical variogram:

- Methods of moments estimator Matheron (Cressie, 1993)
- More robust estimators: Cressie-Hawkins (1980), Genton (1998) In general: use median instead of mean

7.4.3.23 function predict (var *points*)

Predict the mean and/or variance for one or more points x.

Parameters

points	Matrix of input points to be predicted

Return values

values	predicted output values
sigma2	predicted variance of the output values (optional)

Precondition

The kriging object should be fitted using a dataset (`fit`)

7.4.3.24 function predict_derivatives (var *point*)

Predict the derivatives of the mean and/or variance for a points x.

NOTE:

Parameters

point	input point to calculate the derivative of

Return values

dvalues	Derivative w.r.t. the output
sigma2	Derivative w.r.t. the output variance (optional)

- limited to one point at a time (x is a row vector)

7.4.3.25 function predict_limit (var *points*) [inherited]

Limit predictor of kriging (EXPERIMENTAL)

Parameters

points	Matrix of input points to be predicted

Return values

values	predicted output values
sigma2	predicted variance of the output values (optional)

7.4.3.26 function pseudoLikelihood (var *dpsi,* var *dsigma2*) [inherited]

Leave-one-out predictive log probability (pseudo-likelihood)

Calculates the leave-one-out predictive log probability.

Parameters

dpsi	cell array of derivative matrices (optional; for internal use only)

Return values

out	score
dout	Derivatives w.r.t. hyperparameters

Papers:

- "Predictive Approaches for Choosing Hyperparameters in Gaussian Process", S. Sundararajan, S.S. Keerthu, 1999

7.4.3.27 function rcValues ( ) [inherited]

Quantifies magnification of noise (lower is better)

Robustness-criterion (In theory useful only for ordinary kriging).

Return values

rc	robustness-criterion

Returns a 2xn matrix:

- the first row contains the absolute robustness
- the second row contains the relative robustness Papers:
- "Kriging models that are robust w.r.t. simulation errors" A.Y.D. Siem, D. den Hertog (tech report)

7.4.3.28 function regressionFunction (var *options*) [inherited]

Returns the regression function.

Example: [regressionFcn expression terms] = regressionFunction(this, struct('latex', true, 'precision', '%.5g'))

The expression is based on the scaled data

Parameters

options	Options struct

Return values

regressionFcn	Degree matrix representing the regression function
expression	Symbolic expression
terms	Cell array of the individual terms

7.4.3.29 function regressionMatrix (var *points*) [protected], [inherited]

Constructs regression matrix.

Regression matrix (model matrix, Vandermonde matrix, ...) for a set of points Based on this.regressionFcn.

Parameters

points	input point matrix (optional)

Return values

F	Model matrix
dF	Derivative of model matrix w.r.t. the hyperparameters OR the input points

7.4.3.30 function setData (var *samples,* var *values*) [protected]

Sets samples and values matrix.

Scales samples and values, passing the scaled dataset to the underlying base method.

Parameters

samples	input sample matrix
values	output value matrix

7.4.3.31 function setOption (var *key,* var *value*) [inline], [inherited]

Sets a value in the options structure.

Parameters

key	option name
value	option value

7.4.3.32 function tuneParameters (var *F*) [protected], [inherited]

Hyperparameter optimization.

Setups and invokes the optimizer.

Parameters

F	model matrix

Return values

optimHp	optimized hyperparameters
perf	Performance score (likelihood score)

7.4.3.33 function updateModel (var *F*, var *hp*) [protected], [inherited]

Constructs model.

Full update of the model (regression + correlation part)

Parameters

F	model matrix
hp	new hyperparameters

7.4.3.34 function updateRegression (var *F*, var *hp*) [protected], [inherited]

Constructs regression part.

Updates regression part of the model.

Parameters

F	model matrix
hp	new hyperparameters

Return values

err	error string (if any)

Todo Rho is only used by co-kriging, can we abstract this somehow ?

7.4.3.35 function updateStochasticProcess (var *hp*) [protected], [inherited]

Constructs correlation part.

Updates correlation part of the model.

Parameters

hp	hyperparameters

Return values

err	error string (if any)
dpsi	Derivative of correlation matrix w.r.t. the hyperparameters

7.4.4 Member Data Documentation

7.4.4.1 var alpha [protected], [inherited]

Regression coefficients.

7.4.4.2 var C [protected], [inherited]

Choleski Decomposition of extrinsic + intrinsic matrices.

7.4.4.3 var C_reinterp [protected], [inherited]

Reinterpolation version of C.

7.4.4.4 var correlationFcn [protected],[inherited]

string -> function handle

7.4.4.5 var dist [protected],[inherited]

sample inter-distance

7.4.4.6 var distIdxPsi [protected],[inherited]

indexing needed to calculate psi from D

7.4.4.7 var Ft [protected],[inherited]

decorrelated model matrix

7.4.4.8 var Ft_reinterp [protected],[inherited]

Reinterpolation version of `Ft`.

7.4.4.9 var gamma [protected],[inherited]

'Correlation part' coefficients

7.4.4.10 const var HP [inherited]

index of the correlation function parameters

7.4.4.11 var hyperparameters [protected],[inherited]

correlation parameters

7.4.4.12 var hyperparameters0 [protected],[inherited]

initial hp values OR the real ones (if optimization is done outside the class)

7.4.4.13 const var LAMBDA [inherited]

index of the lambda parameter

7.4.4.14 var optimIdx [protected],[inherited]

logical indices to parameters that are optimized

7.4.4.15 var optimNrParameters [protected],[inherited]

number of optimization parameter (vector; one entry per type of parameter)

7.4.4.16 var options [protected],[inherited]

7.4.4.17 var R [protected],[inherited]

from QR decomposition of regression part

7.4.4.18 var R_reinterp [protected],[inherited]

Reinterpolation version of `R`.

7.4.4.19 var regressionFcn [protected],[inherited]

degrees matrix (strings are converted)

7.4.4.20 const var RHO [inherited]

index of the rho parameter

7.4.4.21 var rho [protected],[inherited]

7.4.4.22 var Sigma [protected],[inherited]

intrinsic covariance matrix (amount of regression of stochastic part)

7.4.4.23 const var SIGMA2 [inherited]

index of the sigma2 parameter

7.4.4.24 var sigma2 [protected],[inherited]

process variance of the GP (extrinsic variance)

7.4.4.25 var sigma2_reinterp [protected],[inherited]

Reinterpolation version of `sigma2`.

Note

reinterpolation: might be nicer to just construct and keep a sub-GP... takes more space, some calculations are done twice but performance shouldn't take a very big hit...

7.4.4.26 var tau2 [protected],[inherited]

intrinsic variance

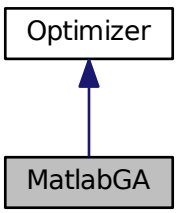
The documentation for this class was generated from the following files:

- /home/ilm/projecten/meta/scripts/ooDACE/@Kriging/Kriging.m
- /home/ilm/projecten/meta/scripts/ooDACE/@Kriging/cvpe.m
- /home/ilm/projecten/meta/scripts/ooDACE/@Kriging/getExpression.m
- /home/ilm/projecten/meta/scripts/ooDACE/@Kriging/predict.m
- /home/ilm/projecten/meta/scripts/ooDACE/@Kriging/predict_derivatives.m
- /home/ilm/projecten/meta/scripts/ooDACE/@Kriging/setData.m

7.5 MatlabGA Class Reference

Wrapper around the matlab optimizers.

Inheritance diagram for MatlabGA:



Public Member Functions

- function MatlabGA (var varargin)

  Class constructor.
- function optimize (var arg)

  This function optimizes the given function handle.
- function getPopulationSize ()

  Get the number of individuals.
- function setInputConstraints (var con)

  Sets input constraints.
- function getBounds ()

  Returns bounds for optimizers that need it.
- function setBounds (var LB, var UB)

  Sets bounds for optimizers that need it.
- function getInitialPopulation ()

  Gets the starting positions for the search.
- function setInitialPopulation (var pop)

  Sets the starting positions for the search.
- function getInputDimension ()

  Returns the number of input variables.
- function getOutputDimension ()

  Returns the number of cost functions.
- function setDimensions (var inDim, var outDim)

  Sets the number of input and output dimensions.
- function setHint (var key, var value)

  Gives a hint to the optimizer.
- function getHint (var key)

  Gets a hint to the optimizer.
- function setState (var state)

  Sets some extra information.
- function getState ()

  Gets some extra information.

7.5.1 Detailed Description

Wrapper around the matlab optimizers.

7.5.2 Constructor & Destructor Documentation

7.5.2.1 function MatlabGA (var *varargin*) [inline]

Class constructor.

Returns

instance of the class

7.5.3 Member Function Documentation

7.5.3.1 function getBounds ( ) [inherited]

Returns bounds for optimizers that need it.

Only needed for optimization methods that support it.

Return values

LB	lower bound
UB	upper bound

7.5.3.2 function getHint (var *key*) [inherited]

Gets a hint to the optimizer.

Returns a particular hint setting.

Parameters

key	property name

Return values

value	property value

7.5.3.3 function getInitialPopulation ( ) [inherited]

Gets the starting positions for the search.

pop may be a matrix for population-based, multi-start, etc. methods

Return values

startx	matrix of initial values

7.5.3.4 function getInputDimension ( ) [inherited]

Returns the number of input variables.

Return values

nvars	Number of input variables

7.5.3.5 function getOutputDimension ( ) [inherited]

Returns the number of cost functions.

Returns the number of output variables.

Return values

nobjectives	Number of cost functions

7.5.3.6 function getPopulationSize ( )

Get the number of individuals.

7.5.3.7 function getState ( ) [inline],[inherited]

Gets some extra information.

Return values

state	structure

7.5.3.8 function optimize (var *arg*)

This function optimizes the given function handle.

7.5.3.9 function setBounds (var *LB*, var *UB*) [inherited]

Sets bounds for optimizers that need it.

Only needed for optimization methods that support it.

Parameters

LB	lower bound
UB	upper bound

7.5.3.10 function setDimensions (var *inDim*, var *outDim*) [inherited]

Sets the number of input and output dimensions.

Includes some input checking to ensure that the bounds and the initial population are still correct.

Parameters

inDim	Number of input variables
outDim	Number of cost functions

If not, these variables are reset to their default values without warning!

7.5.3.11 function setHint (var *key*, var *value*) [inherited]

Gives a hint to the optimizer.

Sets a hint that may or not be honored by the optimizer (depends on the type...).

Parameters

key	property name
value	property value

Only supports 'maxTime', time atm.

7.5.3.12 function setInitialPopulation (var *pop*) [inherited]

Sets the starting positions for the search.

Sets the initial population.

Parameters

pop	matrix of initial values

7.5.3.13 function setInputConstraints (var *con*)

Sets input constraints.

7.5.3.14 function setState (var *state*) [inline],[inherited]

Sets some extra information.

Parameters

state	structure

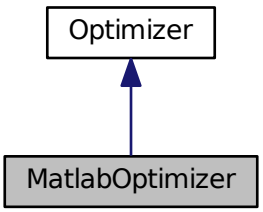
The documentation for this class was generated from the following files:

- /home/ilm/projecten/meta/scripts/ooDACE/optimizers/@MatlabGA/MatlabGA.m
- /home/ilm/projecten/meta/scripts/ooDACE/optimizers/@MatlabGA/getPopulationSize.m
- /home/ilm/projecten/meta/scripts/ooDACE/optimizers/@MatlabGA/optimize.m
- /home/ilm/projecten/meta/scripts/ooDACE/optimizers/@MatlabGA/setInputConstraints.m

7.6 MatlabOptimizer Class Reference

Wrapper around the matlab optimizers.

Inheritance diagram for MatlabOptimizer:



Public Member Functions

- function MatlabOptimizer (var varargin)

  Creates an MatlabOptimizer object.
- function optimize (var arg)

  The hint 'maxTime' is supported.
- function setInputConstraints (var con)

  Supports non-linear and linear inequality constraints.
- function getBounds ()

  Returns bounds for optimizers that need it.
- function setBounds (var LB, var UB)

  Sets bounds for optimizers that need it.
- function getInitialPopulation ()

  Gets the starting positions for the search.
- function setInitialPopulation (var pop)

  Sets the starting positions for the search.
- function getInputDimension ()

  Returns the number of input variables.
- function getOutputDimension ()

Returns the number of cost functions.

- function setDimensions (var inDim, var outDim)

  Sets the number of input and output dimensions.
- function setHint (var key, var value)

  Gives a hint to the optimizer.
- function getHint (var key)

  Gets a hint to the optimizer.
- function setState (var state)

  Sets some extra information.
- function getState ()

  Gets some extra information.
- function getPopulationSize ()

  Get the number of individuals in the population.

7.6.1 Detailed Description

Wrapper around the matlab optimizers.

The matlab Optimiation toolbox is required. If no bounds are set 'fmincon' will be used, else 'fminunc'.

7.6.2 Constructor & Destructor Documentation

7.6.2.1 function MatlabOptimizer (var *varargin*) [inline]

Creates an MatlabOptimizer object.

Takes the same option as the base class + an options structure (see optimset)

Parameters

nvars	Number of dimensions
nobjectives	Number of cost functions
options	Option structure

Returns

instance of the Optimizer class

7.6.3 Member Function Documentation

7.6.3.1 function getBounds ( ) [inherited]

Returns bounds for optimizers that need it.

Only needed for optimization methods that support it.

Return values

LB	lower bound
UB	upper bound

7.6.3.2 function getHint (var *key*) [inherited]

Gets a hint to the optimizer.

Returns a particular hint setting.

Parameters

key	property name

Return values

value	property value

7.6.3.3 function getInitialPopulation ( ) [inherited]

Gets the starting positions for the search.

pop may be a matrix for population-based, multi-start, etc. methods

Return values

startx	matrix of initial values

7.6.3.4 function getInputDimension ( ) [inherited]

Returns the number of input variables.

Return values

nvars	Number of input variables

7.6.3.5 function getOutputDimension ( ) [inherited]

Returns the number of cost functions.

Returns the number of output variables.

Return values

nobjectives	Number of cost functions

7.6.3.6 function getPopulationSize ( ) [inherited]

Get the number of individuals in the population.

The base method assumes only 1 individual.

Return values

size	Population size

Population-based optimization methods should override this

7.6.3.7 function getState ( ) [inline], [inherited]

Gets some extra information.

Return values

state	structure

7.6.3.8 function optimize (var *arg*)

The hint 'maxTime' is supported.

7.6.3.9 function setBounds (var *LB*, var *UB*) [inherited]

Sets bounds for optimizers that need it.

Only needed for optimization methods that support it.

Parameters

LB	lower bound
UB	upper bound

7.6.3.10 function setDimensions (var *inDim*, var *outDim*) [inherited]

Sets the number of input and output dimensions.

Includes some input checking to ensure that the bounds and the initial population are still correct.

Parameters

inDim	Number of input variables
outDim	Number of cost functions

If not, these variables are reset to their default values without warning!

7.6.3.11 function setHint (var *key*, var *value*) [inherited]

Gives a hint to the optimizer.

Sets a hint that may or not be honored by the optimizer (depends on the type...).

Parameters

key	property name
value	property value

Only supports 'maxTime', time atm.

7.6.3.12 function setInitialPopulation (var *pop*) [inherited]

Sets the starting positions for the search.

Sets the initial population.

Parameters

pop	matrix of initial values

7.6.3.13 function setInputConstraints (var *con*)

Supports non-linear and linear inequality constraints.

7.6.3.14 function setState (var *state*) [inline], [inherited]

Sets some extra information.

Parameters

state	structure

The documentation for this class was generated from the following files:

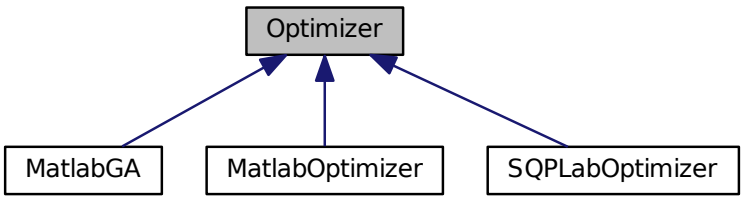
- /home/ilm/projecten/meta/scripts/ooDACE/optimizers/@MatlabOptimizer/MatlabOptimizer.m

- /home/ilm/projecten/meta/scripts/ooDACE/optimizers/@MatlabOptimizer/optimize.m
- /home/ilm/projecten/meta/scripts/ooDACE/optimizers/@MatlabOptimizer/setInputConstraints.m

7.7 Optimizer Class Reference

Abstract base class for an optimizer.

Inheritance diagram for Optimizer:



Public Member Functions

- function Optimizer (var varargin)
  Creates an Optimizer object, not to be called directly.
- function getBounds ()
  Returns bounds for optimizers that need it.
- function setBounds (var LB, var UB)
  Sets bounds for optimizers that need it.
- function getInitialPopulation ()
  Gets the starting positions for the search.
- function setInitialPopulation (var pop)
  Sets the starting positions for the search.
- function getInputDimension ()
  Returns the number of input variables.
- function getOutputDimension ()
  Returns the number of cost functions.
- function setDimensions (var inDim, var outDim)
  Sets the number of input and output dimensions.
- function setHint (var key, var value)
  Gives a hint to the optimizer.
- function getHint (var key)
  Gets a hint to the optimizer.
- function setState (var state)
  Sets some extra information.
- function getState ()
  Gets some extra information.
- function getPopulationSize ()
  Get the number of individuals in the population.
- function setInputConstraints (var con)
  Sets input constraints.
- function optimize (var arg)
  This function optimizes the given function handle,.

7.7.1 Detailed Description

Abstract base class for an optimizer.

Optimization methods are to be derived from this class. It provides a logger object for the derived classes and instantiates all constraints

7.7.2 Constructor & Destructor Documentation

7.7.2.1 function Optimizer (var *varargin*) [inline]

Creates an Optimizer object, not to be called directly.

The constructor of the derived class should be called.

Parameters

nvars	Number of dimensions
nobjectives	Number of cost functions

Returns

instance of the Optimizer class

7.7.3 Member Function Documentation

7.7.3.1 function getBounds ( )

Returns bounds for optimizers that need it.

Only needed for optimization methods that support it.

Return values

LB	lower bound
UB	upper bound

7.7.3.2 function getHint (var *key*)

Gets a hint to the optimizer.

Returns a particular hint setting.

Parameters

key	property name

Return values

value	property value

7.7.3.3 function getInitialPopulation ( )

Gets the starting positions for the search.

pop may be a matrix for population-based, multi-start, etc. methods

Return values

startx	matrix of initial values

7.7.3.4 function getInputDimension ( )

Returns the number of input variables.

Return values

nvars	Number of input variables

7.7.3.5 function getOutputDimension ( )

Returns the number of cost functions.

Returns the number of output variables.

Return values

nobjectives	Number of cost functions

7.7.3.6 function getPopulationSize ( )

Get the number of individuals in the population.

The base method assumes only 1 individual.

Return values

size	Population size

Population-based optimization methods should override this

7.7.3.7 function getState ( ) [inline]

Gets some extra information.

Return values

state	structure

7.7.3.8 function optimize (var *arg*)

This function optimizes the given function handle,.

Dummy function. Subclasses should implement this function.

subject to constraints

Parameters

arg	function handle

Return values

x	optimal input point(s)
fval	optimal function value(s)

7.7.3.9 function setBounds (var *LB,* var *UB*)

Sets bounds for optimizers that need it.

Only needed for optimization methods that support it.

Parameters

LB	lower bound
UB	upper bound

7.7.3.10 function setDimensions (var *inDim,* var *outDim*)

Sets the number of input and output dimensions.

Includes some input checking to ensure that the bounds and the initial population are still correct.

Parameters

inDim	Number of input variables
outDim	Number of cost functions

If not, these variables are reset to their default values without warning!

7.7.3.11 function setHint (var *key,* var *value*)

Gives a hint to the optimizer.

Sets a hint that may or not be honored by the optimizer (depends on the type...).

Parameters

key	property name
value	property value

Only supports 'maxTime', time atm.

7.7.3.12 function setInitialPopulation (var *pop*)

Sets the starting positions for the search.

Sets the initial population.

Parameters

pop	matrix of initial values

7.7.3.13 function setInputConstraints (var *con*)

Sets input constraints.

By default the optimizer doesn't support constraints (gives error).

Parameters

con	constraint objects (cell array)

Derived classes should override this method if they do support it.

7.7.3.14 function setState (var *state*) [inline]

Sets some extra information.

Parameters

state	structure

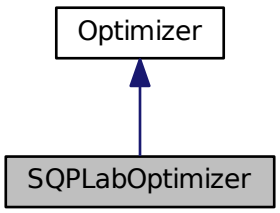
The documentation for this class was generated from the following files:

- /home/ilm/projecten/meta/scripts/ooDACE/optimizers/@Optimizer/Optimizer.m
- /home/ilm/projecten/meta/scripts/ooDACE/optimizers/@Optimizer/getBounds.m
- /home/ilm/projecten/meta/scripts/ooDACE/optimizers/@Optimizer/getHint.m
- /home/ilm/projecten/meta/scripts/ooDACE/optimizers/@Optimizer/getInitialPopulation.m
- /home/ilm/projecten/meta/scripts/ooDACE/optimizers/@Optimizer/getInputDimension.m
- /home/ilm/projecten/meta/scripts/ooDACE/optimizers/@Optimizer/getOutputDimension.m
- /home/ilm/projecten/meta/scripts/ooDACE/optimizers/@Optimizer/getPopulationSize.m
- /home/ilm/projecten/meta/scripts/ooDACE/optimizers/@Optimizer/optimize.m
- /home/ilm/projecten/meta/scripts/ooDACE/optimizers/@Optimizer/setBounds.m
- /home/ilm/projecten/meta/scripts/ooDACE/optimizers/@Optimizer/setDimensions.m
- /home/ilm/projecten/meta/scripts/ooDACE/optimizers/@Optimizer/setHint.m
- /home/ilm/projecten/meta/scripts/ooDACE/optimizers/@Optimizer/setInitialPopulation.m
- /home/ilm/projecten/meta/scripts/ooDACE/optimizers/@Optimizer/setInputConstraints.m

7.8 SQPLabOptimizer Class Reference

Wrapper around the SQPLab optimization package.

Inheritance diagram for SQPLabOptimizer:



Public Member Functions

- function SQPLabOptimizer (var varargin)

  Class constructor.
- function optimize (var arg)

  This function optimizes the given function handle.
- function getBounds ()

  Returns bounds for optimizers that need it.
- function setBounds (var LB, var UB)

  Sets bounds for optimizers that need it.
- function getInitialPopulation ()

  Gets the starting positions for the search.
- function setInitialPopulation (var pop)

  Sets the starting positions for the search.

- function getInputDimension ()

  Returns the number of input variables.
- function getOutputDimension ()

  Returns the number of cost functions.
- function setDimensions (var inDim, var outDim)

  Sets the number of input and output dimensions.
- function setHint (var key, var value)

  Gives a hint to the optimizer.
- function getHint (var key)

  Gets a hint to the optimizer.
- function setState (var state)

  Sets some extra information.
- function getState ()

  Gets some extra information.
- function getPopulationSize ()

  Get the number of individuals in the population.
- function setInputConstraints (var con)

  Sets input constraints.

7.8.1 Detailed Description

Wrapper around the SQPLab optimization package.

7.8.2 Constructor & Destructor Documentation

7.8.2.1 function SQPLabOptimizer (var *varargin*) [inline]

Class constructor.

Returns

instance of the class

7.8.3 Member Function Documentation

7.8.3.1 function getBounds ( ) [inherited]

Returns bounds for optimizers that need it.

Only needed for optimization methods that support it.

Return values

LB	lower bound
UB	upper bound

7.8.3.2 function getHint (var *key*) [inherited]

Gets a hint to the optimizer.

Returns a particular hint setting.

Parameters

key	property name

Return values

value	property value

7.8.3.3 function getInitialPopulation ( ) [inherited]

Gets the starting positions for the search.

pop may be a matrix for population-based, multi-start, etc. methods

Return values

startx	matrix of initial values

7.8.3.4 function getInputDimension ( ) [inherited]

Returns the number of input variables.

Return values

nvars	Number of input variables

7.8.3.5 function getOutputDimension ( ) [inherited]

Returns the number of cost functions.

Returns the number of output variables.

Return values

nobjectives	Number of cost functions

7.8.3.6 function getPopulationSize ( ) [inherited]

Get the number of individuals in the population.

The base method assumes only 1 individual.

Return values

size	Population size

Population-based optimization methods should override this

7.8.3.7 function getState ( ) [inline], [inherited]

Gets some extra information.

Return values

state	structure

7.8.3.8 function optimize (var *arg*)

This function optimizes the given function handle.

7.8.3.9 function setBounds (var *LB*, var *UB*) [inherited]

Sets bounds for optimizers that need it.

Only needed for optimization methods that support it.

Parameters

LB	lower bound
UB	upper bound

7.8.3.10 function setDimensions (var *inDim,* var *outDim*) [inherited]

Sets the number of input and output dimensions.

Includes some input checking to ensure that the bounds and the initial population are still correct.

Parameters

inDim	Number of input variables
outDim	Number of cost functions

If not, these variables are reset to their default values without warning!

7.8.3.11 function setHint (var *key,* var *value*) [inherited]

Gives a hint to the optimizer.

Sets a hint that may or not be honored by the optimizer (depends on the type...).

Parameters

key	property name
value	property value

Only supports 'maxTime', time atm.

7.8.3.12 function setInitialPopulation (var *pop*) [inherited]

Sets the starting positions for the search.

Sets the initial population.

Parameters

pop	matrix of initial values

7.8.3.13 function setInputConstraints (var *con*) [inherited]

Sets input constraints.

By default the optimizer doesn't support constraints (gives error).

Parameters

con	constraint objects (cell array)

Derived classes should override this method if they do support it.

7.8.3.14 function setState (var *state*) [inline], [inherited]

Sets some extra information.

Parameters

state	structure

The documentation for this class was generated from the following files:

- /home/ilm/projecten/meta/scripts/ooDACE/optimizers/@SQPLabOptimizer/SQPLabOptimizer.m
- /home/ilm/projecten/meta/scripts/ooDACE/optimizers/@SQPLabOptimizer/optimize.m

8 File Documentation

8.1 /home/ilm/projecten/meta/scripts/ooDACE/@BasicGaussianProcess/BasicGaussianProcess.m File Reference

Classes

- class BasicGaussianProcess

  A kriging surrogate model (also known as a Gaussian Process)

8.1.1 Detailed Description

Authors

Ivo Couckuyt

Version

1.4 ($Revision$)

Date

$LastChangedDate$
Copyright 2010-2013

This file is part of the ooDACE toolbox and you can redistribute it and/or modify it under the terms of the GNU Affero General Public License version 3 as published by the Free Software Foundation. With the additional provision that a commercial license must be purchased if the ooDACE toolbox is used, modified, or extended in a commercial setting. For details see the included LICENSE.txt file. When referring to the ooDACE toolbox please make reference to the corresponding publications:

- Blind Kriging: Implementation and performance analysis I. Couckuyt, A. Forrester, D. Gorissen, F. De Turck, T. Dhaene, Advances in Engineering Software, Vol. 49, pp. 1-13, July 2012.
- Surrogate-based infill optimization applied to electromagnetic problems I. Couckuyt, F. Declercq, T. Dhaene, H. Rogier, L. Knockaert, International Journal of RF and Microwave Computer-Aided Engineering (RFMiCAE), Special Issue on Advances in Design Optimization of Microwave/RF Circuits and Systems, Vol. 20, No. 5, pp. 492-501, September 2010.

Contact : ivo.couckuyt@ugent.be - http://sumo.intec.ugent.be/?q=ooDACE Signature BasicGaussianProcess

8.2 /home/ilm/projecten/meta/scripts/ooDACE/@BasicGaussianProcess/correlationFunction.m File Reference

8.2.1 Detailed Description

Authors

Ivo Couckuyt

Version

1.4 ($Revision$)

Date

$LastChangedDate$
 Signature [correlationFcn expression] = correlationFunction(this, options)

8.3 /home/ilm/projecten/meta/scripts/ooDACE/@CoKriging/correlationFunction.m File Reference

8.3.1 Detailed Description

Authors

Ivo Couckuyt

Version

1.4 ($Revision$)

Date

$LastChangedDate$
 Signature [correlationFcn expression] = correlationFunction(this, varargin)

8.4 /home/ilm/projecten/meta/scripts/ooDACE/@BasicGaussianProcess/cvpe.m File Reference

8.4.1 Detailed Description

Authors

Ivo Couckuyt

Version

1.4 ($Revision$)

Date

$LastChangedDate$
 Signature out = cvpe(this)

8.5 /home/ilm/projecten/meta/scripts/ooDACE/@Kriging/cvpe.m File Reference

8.5.1 Detailed Description

Authors

Ivo Couckuyt

Version

1.4 ($Revision$)

Date

$LastChangedDate$

- Blind Kriging: Implementation and performance analysis I. Couckuyt, A. Forrester, D. Gorissen, F. De Turck, T. Dhaene, Advances in Engineering Software, Vol. 49, pp. 1-13, July 2012.
- Surrogate-based infill optimization applied to electromagnetic problems I. Couckuyt, F. Declercq, T. Dhaene, H. Rogier, L. Knockaert, International Journal of RF and Microwave Computer-Aided Engineering (RFMiCA-E), Special Issue on Advances in Design Optimization of Microwave/RF Circuits and Systems, Vol. 20, No. 5, pp. 492-501, September 2010.

Contact : ivo.couckuyt@ugent.be - http://sumo.intec.ugent.be/?q=ooDACE Signature out = cvpe(this)

8.6 /home/ilm/projecten/meta/scripts/ooDACE/@BasicGaussianProcess/extrinsicCorrelationMatrix.m File Reference

8.6.1 Detailed Description

Authors

Ivo Couckuyt

Version

1.4 ($Revision$)

Date

$LastChangedDate$
Copyright 2010-2013

This file is part of the ooDACE toolbox and you can redistribute it and/or modify it under the terms of the GNU Affero General Public License version 3 as published by the Free Software Foundation. With the additional provision that a commercial license must be purchased if the ooDACE toolbox is used, modified, or extended in a commercial setting. For details see the included LICENSE.txt file. When referring to the ooDACE toolbox please make reference to the corresponding publications:

- Blind Kriging: Implementation and performance analysis I. Couckuyt, A. Forrester, D. Gorissen, F. De Turck, T. Dhaene, Advances in Engineering Software, Vol. 49, pp. 1-13, July 2012.
- Surrogate-based infill optimization applied to electromagnetic problems I. Couckuyt, F. Declercq, T. Dhaene, H. Rogier, L. Knockaert, International Journal of RF and Microwave Computer-Aided Engineering (RFMiCA-E), Special Issue on Advances in Design Optimization of Microwave/RF Circuits and Systems, Vol. 20, No. 5, pp. 492-501, September 2010.

Contact : ivo.couckuyt@ugent.be - http://sumo.intec.ugent.be/?q=ooDACE Signature [psi dpsi] = extrinsicCorrelationMatrix(this, points1, points2)

8.7 /home/ilm/projecten/meta/scripts/ooDACE/@CoKriging/extrinsicCorrelationMatrix.m File Reference

8.7.1 Detailed Description

Authors

Ivo Couckuyt

Version

1.4 ($Revision$)

Date

$LastChangedDate$
 Signature [psi dpsi] = extrinsicCorrelationMatrix(this, points1, points2)

8.8 /home/ilm/projecten/meta/scripts/ooDACE/@BasicGaussianProcess/fit.m File Reference

8.8.1 Detailed Description

Authors

Ivo Couckuyt

Version

1.4 ($Revision$)

Date

$LastChangedDate$
 Signature this = fit(this, samples, values)

8.9 /home/ilm/projecten/meta/scripts/ooDACE/@BlindKriging/fit.m File Reference

8.9.1 Detailed Description

Authors

Ivo Couckuyt

Version

1.4 ($Revision$)

Date

$LastChangedDate$
Copyright 2010-2013

This file is part of the ooDACE toolbox and you can redistribute it and/or modify it under the terms of the GNU Affero General Public License version 3 as published by the Free Software Foundation. With the additional provision that a commercial license must be purchased if the ooDACE toolbox is used, modified, or extended in a commercial setting. For details see the included LICENSE.txt file. When referring to the ooDACE toolbox please make reference to the corresponding publications:

- Blind Kriging: Implementation and performance analysis I. Couckuyt, A. Forrester, D. Gorissen, F. De Turck, T. Dhaene, Advances in Engineering Software, Vol. 49, pp. 1-13, July 2012.
- Surrogate-based infill optimization applied to electromagnetic problems I. Couckuyt, F. Declercq, T. Dhaene, H. Rogier, L. Knockaert, International Journal of RF and Microwave Computer-Aided Engineering (RFMiCAE), Special Issue on Advances in Design Optimization of Microwave/RF Circuits and Systems, Vol. 20, No. 5, pp. 492-501, September 2010.

Contact : ivo.couckuyt@ugent.be - http://sumo.intec.ugent.be/?q=ooDACE Signature [this IK] = fit(this, samples, values)

8.10 /home/ilm/projecten/meta/scripts/ooDACE/@CoKriging/fit.m File Reference

8.10.1 Detailed Description

Authors

Ivo Couckuyt

Version

1.4 ($Revision$)

Date

$LastChangedDate$
Copyright 2010-2013

This file is part of the ooDACE toolbox and you can redistribute it and/or modify it under the terms of the GNU Affero General Public License version 3 as published by the Free Software Foundation. With the additional provision that a commercial license must be purchased if the ooDACE toolbox is used, modified, or extended in a commercial setting. For details see the included LICENSE.txt file. When referring to the ooDACE toolbox please make reference to the corresponding publications:

- Blind Kriging: Implementation and performance analysis I. Couckuyt, A. Forrester, D. Gorissen, F. De Turck, T. Dhaene, Advances in Engineering Software, Vol. 49, pp. 1-13, July 2012.
- Surrogate-based infill optimization applied to electromagnetic problems I. Couckuyt, F. Declercq, T. Dhaene, H. Rogier, L. Knockaert, International Journal of RF and Microwave Computer-Aided Engineering (RFMiCA-E), Special Issue on Advances in Design Optimization of Microwave/RF Circuits and Systems, Vol. 20, No. 5, pp. 492-501, September 2010.

Contact : ivo.couckuyt@ugent.be - http://sumo.intec.ugent.be/?q=ooDACE Signature this = fit(this, samples, values)

8.11 /home/ilm/projecten/meta/scripts/ooDACE/@BasicGaussianProcess/generateDegrees.m File Reference

8.11.1 Detailed Description

Authors

Ivo Couckuyt

Version

1.4 ($Revision$)

Date

$LastChangedDate$

Contact : ivo.couckuyt@ugent.be - http://sumo.intec.ugent.be/?q=ooDACE Signature [degrees usedIdx] = generateDegrees(this, dj)

8.12 /home/ilm/projecten/meta/scripts/ooDACE/@BasicGaussianProcess/getExpression.m File Reference

8.12.1 Detailed Description

Authors

Ivo Couckuyt

Version

1.4 ($Revision$)

Date

$LastChangedDate$
Copyright 2010-2013

This file is part of the ooDACE toolbox and you can redistribute it and/or modify it under the terms of the GNU Affero General Public License version 3 as published by the Free Software Foundation. With the additional provision that a commercial license must be purchased if the ooDACE toolbox is used, modified, or extended in a commercial setting. For details see the included LICENSE.txt file. When referring to the ooDACE toolbox please make reference to the corresponding publications:

- Blind Kriging: Implementation and performance analysis I. Couckuyt, A. Forrester, D. Gorissen, F. De Turck, T. Dhaene, Advances in Engineering Software, Vol. 49, pp. 1-13, July 2012.
- Surrogate-based infill optimization applied to electromagnetic problems I. Couckuyt, F. Declercq, T. Dhaene, H. Rogier, L. Knockaert, International Journal of RF and Microwave Computer-Aided Engineering (RFMiCAE), Special Issue on Advances in Design Optimization of Microwave/RF Circuits and Systems, Vol. 20, No. 5, pp. 492-501, September 2010.

Contact : ivo.couckuyt@ugent.be - http://sumo.intec.ugent.be/?q=ooDACE Signature expression = getExpression(this, outputIndex)

8.13 /home/ilm/projecten/meta/scripts/ooDACE/@Kriging/getExpression.m File Reference

8.13.1 Detailed Description

Authors

Ivo Couckuyt

Version

1.4 ($Revision$)

Date

$LastChangedDate$
Copyright 2010-2013

This file is part of the ooDACE toolbox and you can redistribute it and/or modify it under the terms of the GNU Affero General Public License version 3 as published by the Free Software Foundation. With the additional provision that a commercial license must be purchased if the ooDACE toolbox is used, modified, or extended in a commercial setting. For details see the included LICENSE.txt file. When referring to the ooDACE toolbox please make reference to the corresponding publications:

- Blind Kriging: Implementation and performance analysis I. Couckuyt, A. Forrester, D. Gorissen, F. De Turck, T. Dhaene, Advances in Engineering Software, Vol. 49, pp. 1-13, July 2012.
- Surrogate-based infill optimization applied to electromagnetic problems I. Couckuyt, F. Declercq, T. Dhaene, H. Rogier, L. Knockaert, International Journal of RF and Microwave Computer-Aided Engineering (RFMiCAE), Special Issue on Advances in Design Optimization of Microwave/RF Circuits and Systems, Vol. 20, No. 5, pp. 492-501, September 2010.

Contact : ivo.couckuyt@ugent.be - http://sumo.intec.ugent.be/?q=ooDACE Signature expression = getExpression(this, outputIndex)

8.14 /home/ilm/projecten/meta/scripts/ooDACE/@BasicGaussianProcess/imse.m File Reference

8.14.1 Detailed Description

Authors

Ivo Couckuyt

Version

1.4 ($Revision$)

Date

$LastChangedDate$
 Signature out = imse(this)

8.15 /home/ilm/projecten/meta/scripts/ooDACE/@BasicGaussianProcess/intrinsicCovarianceMatrix.m File Reference

8.15.1 Detailed Description

Authors

Ivo Couckuyt

Version

1.4 ($Revision$)

Date

$LastChangedDate$

- Blind Kriging: Implementation and performance analysis I. Couckuyt, A. Forrester, D. Gorissen, F. De Turck, T. Dhaene, Advances in Engineering Software, Vol. 49, pp. 1-13, July 2012.
- Surrogate-based infill optimization applied to electromagnetic problems I. Couckuyt, F. Declercq, T. Dhaene, H. Rogier, L. Knockaert, International Journal of RF and Microwave Computer-Aided Engineering (RFMiCA-E), Special Issue on Advances in Design Optimization of Microwave/RF Circuits and Systems, Vol. 20, No. 5, pp. 492-501, September 2010.

Contact : ivo.couckuyt@ugent.be - http://sumo.intec.ugent.be/?q=ooDACE Signature [Sigma dSigma] = intrinsicCovarianceMatrix(this, points1, points2)

8.16 /home/ilm/projecten/meta/scripts/ooDACE/@CoKriging/intrinsicCovarianceMatrix.m File Reference

8.16.1 Detailed Description

Authors

Ivo Couckuyt

Version

1.4 ($Revision$)

Date

$LastChangedDate$

Contact : ivo.couckuyt@ugent.be - http://sumo.intec.ugent.be/?q=ooDACE Signature [Sigma dSigma] = intrinsicCovarianceMatrix(this, points1, points2)

8.17 /home/ilm/projecten/meta/scripts/ooDACE/@BasicGaussianProcess/likelihood.m File Reference

8.17.1 Detailed Description

Authors

Ivo Couckuyt

Version

1.4 ($Revision$)

Date

$LastChangedDate$
Copyright 2010-2013

This file is part of the ooDACE toolbox and you can redistribute it and/or modify it under the terms of the GNU Affero General Public License version 3 as published by the Free Software Foundation. With the additional provision that a commercial license must be purchased if the ooDACE toolbox is used, modified, or extended in a commercial setting. For details see the included LICENSE.txt file. When referring to the ooDACE toolbox please make reference to the corresponding publications:

- Blind Kriging: Implementation and performance analysis I. Couckuyt, A. Forrester, D. Gorissen, F. De Turck, T. Dhaene, Advances in Engineering Software, Vol. 49, pp. 1-13, July 2012.
- Surrogate-based infill optimization applied to electromagnetic problems I. Couckuyt, F. Declercq, T. Dhaene, H. Rogier, L. Knockaert, International Journal of RF and Microwave Computer-Aided Engineering (RFMiCAE), Special Issue on Advances in Design Optimization of Microwave/RF Circuits and Systems, Vol. 20, No. 5, pp. 492-501, September 2010.

Contact : ivo.couckuyt@ugent.be - http://sumo.intec.ugent.be/?q=ooDACE Signature [out dout] = likelihood(this, F, hp)

8.18 /home/ilm/projecten/meta/scripts/ooDACE/@BasicGaussianProcess/marginalLikelihood.m File Reference

8.18.1 Detailed Description

Authors

Ivo Couckuyt

Version

1.4 ($Revision$)

Date

$LastChangedDate$
Copyright 2010-2013

This file is part of the ooDACE toolbox and you can redistribute it and/or modify it under the terms of the GNU Affero General Public License version 3 as published by the Free Software Foundation. With the additional provision that a commercial license must be purchased if the ooDACE toolbox is used, modified, or extended in a commercial setting. For details see the included LICENSE.txt file. When referring to the ooDACE toolbox please make reference to the corresponding publications:

- Blind Kriging: Implementation and performance analysis I. Couckuyt, A. Forrester, D. Gorissen, F. De Turck, T. Dhaene, Advances in Engineering Software, Vol. 49, pp. 1-13, July 2012.
- Surrogate-based infill optimization applied to electromagnetic problems I. Couckuyt, F. Declercq, T. Dhaene, H. Rogier, L. Knockaert, International Journal of RF and Microwave Computer-Aided Engineering (RFMiCAE), Special Issue on Advances in Design Optimization of Microwave/RF Circuits and Systems, Vol. 20, No. 5, pp. 492-501, September 2010.

Contact : ivo.couckuyt@ugent.be - http://sumo.intec.ugent.be/?q=ooDACE Signature [out dout] = marginalLikelihood(this, dpsi, dsigma2)

8.19 /home/ilm/projecten/meta/scripts/ooDACE/@BasicGaussianProcess/mseTestset.m File Reference

8.19.1 Detailed Description

Authors

Ivo Couckuyt

Version

1.4 ($Revision$)

Date

$LastChangedDate$
 Signature out = mseTestset(this, testx, testy)

8.20 /home/ilm/projecten/meta/scripts/ooDACE/@BasicGaussianProcess/plotLikelihood.m File Reference

8.20.1 Detailed Description

Authors

Ivo Couckuyt

Version

1.4 ($Revision$)

Date

$LastChangedDate$

- Blind Kriging: Implementation and performance analysis I. Couckuyt, A. Forrester, D. Gorissen, F. De Turck, T. Dhaene, Advances in Engineering Software, Vol. 49, pp. 1-13, July 2012.
- Surrogate-based infill optimization applied to electromagnetic problems I. Couckuyt, F. Declercq, T. Dhaene, H. Rogier, L. Knockaert, International Journal of RF and Microwave Computer-Aided Engineering (RFMiCA-E), Special Issue on Advances in Design Optimization of Microwave/RF Circuits and Systems, Vol. 20, No. 5, pp. 492-501, September 2010.

Contact : ivo.couckuyt@ugent.be - http://sumo.intec.ugent.be/?q=ooDACE Signature h = plotLikelihood(this, func, param, perf)

8.21 /home/ilm/projecten/meta/scripts/ooDACE/@BasicGaussianProcess/plotVariogram.m File Reference

8.21.1 Detailed Description

Authors

Ivo Couckuyt

Version

1.4 ($Revision$)

Date

$LastChangedDate$

Contact : ivo.couckuyt@ugent.be - http://sumo.intec.ugent.be/?q=ooDACE Signature handle = plotVariogram(this)

8.22 /home/ilm/projecten/meta/scripts/ooDACE/@BasicGaussianProcess/predict.m File Reference

8.22.1 Detailed Description

Authors

Ivo Couckuyt

Version

1.4 ($Revision$)

Date

$LastChangedDate$
Copyright 2010-2013

This file is part of the ooDACE toolbox and you can redistribute it and/or modify it under the terms of the GNU Affero General Public License version 3 as published by the Free Software Foundation. With the additional provision that a commercial license must be purchased if the ooDACE toolbox is used, modified, or extended in a commercial setting. For details see the included LICENSE.txt file. When referring to the ooDACE toolbox please make reference to the corresponding publications:

- Blind Kriging: Implementation and performance analysis I. Couckuyt, A. Forrester, D. Gorissen, F. De Turck, T. Dhaene, Advances in Engineering Software, Vol. 49, pp. 1-13, July 2012.
- Surrogate-based infill optimization applied to electromagnetic problems I. Couckuyt, F. Declercq, T. Dhaene, H. Rogier, L. Knockaert, International Journal of RF and Microwave Computer-Aided Engineering (RFMiCAE), Special Issue on Advances in Design Optimization of Microwave/RF Circuits and Systems, Vol. 20, No. 5, pp. 492-501, September 2010.

Contact : ivo.couckuyt@ugent.be - http://sumo.intec.ugent.be/?q=ooDACE Signature [y sigma2] = predict(this, points)

8.23 /home/ilm/projecten/meta/scripts/ooDACE/@Kriging/predict.m File Reference

8.23.1 Detailed Description

Authors

Ivo Couckuyt

Version

1.4 ($Revision$)

Date

$LastChangedDate$
Copyright 2010-2013

This file is part of the ooDACE toolbox and you can redistribute it and/or modify it under the terms of the GNU Affero General Public License version 3 as published by the Free Software Foundation. With the additional provision that a commercial license must be purchased if the ooDACE toolbox is used, modified, or extended in a commercial setting. For details see the included LICENSE.txt file. When referring to the ooDACE toolbox please make reference to the corresponding publications:

- Blind Kriging: Implementation and performance analysis I. Couckuyt, A. Forrester, D. Gorissen, F. De Turck, T. Dhaene, Advances in Engineering Software, Vol. 49, pp. 1-13, July 2012.
- Surrogate-based infill optimization applied to electromagnetic problems I. Couckuyt, F. Declercq, T. Dhaene, H. Rogier, L. Knockaert, International Journal of RF and Microwave Computer-Aided Engineering (RFMiCAE), Special Issue on Advances in Design Optimization of Microwave/RF Circuits and Systems, Vol. 20, No. 5, pp. 492-501, September 2010.

Contact : ivo.couckuyt@ugent.be - http://sumo.intec.ugent.be/?q=ooDACE Signature [y sigma2] = predict(this, points)

8.24 /home/ilm/projecten/meta/scripts/ooDACE/@BasicGaussianProcess/predict_derivatives.m File Reference

8.24.1 Detailed Description

Authors

Ivo Couckuyt

Version

1.4 ($Revision$)

Date

$LastChangedDate$
 Signature [dy dsigma2] = predict_derivatives(this, point)

8.25 /home/ilm/projecten/meta/scripts/ooDACE/@Kriging/predict_derivatives.m File Reference

8.25.1 Detailed Description

Authors

Ivo Couckuyt

Version

1.4 ($Revision$)

Date

$LastChangedDate$

- Blind Kriging: Implementation and performance analysis I. Couckuyt, A. Forrester, D. Gorissen, F. De Turck, T. Dhaene, Advances in Engineering Software, Vol. 49, pp. 1-13, July 2012.
- Surrogate-based infill optimization applied to electromagnetic problems I. Couckuyt, F. Declercq, T. Dhaene, H. Rogier, L. Knockaert, International Journal of RF and Microwave Computer-Aided Engineering (RFMiCA-E), Special Issue on Advances in Design Optimization of Microwave/RF Circuits and Systems, Vol. 20, No. 5, pp. 492-501, September 2010.

Contact : ivo.couckuyt@ugent.be - http://sumo.intec.ugent.be/?q=ooDACE Signature [dy dsigma2] = predict_derivatives(this, point)

8.26 /home/ilm/projecten/meta/scripts/ooDACE/@BasicGaussianProcess/pseudoLikelihood.m File Reference

8.26.1 Detailed Description

Authors

Ivo Couckuyt

Version

1.4 ($Revision$)

Date

$LastChangedDate$

- Blind Kriging: Implementation and performance analysis I. Couckuyt, A. Forrester, D. Gorissen, F. De Turck, T. Dhaene, Advances in Engineering Software, Vol. 49, pp. 1-13, July 2012.
- Surrogate-based infill optimization applied to electromagnetic problems I. Couckuyt, F. Declercq, T. Dhaene, H. Rogier, L. Knockaert, International Journal of RF and Microwave Computer-Aided Engineering (RFMiCA-E), Special Issue on Advances in Design Optimization of Microwave/RF Circuits and Systems, Vol. 20, No. 5, pp. 492-501, September 2010.

Contact : ivo.couckuyt@ugent.be - http://sumo.intec.ugent.be/?q=ooDACE Signature [out dout] = pseudoLikelihood(this, dpsi, dsigma2)

8.27 /home/ilm/projecten/meta/scripts/ooDACE/@BasicGaussianProcess/rcValues.m File Reference

8.27.1 Detailed Description

Authors

Ivo Couckuyt

Version

1.4 ($Revision$)

Date

$LastChangedDate$
Copyright 2010-2013

This file is part of the ooDACE toolbox and you can redistribute it and/or modify it under the terms of the GNU Affero General Public License version 3 as published by the Free Software Foundation. With the additional provision that a commercial license must be purchased if the ooDACE toolbox is used, modified, or extended in a commercial setting. For details see the included LICENSE.txt file. When referring to the ooDACE toolbox please make reference to the corresponding publications:

- Blind Kriging: Implementation and performance analysis I. Couckuyt, A. Forrester, D. Gorissen, F. De Turck, T. Dhaene, Advances in Engineering Software, Vol. 49, pp. 1-13, July 2012.
- Surrogate-based infill optimization applied to electromagnetic problems I. Couckuyt, F. Declercq, T. Dhaene, H. Rogier, L. Knockaert, International Journal of RF and Microwave Computer-Aided Engineering (RFMiCA-E), Special Issue on Advances in Design Optimization of Microwave/RF Circuits and Systems, Vol. 20, No. 5, pp. 492-501, September 2010.

Contact : ivo.couckuyt@ugent.be - http://sumo.intec.ugent.be/?q=ooDACE Signature rc = rcValues(this)

8.28 /home/ilm/projecten/meta/scripts/ooDACE/@BasicGaussianProcess/regressionFunction.m File Reference

8.28.1 Detailed Description

Authors

Ivo Couckuyt

Version

1.4 ($Revision$)

Date

$LastChangedDate$
Copyright 2010-2013

This file is part of the ooDACE toolbox and you can redistribute it and/or modify it under the terms of the GNU Affero General Public License version 3 as published by the Free Software Foundation. With the additional provision that a commercial license must be purchased if the ooDACE toolbox is used, modified, or extended in a commercial setting. For details see the included LICENSE.txt file. When referring to the ooDACE toolbox please make reference to the corresponding publications:

- Blind Kriging: Implementation and performance analysis I. Couckuyt, A. Forrester, D. Gorissen, F. De Turck, T. Dhaene, Advances in Engineering Software, Vol. 49, pp. 1-13, July 2012.
- Surrogate-based infill optimization applied to electromagnetic problems I. Couckuyt, F. Declercq, T. Dhaene, H. Rogier, L. Knockaert, International Journal of RF and Microwave Computer-Aided Engineering (RFMiCA-E), Special Issue on Advances in Design Optimization of Microwave/RF Circuits and Systems, Vol. 20, No. 5, pp. 492-501, September 2010.

Contact : ivo.couckuyt@ugent.be - http://sumo.intec.ugent.be/?q=ooDACE Signature [regressionFcn expression terms] = regressionFunction(this, options)

8.29 /home/ilm/projecten/meta/scripts/ooDACE/@BlindKriging/regressionFunction.m File Reference

8.29.1 Detailed Description

Authors

Ivo Couckuyt

Version

1.4 ($Revision$)

Date

$LastChangedDate$
Copyright 2010-2013

This file is part of the ooDACE toolbox and you can redistribute it and/or modify it under the terms of the GNU Affero General Public License version 3 as published by the Free Software Foundation. With the additional provision that a commercial license must be purchased if the ooDACE toolbox is used, modified, or extended in a commercial setting. For details see the included LICENSE.txt file. When referring to the ooDACE toolbox please make reference to the corresponding publications:

- Blind Kriging: Implementation and performance analysis I. Couckuyt, A. Forrester, D. Gorissen, F. De Turck, T. Dhaene, Advances in Engineering Software, Vol. 49, pp. 1-13, July 2012.
- Surrogate-based infill optimization applied to electromagnetic problems I. Couckuyt, F. Declercq, T. Dhaene, H. Rogier, L. Knockaert, International Journal of RF and Microwave Computer-Aided Engineering (RFMiCA-E), Special Issue on Advances in Design Optimization of Microwave/RF Circuits and Systems, Vol. 20, No. 5, pp. 492-501, September 2010.

Contact : ivo.couckuyt@ugent.be - http://sumo.intec.ugent.be/?q=ooDACE Signature [regressionFcn expression terms] = regressionFunction(this, options)

8.30 /home/ilm/projecten/meta/scripts/ooDACE/@CoKriging/regressionFunction.m File Reference

8.30.1 Detailed Description

Authors

Ivo Couckuyt

Version

1.4 ($Revision$)

Date

$LastChangedDate$
Copyright 2010-2013

This file is part of the ooDACE toolbox and you can redistribute it and/or modify it under the terms of the GNU Affero General Public License version 3 as published by the Free Software Foundation. With the additional provision that a commercial license must be purchased if the ooDACE toolbox is used, modified, or extended in a commercial setting. For details see the included LICENSE.txt file. When referring to the ooDACE toolbox please make reference to the corresponding publications:

- Blind Kriging: Implementation and performance analysis I. Couckuyt, A. Forrester, D. Gorissen, F. De Turck, T. Dhaene, Advances in Engineering Software, Vol. 49, pp. 1-13, July 2012.
- Surrogate-based infill optimization applied to electromagnetic problems I. Couckuyt, F. Declercq, T. Dhaene, H. Rogier, L. Knockaert, International Journal of RF and Microwave Computer-Aided Engineering (RFMiCA-E), Special Issue on Advances in Design Optimization of Microwave/RF Circuits and Systems, Vol. 20, No. 5, pp. 492-501, September 2010.

Contact : ivo.couckuyt@ugent.be - http://sumo.intec.ugent.be/?q=ooDACE Signature [regressionFcn expression terms] = regressionFunction(this, varargin)

8.31 /home/ilm/projecten/meta/scripts/ooDACE/@BasicGaussianProcess/regressionMatrix.m File Reference

8.31.1 Detailed Description

Authors

Ivo Couckuyt

Version

1.4 ($Revision$)

Date

$LastChangedDate$

Contact : ivo.couckuyt@ugent.be - http://sumo.intec.ugent.be/?q=ooDACE Signature [F dF] = regressionMatrix(this, points)

8.32 /home/ilm/projecten/meta/scripts/ooDACE/@BlindKriging/regressionMatrix.m File Reference

8.32.1 Detailed Description

Authors

Ivo Couckuyt

Version

1.4 ($Revision$)

Date

$LastChangedDate$

8.33 /home/ilm/projecten/meta/scripts/ooDACE/@CoKriging/regressionMatrix.m File Reference

8.33.1 Detailed Description

Authors

Ivo Couckuyt

Version

1.4 ($Revision$)

Date

$LastChangedDate$

8.34 /home/ilm/projecten/meta/scripts/ooDACE/@BasicGaussianProcess/tuneParameters.m File Reference

8.34.1 Detailed Description

Authors

Ivo Couckuyt

Version

1.4 ($Revision$)

Date

$LastChangedDate$
 Signature [this optimHp perf] = tuneParameters(this, F)

8.35 /home/ilm/projecten/meta/scripts/ooDACE/@BasicGaussianProcess/updateModel.m File Reference

8.35.1 Detailed Description

Authors

Ivo Couckuyt

Version

1.4 ($Revision$)

Date

$LastChangedDate$

- Blind Kriging: Implementation and performance analysis I. Couckuyt, A. Forrester, D. Gorissen, F. De Turck, T. Dhaene, Advances in Engineering Software, Vol. 49, pp. 1-13, July 2012.
- Surrogate-based infill optimization applied to electromagnetic problems I. Couckuyt, F. Declercq, T. Dhaene, H. Rogier, L. Knockaert, International Journal of RF and Microwave Computer-Aided Engineering (RFMiCA-E), Special Issue on Advances in Design Optimization of Microwave/RF Circuits and Systems, Vol. 20, No. 5, pp. 492-501, September 2010.

Contact : ivo.couckuyt@ugent.be - http://sumo.intec.ugent.be/?q=ooDACE Signature this = updateModel(this, F, hp)

8.36 /home/ilm/projecten/meta/scripts/ooDACE/@BasicGaussianProcess/updateRegression.m File Reference

8.36.1 Detailed Description

Authors

Ivo Couckuyt

Version

1.4 ($Revision$)

Date

$LastChangedDate$

Contact : ivo.couckuyt@ugent.be - http://sumo.intec.ugent.be/?q=ooDACE Signature [this err dsigma2] = updateRegression(this, F, hp)

8.37 /home/ilm/projecten/meta/scripts/ooDACE/@BasicGaussianProcess/updateStochasticProcess.m File Reference

8.37.1 Detailed Description

Authors

Ivo Couckuyt

Version

1.4 ($Revision$)

Date

$LastChangedDate$

Contact : ivo.couckuyt@ugent.be - http://sumo.intec.ugent.be/?q=ooDACE Signature [this err dpsi] = updateStochasticProcess(this, hp)

8.38 /home/ilm/projecten/meta/scripts/ooDACE/@BlindKriging/BlindKriging.m File Reference

Classes

- class BlindKriging

  A blind kriging surrogate model.

8.38.1 Detailed Description

Authors

Ivo Couckuyt

Version

1.4 ($Revision$)

Date

$LastChangedDate$

- Surrogate-based infill optimization applied to electromagnetic problems I. Couckuyt, F. Declercq, T. Dhaene, H. Rogier, L. Knockaert, International Journal of RF and Microwave Computer-Aided Engineering (RFMiCA-E), Special Issue on Advances in Design Optimization of Microwave/RF Circuits and Systems, Vol. 20, No. 5, pp. 492-501, September 2010.

Contact : ivo.couckuyt@ugent.be - http://sumo.intec.ugent.be/?q=ooDACE Signature Blind-Kriging

8.39 /home/ilm/projecten/meta/scripts/ooDACE/@BlindKriging/polynomialCoding.m File Reference

8.39.1 Detailed Description

Authors

Ivo Couckuyt

Version

1.4 ($Revision$)

Date

$LastChangedDate$
 When referring to the ooDACE toolbox please make reference to the corresponding publications:

- Blind Kriging: Implementation and performance analysis I. Couckuyt, A. Forrester, D. Gorissen, F. De Turck, T. Dhaene, Advances in Engineering Software, Vol. 49, pp. 1-13, July 2012.
- Surrogate-based infill optimization applied to electromagnetic problems I. Couckuyt, F. Declercq, T. Dhaene, H. Rogier, L. Knockaert, International Journal of RF and Microwave Computer-Aided Engineering (RFMiCA-E), Special Issue on Advances in Design Optimization of Microwave/RF Circuits and Systems, Vol. 20, No. 5, pp. 492-501, September 2010.

Contact : ivo.couckuyt@ugent.be - http://sumo.intec.ugent.be/?q=ooDACE Signature [U dU] = polynomialCoding(this, samples, m, k, delta)

8.40 /home/ilm/projecten/meta/scripts/ooDACE/@BlindKriging/posteriorBeta.m File Reference

8.40.1 Detailed Description

Authors

Ivo Couckuyt

Version

1.4 ($Revision$)

Date

$LastChangedDate$
 Signature beta = posteriorBeta(this, R, U)

8.41 /home/ilm/projecten/meta/scripts/ooDACE/@BlindKriging/Rmatrix.m File Reference

8.41.1 Detailed Description

Authors

Ivo Couckuyt

Version

1.4 ($Revision$)

Date

$LastChangedDate$
 Signature out = Rmatrix(this, usedIdx)

8.42 /home/ilm/projecten/meta/scripts/ooDACE/@CoKriging/CoKriging.m File Reference

Classes

- class CoKriging

  A cokriging surrogate model.

8.42.1 Detailed Description

Authors

Ivo Couckuyt

Version

1.4 ($Revision$)

Date

$LastChangedDate$

- Blind Kriging: Implementation and performance analysis I. Couckuyt, A. Forrester, D. Gorissen, F. De Turck, T. Dhaene, Advances in Engineering Software, Vol. 49, pp. 1-13, July 2012.
- Surrogate-based infill optimization applied to electromagnetic problems I. Couckuyt, F. Declercq, T. Dhaene, H. Rogier, L. Knockaert, International Journal of RF and Microwave Computer-Aided Engineering (RFMiCA-E), Special Issue on Advances in Design Optimization of Microwave/RF Circuits and Systems, Vol. 20, No. 5, pp. 492-501, September 2010.

Contact : ivo.couckuyt@ugent.be - http://sumo.intec.ugent.be/?q=ooDACE Signature CoKriging

8.43 /home/ilm/projecten/meta/scripts/ooDACE/@CoKriging/setData.m File Reference

8.43.1 Detailed Description

Authors

Ivo Couckuyt

Version

1.4 ($Revision$)

Date

$LastChangedDate$
Copyright 2010-2013

This file is part of the ooDACE toolbox and you can redistribute it and/or modify it under the terms of the GNU Affero General Public License version 3 as published by the Free Software Foundation. With the additional provision that a commercial license must be purchased if the ooDACE toolbox is used, modified, or extended in a commercial setting. For details see the included LICENSE.txt file. When referring to the ooDACE toolbox please make reference to the corresponding publications:

- Blind Kriging: Implementation and performance analysis I. Couckuyt, A. Forrester, D. Gorissen, F. De Turck, T. Dhaene, Advances in Engineering Software, Vol. 49, pp. 1-13, July 2012.
- Surrogate-based infill optimization applied to electromagnetic problems I. Couckuyt, F. Declercq, T. Dhaene, H. Rogier, L. Knockaert, International Journal of RF and Microwave Computer-Aided Engineering (RFMiCAE), Special Issue on Advances in Design Optimization of Microwave/RF Circuits and Systems, Vol. 20, No. 5, pp. 492-501, September 2010.

Contact : ivo.couckuyt@ugent.be - http://sumo.intec.ugent.be/?q=ooDACE Signature this = setData(this, samples, values)

8.44 /home/ilm/projecten/meta/scripts/ooDACE/@Kriging/setData.m File Reference

8.44.1 Detailed Description

Authors

Ivo Couckuyt

Version

1.4 ($Revision$)

Date

$LastChangedDate$
Copyright 2010-2013

This file is part of the ooDACE toolbox and you can redistribute it and/or modify it under the terms of the GNU Affero General Public License version 3 as published by the Free Software Foundation. With the additional provision that a commercial license must be purchased if the ooDACE toolbox is used, modified, or extended in a commercial setting. For details see the included LICENSE.txt file. When referring to the ooDACE toolbox please make reference to the corresponding publications:

- Blind Kriging: Implementation and performance analysis I. Couckuyt, A. Forrester, D. Gorissen, F. De Turck, T. Dhaene, Advances in Engineering Software, Vol. 49, pp. 1-13, July 2012.
- Surrogate-based infill optimization applied to electromagnetic problems I. Couckuyt, F. Declercq, T. Dhaene, H. Rogier, L. Knockaert, International Journal of RF and Microwave Computer-Aided Engineering (RFMiCAE), Special Issue on Advances in Design Optimization of Microwave/RF Circuits and Systems, Vol. 20, No. 5, pp. 492-501, September 2010.

Contact : ivo.couckuyt@ugent.be - http://sumo.intec.ugent.be/?q=ooDACE Signature this = setData(this, samples, values)

8.45 /home/ilm/projecten/meta/scripts/ooDACE/@Kriging/Kriging.m File Reference

Classes

- class Kriging

  A kriging surrogate model.

8.45.1 Detailed Description

Authors

Ivo Couckuyt

Version

1.4 ($Revision$)

Date

$LastChangedDate$

8.46 /home/ilm/projecten/meta/scripts/ooDACE/basisfunctions/corrcubic.m File Reference

Functions

- function corrcubic (var theta, var d)

  Cubic correlation function.

8.46.1 Detailed Description

Authors

Ivo Couckuyt

Version

1.4 ($Revision$)

Date

$LastChangedDate$
 Signature [corr, dx, dtheta, rho] = corrcubic(theta, d)

8.46.2 Function Documentation

8.46.2.1 function corrcubic (var *theta,* var *d*)

Cubic correlation function.

8.47 /home/ilm/projecten/meta/scripts/ooDACE/basisfunctions/correxp.m File Reference

Functions

- function correxp (var theta, var d)

  Exponential correlation function.

8.47.1 Detailed Description

Authors

Ivo Couckuyt

Version

1.4 ($Revision$)

Date

$LastChangedDate$

- Blind Kriging: Implementation and performance analysis I. Couckuyt, A. Forrester, D. Gorissen, F. De Turck, T. Dhaene, Advances in Engineering Software, Vol. 49, pp. 1-13, July 2012.
- Surrogate-based infill optimization applied to electromagnetic problems I. Couckuyt, F. Declercq, T. Dhaene, H. Rogier, L. Knockaert, International Journal of RF and Microwave Computer-Aided Engineering (RFMiCA-E), Special Issue on Advances in Design Optimization of Microwave/RF Circuits and Systems, Vol. 20, No. 5, pp. 492-501, September 2010.

Contact : ivo.couckuyt@ugent.be - http://sumo.intec.ugent.be/?q=ooDACE Signature [corr, dx, dtheta, rho] = correxp(theta, d)

8.47.2 Function Documentation

8.47.2.1 function correxp (var *theta,* var *d*)

Exponential correlation function.

8.48 /home/ilm/projecten/meta/scripts/ooDACE/basisfunctions/corrgauss.m File Reference

Functions

- function corrgauss (var theta, var d)

  Gaussian correlation function.

8.48.1 Detailed Description

Authors

Ivo Couckuyt

Version

1.4 ($Revision$)

Date

$LastChangedDate$
Copyright 2010-2013

This file is part of the ooDACE toolbox and you can redistribute it and/or modify it under the terms of the GNU Affero General Public License version 3 as published by the Free Software Foundation. With the additional provision that a commercial license must be purchased if the ooDACE toolbox is used, modified, or extended in a commercial setting. For details see the included LICENSE.txt file. When referring to the ooDACE toolbox please make reference to the corresponding publications:

- Blind Kriging: Implementation and performance analysis I. Couckuyt, A. Forrester, D. Gorissen, F. De Turck, T. Dhaene, Advances in Engineering Software, Vol. 49, pp. 1-13, July 2012.
- Surrogate-based infill optimization applied to electromagnetic problems I. Couckuyt, F. Declercq, T. Dhaene, H. Rogier, L. Knockaert, International Journal of RF and Microwave Computer-Aided Engineering (RFMiCA-E), Special Issue on Advances in Design Optimization of Microwave/RF Circuits and Systems, Vol. 20, No. 5, pp. 492-501, September 2010.

Contact : ivo.couckuyt@ugent.be - http://sumo.intec.ugent.be/?q=ooDACE Signature [corr, dx, dtheta, rho] = corrgauss(theta, d)

8.48.2 Function Documentation

8.48.2.1 function corrgauss (var *theta,* var *d*)

Gaussian correlation function.

8.49 /home/ilm/projecten/meta/scripts/ooDACE/basisfunctions/corrgaussp.m File Reference

Functions

- function corrgaussp (var hp, var d)
  Gaussian correlation function.

8.49.1 Detailed Description

Authors

Ivo Couckuyt

Version

1.4 ($Revision$)

Date

$LastChangedDate$
Copyright 2010-2013

This file is part of the ooDACE toolbox and you can redistribute it and/or modify it under the terms of the GNU Affero General Public License version 3 as published by the Free Software Foundation. With the additional provision that a commercial license must be purchased if the ooDACE toolbox is used, modified, or extended in a commercial setting. For details see the included LICENSE.txt file. When referring to the ooDACE toolbox please make reference to the corresponding publications:

- Blind Kriging: Implementation and performance analysis I. Couckuyt, A. Forrester, D. Gorissen, F. De Turck, T. Dhaene, Advances in Engineering Software, Vol. 49, pp. 1-13, July 2012.
- Surrogate-based infill optimization applied to electromagnetic problems I. Couckuyt, F. Declercq, T. Dhaene, H. Rogier, L. Knockaert, International Journal of RF and Microwave Computer-Aided Engineering (RFMiCA-E), Special Issue on Advances in Design Optimization of Microwave/RF Circuits and Systems, Vol. 20, No. 5, pp. 492-501, September 2010.

Contact : ivo.couckuyt@ugent.be - http://sumo.intec.ugent.be/?q=ooDACE Signature [corr, dx, dhp rho] = corrgaussp(hp, d)

8.49.2 Function Documentation

8.49.2.1 function corrgaussp (var *hp,* var *d*)

Gaussian correlation function.

where p is also a hyperparameter

8.50 /home/ilm/projecten/meta/scripts/ooDACE/basisfunctions/corrlin.m File Reference

Functions

- function corrlin (var theta, var d)

  Linear correlation function,.

8.50.1 Detailed Description

Authors

Ivo Couckuyt

Version

1.4 ($Revision$)

Date

$LastChangedDate$
Copyright 2010-2013

This file is part of the ooDACE toolbox and you can redistribute it and/or modify it under the terms of the GNU Affero General Public License version 3 as published by the Free Software Foundation. With the additional provision that a commercial license must be purchased if the ooDACE toolbox is used, modified, or extended in a commercial setting. For details see the included LICENSE.txt file. When referring to the ooDACE toolbox please make reference to the corresponding publications:

- Blind Kriging: Implementation and performance analysis I. Couckuyt, A. Forrester, D. Gorissen, F. De Turck, T. Dhaene, Advances in Engineering Software, Vol. 49, pp. 1-13, July 2012.
- Surrogate-based infill optimization applied to electromagnetic problems I. Couckuyt, F. Declercq, T. Dhaene, H. Rogier, L. Knockaert, International Journal of RF and Microwave Computer-Aided Engineering (RFMiCA-E), Special Issue on Advances in Design Optimization of Microwave/RF Circuits and Systems, Vol. 20, No. 5, pp. 492-501, September 2010.

Contact : ivo.couckuyt@ugent.be - http://sumo.intec.ugent.be/?q=ooDACE Signature [corr, dx, dtheta, rho] = corrlin(theta, d)

8.50.2 Function Documentation

8.50.2.1 function corrlin (var *theta,* var *d*)

Linear correlation function,.

8.51 /home/ilm/projecten/meta/scripts/ooDACE/basisfunctions/corrmatern32.m File Reference

Functions

- function corrmatern32 (var hp, var d)

  Class of Matern covariance functions (nu = {1/2 3/2 5/2}.

8.51.1 Detailed Description

Authors

Ivo Couckuyt

Version

1.4 ($Revision$)

Date

$LastChangedDate$

Contact : ivo.couckuyt@ugent.be - http://sumo.intec.ugent.be/?q=ooDACE Signature [corr, dx, dhp, rho] = corrmatern32(hp, d)

8.51.2 Function Documentation

8.51.2.1 function corrmatern32 (var *hp,* var *d*)

Class of Matern covariance functions (nu = {1/2 3/2 5/2}.

Implementation based on the book of Rasmussen et al.

- "Gaussian Processes for Machine Learning" (Chapter 4), C. E. Rasmussen and C. K. I. Williams, MIT Press, 2006

The correlation function is: corr = f(sqrt(nud)∗dist2) ∗ exp(-sqrt(nud)∗dist2) with nud = 2.∗nu

8.52 /home/ilm/projecten/meta/scripts/ooDACE/basisfunctions/corrmatern32_iso.m File Reference

Functions

- function corrmatern32_iso (var hp, var d)

  Class of Matern covariance functions (nu = {1/2 3/2 5/2}.

8.52.1 Detailed Description

Authors

Ivo Couckuyt

Version

1.4 ($Revision$)

Date

$LastChangedDate$

Contact : ivo.couckuyt@ugent.be - http://sumo.intec.ugent.be/?q=ooDACE Signature [corr, dx, dhp, rho] = corrmatern32_iso(hp, d)

8.52.2 Function Documentation

8.52.2.1 function corrmatern32_iso (var *hp,* var *d*)

Class of Matern covariance functions (nu = {1/2 3/2 5/2}.

Implementation based on the book of Rasmussen et al.

- "Gaussian Processes for Machine Learning" (Chapter 4), C. E. Rasmussen and C. K. I. Williams, MIT Press, 2006

The correlation function is: corr = f(sqrt(nud)∗dist2) ∗ exp(-sqrt(nud)∗dist2) with nud = 2.∗nu

8.53 /home/ilm/projecten/meta/scripts/ooDACE/basisfunctions/corrmatern32_variance.m File Reference

Functions

- function corrmatern32_variance (var hp, var d)

  Class of Matern covariance functions (nu = {1/2 3/2 5/2}.

8.53.1 Detailed Description

Authors

Ivo Couckuyt

Version

1.4 ($Revision$)

Date

$LastChangedDate$

Contact : ivo.couckuyt@ugent.be - http://sumo.intec.ugent.be/?q=ooDACE Signature [corr, dx, dhp, rho] = corrmatern32_variance(hp, d)

8.53.2 Function Documentation

8.53.2.1 function corrmatern32_variance (var *hp,* var *d*)

Class of Matern covariance functions (nu = {1/2 3/2 5/2}.

Implementation based on the book of Rasmussen et al.

- "Gaussian Processes for Machine Learning" (Chapter 4), C. E. Rasmussen and C. K. I. Williams, MIT Press, 2006

The correlation function is: corr = f(sqrt(nud)∗dist2) ∗ exp(-sqrt(nud)∗dist2) with nud = 2.∗nu

8.54 /home/ilm/projecten/meta/scripts/ooDACE/basisfunctions/corrmatern32iso.m File Reference

Functions

- function corrmatern32iso (var hp, var d)

  Class of Matern covariance functions (nu = {1/2 3/2 5/2}.

8.54.1 Detailed Description

Authors

Ivo Couckuyt

Version

1.4 ($Revision$)

Date

$LastChangedDate$
 Signature [corr, dx, dhp, rho] = corrmatern32iso(hp, d)

8.54.2 Function Documentation

8.54.2.1 function corrmatern32iso (var *hp,* var *d*)

Class of Matern covariance functions (nu = {1/2 3/2 5/2}.

Implementation based on the book of Rasmussen et al.

- "Gaussian Processes for Machine Learning" (Chapter 4), C. E. Rasmussen and C. K. I. Williams, MIT Press, 2006

The correlation function is: corr = f(sqrt(nud)∗dist2) ∗ exp(-sqrt(nud)∗dist2) with nud = 2.∗nu

8.55 /home/ilm/projecten/meta/scripts/ooDACE/basisfunctions/corrmatern52.m File Reference

Functions

- function corrmatern52 (var hp, var d)

  Class of Matern covariance functions (nu = {1/2 3/2 5/2}.

8.55.1 Detailed Description

Authors

Ivo Couckuyt

Version

1.4 ($Revision$)

Date

$LastChangedDate$

Contact : ivo.couckuyt@ugent.be - http://sumo.intec.ugent.be/?q=ooDACE Signature [corr, dx, dhp, rho] = corrmatern52(hp, d)

8.55.2 Function Documentation

8.55.2.1 function corrmatern52 (var *hp,* var *d*)

Class of Matern covariance functions (nu = {1/2 3/2 5/2}.

Implementation based on the book of Rasmussen et al.

- "Gaussian Processes for Machine Learning" (Chapter 4), C. E. Rasmussen and C. K. I. Williams, MIT Press, 2006

The correlation function is: corr = f(sqrt(nud)∗dist2) ∗ exp(-sqrt(nud)∗dist2) with nud = 2.∗nu

8.56 /home/ilm/projecten/meta/scripts/ooDACE/basisfunctions/corrspherical.m File Reference

Functions

- function corrspherical (var theta, var d)

  Spherical correlation function.

8.56.1 Detailed Description

Authors

Ivo Couckuyt

Version

1.4 ($Revision$)

Date

$LastChangedDate$
Copyright 2010-2013

This file is part of the ooDACE toolbox and you can redistribute it and/or modify it under the terms of the GNU Affero General Public License version 3 as published by the Free Software Foundation. With the additional provision that a commercial license must be purchased if the ooDACE toolbox is used, modified, or extended in a commercial setting. For details see the included LICENSE.txt file. When referring to the ooDACE toolbox please make reference to the corresponding publications:

- Blind Kriging: Implementation and performance analysis I. Couckuyt, A. Forrester, D. Gorissen, F. De Turck, T. Dhaene, Advances in Engineering Software, Vol. 49, pp. 1-13, July 2012.
- Surrogate-based infill optimization applied to electromagnetic problems I. Couckuyt, F. Declercq, T. Dhaene, H. Rogier, L. Knockaert, International Journal of RF and Microwave Computer-Aided Engineering (RFMiCAE), Special Issue on Advances in Design Optimization of Microwave/RF Circuits and Systems, Vol. 20, No. 5, pp. 492-501, September 2010.

Contact : ivo.couckuyt@ugent.be - http://sumo.intec.ugent.be/?q=ooDACE Signature [corr, dx, dtheta, rho] = corrspherical(theta, d)

8.56.2 Function Documentation

8.56.2.1 function corrspherical (var *theta,* var *d*)

Spherical correlation function.

8.57 /home/ilm/projecten/meta/scripts/ooDACE/basisfunctions/corrspline.m File Reference

Functions

- function corrspline (var theta, var d)
  Cubic spline correlation function,.

8.57.1 Detailed Description

Authors

Ivo Couckuyt

Version

1.4 ($Revision$)

Date

$LastChangedDate$
Copyright 2010-2013

This file is part of the ooDACE toolbox and you can redistribute it and/or modify it under the terms of the GNU Affero General Public License version 3 as published by the Free Software Foundation. With the additional provision that a commercial license must be purchased if the ooDACE toolbox is used, modified, or extended in a commercial setting. For details see the included LICENSE.txt file. When referring to the ooDACE toolbox please make reference to the corresponding publications:

- Blind Kriging: Implementation and performance analysis I. Couckuyt, A. Forrester, D. Gorissen, F. De Turck, T. Dhaene, Advances in Engineering Software, Vol. 49, pp. 1-13, July 2012.
- Surrogate-based infill optimization applied to electromagnetic problems I. Couckuyt, F. Declercq, T. Dhaene, H. Rogier, L. Knockaert, International Journal of RF and Microwave Computer-Aided Engineering (RFMiCA-E), Special Issue on Advances in Design Optimization of Microwave/RF Circuits and Systems, Vol. 20, No. 5, pp. 492-501, September 2010.

Contact : ivo.couckuyt@ugent.be - http://sumo.intec.ugent.be/?q=ooDACE Signature [corr, dx, dtheta, rho] = corrspline(theta, d)

8.57.2 Function Documentation

8.57.2.1 function corrspline (var *theta,* var *d*)

Cubic spline correlation function,.

8.58 /home/ilm/projecten/meta/scripts/ooDACE/basisfunctions/covmatern32.m File Reference

Functions

- function corrmatern32_variance (var hp, var d)

  Class of Matern covariance functions (nu = {1/2 3/2 5/2}.

8.58.1 Detailed Description

Authors

Ivo Couckuyt

Version

1.4 ($Revision$)

Date

$LastChangedDate$
Copyright 2010-2013

This file is part of the ooDACE toolbox and you can redistribute it and/or modify it under the terms of the GNU Affero General Public License version 3 as published by the Free Software Foundation. With the additional provision that a commercial license must be purchased if the ooDACE toolbox is used, modified, or extended in a commercial setting. For details see the included LICENSE.txt file. When referring to the ooDACE toolbox please make reference to the corresponding publications:

- Blind Kriging: Implementation and performance analysis I. Couckuyt, A. Forrester, D. Gorissen, F. De Turck, T. Dhaene, Advances in Engineering Software, Vol. 49, pp. 1-13, July 2012.
- Surrogate-based infill optimization applied to electromagnetic problems I. Couckuyt, F. Declercq, T. Dhaene, H. Rogier, L. Knockaert, International Journal of RF and Microwave Computer-Aided Engineering (RFMiCA-E), Special Issue on Advances in Design Optimization of Microwave/RF Circuits and Systems, Vol. 20, No. 5, pp. 492-501, September 2010.

Contact : ivo.couckuyt@ugent.be - http://sumo.intec.ugent.be/?q=ooDACE Signature [corr, dx, dhp, rho] = corrmatern32_variance(hp, d)

8.58.2 Function Documentation

8.58.2.1 function corrmatern32_variance (var *hp,* var *d*)

Class of Matern covariance functions (nu = {1/2 3/2 5/2}.

Implementation based on the book of Rasmussen et al.

- "Gaussian Processes for Machine Learning" (Chapter 4), C. E. Rasmussen and C. K. I. Williams, MIT Press, 2006

The correlation function is: corr = f(sqrt(nud)∗dist2) ∗ exp(-sqrt(nud)∗dist2) with nud = 2.∗nu

8.59 /home/ilm/projecten/meta/scripts/ooDACE/dacefit.m File Reference

Functions

- function dacefit (var samples, var values, var regr, var corr, var theta0, var lb, var ub)
  Creates and fits a kriging model.

8.59.1 Detailed Description

Authors

Ivo Couckuyt

Version

1.4 ($Revision$)

Date

$LastChangedDate$
Copyright 2010-2013

This file is part of the ooDACE toolbox and you can redistribute it and/or modify it under the terms of the GNU Affero General Public License version 3 as published by the Free Software Foundation. With the additional provision that a commercial license must be purchased if the ooDACE toolbox is used, modified, or extended in a commercial setting. For details see the included LICENSE.txt file. When referring to the ooDACE toolbox please make reference to the corresponding publications:

- Blind Kriging: Implementation and performance analysis I. Couckuyt, A. Forrester, D. Gorissen, F. De Turck, T. Dhaene, Advances in Engineering Software, Vol. 49, pp. 1-13, July 2012.
- Surrogate-based infill optimization applied to electromagnetic problems I. Couckuyt, F. Declercq, T. Dhaene, H. Rogier, L. Knockaert, International Journal of RF and Microwave Computer-Aided Engineering (RFMiCA-E), Special Issue on Advances in Design Optimization of Microwave/RF Circuits and Systems, Vol. 20, No. 5, pp. 492-501, September 2010.

Contact : ivo.couckuyt@ugent.be - http://sumo.intec.ugent.be/?q=ooDACE Signature [krige perf] = dacefit(samples, values, regr, corr, theta0, lb, ub)

8.59.2 Function Documentation

8.59.2.1 function dacefit (var *samples,* var *values,* var *regr,* var *corr,* var *theta0,* var *lb,* var *ub*)

Creates and fits a kriging model.

DACE toolbox compatible interface to ooDACE (wrapper)

Parameters

samples	input sample matrix
values	output value matrix
regr	regression function (string)
corr	correlation function (string)
theta0	initial hyperparameter values
lb	lower bound of hyperparameters
ub	upper bound of hyperparameters

Return values

krige	a ready-to-use kriging model
perf	a structure with some useful metrics

8.60 /home/ilm/projecten/meta/scripts/ooDACE/datasets/generateDatasets.m File Reference

Functions

- function generateDatasets ()

  Generates some example datasets used by the demo.
- function branin (var x)
- function birdfcn (var x)

8.60.1 Detailed Description

Authors

Ivo Couckuyt

Version

1.4 ($Revision$)

Date

$LastChangedDate$
Copyright 2010-2013

This file is part of the ooDACE toolbox and you can redistribute it and/or modify it under the terms of the GNU Affero General Public License version 3 as published by the Free Software Foundation. With the additional provision that a commercial license must be purchased if the ooDACE toolbox is used, modified, or extended in a commercial setting. For details see the included LICENSE.txt file. When referring to the ooDACE toolbox please make reference to the corresponding publications:

- Blind Kriging: Implementation and performance analysis I. Couckuyt, A. Forrester, D. Gorissen, F. De Turck, T. Dhaene, Advances in Engineering Software, Vol. 49, pp. 1-13, July 2012.
- Surrogate-based infill optimization applied to electromagnetic problems I. Couckuyt, F. Declercq, T. Dhaene, H. Rogier, L. Knockaert, International Journal of RF and Microwave Computer-Aided Engineering (RFMiCA-E), Special Issue on Advances in Design Optimization of Microwave/RF Circuits and Systems, Vol. 20, No. 5, pp. 492-501, September 2010.

Contact : ivo.couckuyt@ugent.be - http://sumo.intec.ugent.be/?q=ooDACE Signature generateDatasets()

8.60.2 Function Documentation

8.60.2.1 function birdfcn (var *x*)

8.60.2.2 function branin (var *x*)

8.60.2.3 function generateDatasets ( )

Generates some example datasets used by the demo.

8.61 /home/ilm/projecten/meta/scripts/ooDACE/demo.m File Reference

Functions

- function demo (var id)

  Provides some examples on how to use the ooDACE toolbox.

8.61.1 Detailed Description

Authors

Ivo Couckuyt

Version

1.4 ($Revision$)

Date

$LastChangedDate$
Copyright 2010-2013

This file is part of the ooDACE toolbox and you can redistribute it and/or modify it under the terms of the GNU Affero General Public License version 3 as published by the Free Software Foundation. With the additional provision that a commercial license must be purchased if the ooDACE toolbox is used, modified, or extended in a commercial setting. For details see the included LICENSE.txt file. When referring to the ooDACE toolbox please make reference to the corresponding publications:

- Blind Kriging: Implementation and performance analysis I. Couckuyt, A. Forrester, D. Gorissen, F. De Turck, T. Dhaene, Advances in Engineering Software, Vol. 49, pp. 1-13, July 2012.
- Surrogate-based infill optimization applied to electromagnetic problems I. Couckuyt, F. Declercq, T. Dhaene, H. Rogier, L. Knockaert, International Journal of RF and Microwave Computer-Aided Engineering (RFMiCA-E), Special Issue on Advances in Design Optimization of Microwave/RF Circuits and Systems, Vol. 20, No. 5, pp. 492-501, September 2010.

Contact : ivo.couckuyt@ugent.be - http://sumo.intec.ugent.be/?q=ooDACE Signature k = demo(id)

8.61.2 Function Documentation

8.61.2.1 function demo (var *id*)

Provides some examples on how to use the ooDACE toolbox.

Test Test case 1: Fits an ordinary kriging model on the branin function Covers: ordinary kriging interpolation, marginalLikelihood

Test case 2: Fits an ordinary kriging model on the bird function Covers: ordinary kriging regression, re-interpolation of variance, pseudoLikelihood, likelihood debug plot

Test case 3: Fits a blind kriging model on the branin function. Covers: blind kriging

Test case 4: Fits a cokriging model on a mathematical 1D function. Covers: cokriging

Test case 5: Fits a stochastic kriging model on the branin function plus some stochastic noise. Covers: stochastic kriging, sigma2 optimization

Parameters

id	id of the dataset to use (optional)

Return values

k	a ready-to-use kriging model

8.62 /home/ilm/projecten/meta/scripts/ooDACE/doc/mainpage.m File Reference

8.63 /home/ilm/projecten/meta/scripts/ooDACE/oodacefit.m File Reference

Functions

- function oodacefit (var samples, var values, var userOpts)
  Creates and fits a kriging model with sensible options.

8.63.1 Detailed Description

Authors

Ivo Couckuyt

Version

1.4 ($Revision$)

Date

$LastChangedDate$
 When referring to the ooDACE toolbox please make reference to the corresponding publications:

- Blind Kriging: Implementation and performance analysis I. Couckuyt, A. Forrester, D. Gorissen, F. De Turck, T. Dhaene, Advances in Engineering Software, Vol. 49, pp. 1-13, July 2012.
- Surrogate-based infill optimization applied to electromagnetic problems I. Couckuyt, F. Declercq, T. Dhaene, H. Rogier, L. Knockaert, International Journal of RF and Microwave Computer-Aided Engineering (RFMiCA-E), Special Issue on Advances in Design Optimization of Microwave/RF Circuits and Systems, Vol. 20, No. 5, pp. 492-501, September 2010.

Contact : ivo.couckuyt@ugent.be - http://sumo.intec.ugent.be/?q=ooDACE Signature k = oodacefit(samples, values, userOpts)

8.63.2 Function Documentation

8.63.2.1 function oodacefit (var *samples,* var *values,* var *userOpts*)

Creates and fits a kriging model with sensible options.

```
For easy fitting when you don't care much about parameters.
The function tries to determine the best kriging model to use:
- samples is numeric array -> kriging
- samples is cell array -> cokriging
```

Parameters

samples	input sample matrix
values	output value matrix
userOpts	struct of user options (optional) • userOpts.type overrides the type of kriging model to use, e.g., "BasicGaussianProcess", "Kriging", "CoKriging", etc. • For regression kriging set the following fields: – userOpts.lambda0 = 0; – userOpts.lambdaBounds = [-5 ; 5]; % log scale • For stochastic kriging set the following fields: – userOpts.Sigma = SigmaVector; % variance of the output values – userOpts.sigma20 = 1; – userOpts.sigma2Bounds = [0.001 ; 150]; • Please see the BasicGaussianProcess::getDefaultOptions(), Kriging::getDefaultOptions(), etc. methods for more options.

Return values

k	a ready-to-use kriging model

8.64 /home/ilm/projecten/meta/scripts/ooDACE/optimizers/@MatlabGA/getPopulationSize.m File Reference

8.64.1 Detailed Description

Authors

Ivo Couckuyt

Version

1.4 ($Revision$)

Date

$LastChangedDate$

- Blind Kriging: Implementation and performance analysis I. Couckuyt, A. Forrester, D. Gorissen, F. De Turck, T. Dhaene, Advances in Engineering Software, Vol. 49, pp. 1-13, July 2012.
- Surrogate-based infill optimization applied to electromagnetic problems I. Couckuyt, F. Declercq, T. Dhaene, H. Rogier, L. Knockaert, International Journal of RF and Microwave Computer-Aided Engineering (RFMiCA-E), Special Issue on Advances in Design Optimization of Microwave/RF Circuits and Systems, Vol. 20, No. 5, pp. 492-501, September 2010.

Contact : ivo.couckuyt@ugent.be - http://sumo.intec.ugent.be/?q=ooDACE Signature size = getPopulationSize(this)

8.65 /home/ilm/projecten/meta/scripts/ooDACE/optimizers/@Optimizer/getPopulationSize.m File Reference

8.65.1 Detailed Description

Authors

Ivo Couckuyt

Version

1.4 ($Revision$)

Date

$LastChangedDate$

- Blind Kriging: Implementation and performance analysis I. Couckuyt, A. Forrester, D. Gorissen, F. De Turck, T. Dhaene, Advances in Engineering Software, Vol. 49, pp. 1-13, July 2012.
- Surrogate-based infill optimization applied to electromagnetic problems I. Couckuyt, F. Declercq, T. Dhaene, H. Rogier, L. Knockaert, International Journal of RF and Microwave Computer-Aided Engineering (RFMiCA-E), Special Issue on Advances in Design Optimization of Microwave/RF Circuits and Systems, Vol. 20, No. 5, pp. 492-501, September 2010.

Contact : ivo.couckuyt@ugent.be - http://sumo.intec.ugent.be/?q=ooDACE Signature size = getPopulationSize(this)

8.66 /home/ilm/projecten/meta/scripts/ooDACE/optimizers/@MatlabGA/MatlabGA.m File Reference

Classes

- class MatlabGA

  Wrapper around the matlab optimizers.

8.66.1 Detailed Description

Authors

Ivo Couckuyt

Version

1.4 ($Revision$)

Date

$LastChangedDate$
Copyright 2010-2013

This file is part of the ooDACE toolbox and you can redistribute it and/or modify it under the terms of the GNU Affero General Public License version 3 as published by the Free Software Foundation. With the additional provision that a commercial license must be purchased if the ooDACE toolbox is used, modified, or extended in a commercial setting. For details see the included LICENSE.txt file. When referring to the ooDACE toolbox please make reference to the corresponding publications:

- Blind Kriging: Implementation and performance analysis I. Couckuyt, A. Forrester, D. Gorissen, F. De Turck, T. Dhaene, Advances in Engineering Software, Vol. 49, pp. 1-13, July 2012.
- Surrogate-based infill optimization applied to electromagnetic problems I. Couckuyt, F. Declercq, T. Dhaene, H. Rogier, L. Knockaert, International Journal of RF and Microwave Computer-Aided Engineering (RFMiCA-E), Special Issue on Advances in Design Optimization of Microwave/RF Circuits and Systems, Vol. 20, No. 5, pp. 492-501, September 2010.

Contact : ivo.couckuyt@ugent.be - http://sumo.intec.ugent.be/?q=ooDACE Signature MatlabGA

8.67 /home/ilm/projecten/meta/scripts/ooDACE/optimizers/@MatlabGA/optimize.m File Reference

8.67.1 Detailed Description

Authors

Ivo Couckuyt

Version

1.4 ($Revision$)

Date

$LastChangedDate$
Copyright 2010-2013

This file is part of the ooDACE toolbox and you can redistribute it and/or modify it under the terms of the GNU Affero General Public License version 3 as published by the Free Software Foundation. With the additional provision that a commercial license must be purchased if the ooDACE toolbox is used, modified, or extended in a commercial setting. For details see the included LICENSE.txt file. When referring to the ooDACE toolbox please make reference to the corresponding publications:

- Blind Kriging: Implementation and performance analysis I. Couckuyt, A. Forrester, D. Gorissen, F. De Turck, T. Dhaene, Advances in Engineering Software, Vol. 49, pp. 1-13, July 2012.

- Surrogate-based infill optimization applied to electromagnetic problems I. Couckuyt, F. Declercq, T. Dhaene, H. Rogier, L. Knockaert, International Journal of RF and Microwave Computer-Aided Engineering (RFMiCA-E), Special Issue on Advances in Design Optimization of Microwave/RF Circuits and Systems, Vol. 20, No. 5, pp. 492-501, September 2010.

Contact : ivo.couckuyt@ugent.be - http://sumo.intec.ugent.be/?q=ooDACE Signature [this, x, fval] = optimize(this, arg)

8.68 /home/ilm/projecten/meta/scripts/ooDACE/optimizers/@MatlabOptimizer/optimize.m File Reference

8.68.1 Detailed Description

Authors

Ivo Couckuyt

Version

1.4 ($Revision$)

Date

$LastChangedDate$
Copyright 2010-2013

This file is part of the ooDACE toolbox and you can redistribute it and/or modify it under the terms of the GNU Affero General Public License version 3 as published by the Free Software Foundation. With the additional provision that a commercial license must be purchased if the ooDACE toolbox is used, modified, or extended in a commercial setting. For details see the included LICENSE.txt file. When referring to the ooDACE toolbox please make reference to the corresponding publications:

- Blind Kriging: Implementation and performance analysis I. Couckuyt, A. Forrester, D. Gorissen, F. De Turck, T. Dhaene, Advances in Engineering Software, Vol. 49, pp. 1-13, July 2012.
- Surrogate-based infill optimization applied to electromagnetic problems I. Couckuyt, F. Declercq, T. Dhaene, H. Rogier, L. Knockaert, International Journal of RF and Microwave Computer-Aided Engineering (RFMiCA-E), Special Issue on Advances in Design Optimization of Microwave/RF Circuits and Systems, Vol. 20, No. 5, pp. 492-501, September 2010.

Contact : ivo.couckuyt@ugent.be - http://sumo.intec.ugent.be/?q=ooDACE Signature [this, xmin, fmin] = optimize(this, arg)

8.69 /home/ilm/projecten/meta/scripts/ooDACE/optimizers/@Optimizer/optimize.m File Reference

8.69.1 Detailed Description

Authors

Ivo Couckuyt

Version

1.4 ($Revision$)

Date

$LastChangedDate$
Copyright 2010-2013

This file is part of the ooDACE toolbox and you can redistribute it and/or modify it under the terms of the GNU Affero General Public License version 3 as published by the Free Software Foundation. With the additional provision that a commercial license must be purchased if the ooDACE toolbox is used, modified, or extended in a commercial setting. For details see the included LICENSE.txt file. When referring to the ooDACE toolbox please make reference to the corresponding publications:

- Blind Kriging: Implementation and performance analysis I. Couckuyt, A. Forrester, D. Gorissen, F. De Turck, T. Dhaene, Advances in Engineering Software, Vol. 49, pp. 1-13, July 2012.
- Surrogate-based infill optimization applied to electromagnetic problems I. Couckuyt, F. Declercq, T. Dhaene, H. Rogier, L. Knockaert, International Journal of RF and Microwave Computer-Aided Engineering (RFMiCA-E), Special Issue on Advances in Design Optimization of Microwave/RF Circuits and Systems, Vol. 20, No. 5, pp. 492-501, September 2010.

Contact : ivo.couckuyt@ugent.be - http://sumo.intec.ugent.be/?q=ooDACE Signature [this, x, fval] = optimize(this, arg)

8.70 /home/ilm/projecten/meta/scripts/ooDACE/optimizers/@SQPLabOptimizer/optimize.m File Reference

8.70.1 Detailed Description

Authors

Ivo Couckuyt

Version

1.4 ($Revision$)

Date

$LastChangedDate$
Copyright 2010-2013

This file is part of the ooDACE toolbox and you can redistribute it and/or modify it under the terms of the GNU Affero General Public License version 3 as published by the Free Software Foundation. With the additional provision that a commercial license must be purchased if the ooDACE toolbox is used, modified, or extended in a commercial setting. For details see the included LICENSE.txt file. When referring to the ooDACE toolbox please make reference to the corresponding publications:

- Blind Kriging: Implementation and performance analysis I. Couckuyt, A. Forrester, D. Gorissen, F. De Turck, T. Dhaene, Advances in Engineering Software, Vol. 49, pp. 1-13, July 2012.
- Surrogate-based infill optimization applied to electromagnetic problems I. Couckuyt, F. Declercq, T. Dhaene, H. Rogier, L. Knockaert, International Journal of RF and Microwave Computer-Aided Engineering (RFMiCA-E), Special Issue on Advances in Design Optimization of Microwave/RF Circuits and Systems, Vol. 20, No. 5, pp. 492-501, September 2010.

Contact : ivo.couckuyt@ugent.be - http://sumo.intec.ugent.be/?q=ooDACE Signature [this, x, fval] = optimize(this, arg)

8.71 /home/ilm/projecten/meta/scripts/ooDACE/optimizers/@MatlabGA/setInputConstraints.m File Reference

8.71.1 Detailed Description

Authors

Ivo Couckuyt

Version

1.4 ($Revision$)

Date

$LastChangedDate$

Contact : ivo.couckuyt@ugent.be - http://sumo.intec.ugent.be/?q=ooDACE Signature this = setInputConstraints(this, con)

8.72 /home/ilm/projecten/meta/scripts/ooDACE/optimizers/@MatlabOptimizer/setInputConstraints.m File Reference

8.72.1 Detailed Description

Authors

Ivo Couckuyt

Version

1.4 ($Revision$)

Date

$LastChangedDate$

- Blind Kriging: Implementation and performance analysis I. Couckuyt, A. Forrester, D. Gorissen, F. De Turck, T. Dhaene, Advances in Engineering Software, Vol. 49, pp. 1-13, July 2012.
- Surrogate-based infill optimization applied to electromagnetic problems I. Couckuyt, F. Declercq, T. Dhaene, H. Rogier, L. Knockaert, International Journal of RF and Microwave Computer-Aided Engineering (RFMiCA-E), Special Issue on Advances in Design Optimization of Microwave/RF Circuits and Systems, Vol. 20, No. 5, pp. 492-501, September 2010.

Contact : ivo.couckuyt@ugent.be - http://sumo.intec.ugent.be/?q=ooDACE Signature this = setInputConstraints(this, con)

8.73 /home/ilm/projecten/meta/scripts/ooDACE/optimizers/@Optimizer/setInputConstraints.m File Reference

8.73.1 Detailed Description

Authors

Ivo Couckuyt

Version

1.4 ($Revision$)

Date

$LastChangedDate$

- Blind Kriging: Implementation and performance analysis I. Couckuyt, A. Forrester, D. Gorissen, F. De Turck, T. Dhaene, Advances in Engineering Software, Vol. 49, pp. 1-13, July 2012.
- Surrogate-based infill optimization applied to electromagnetic problems I. Couckuyt, F. Declercq, T. Dhaene, H. Rogier, L. Knockaert, International Journal of RF and Microwave Computer-Aided Engineering (RFMiCA-E), Special Issue on Advances in Design Optimization of Microwave/RF Circuits and Systems, Vol. 20, No. 5, pp. 492-501, September 2010.

Contact : ivo.couckuyt@ugent.be - http://sumo.intec.ugent.be/?q=ooDACE Signature this = setInputConstraints(this, con)

8.74 /home/ilm/projecten/meta/scripts/ooDACE/optimizers/@MatlabOptimizer/MatlabOptimizer.m File Reference

Classes

- class MatlabOptimizer

  Wrapper around the matlab optimizers.

8.74.1 Detailed Description

Authors

Ivo Couckuyt

Version

1.4 ($Revision$)

Date

$LastChangedDate$

Contact : ivo.couckuyt@ugent.be - http://sumo.intec.ugent.be/?q=ooDACE Signature MatlabOptimizer

8.75 /home/ilm/projecten/meta/scripts/ooDACE/optimizers/@Optimizer/getBounds.m File Reference

8.75.1 Detailed Description

Authors

Ivo Couckuyt

Version

1.4 ($Revision$)

Date

$LastChangedDate$

- Surrogate-based infill optimization applied to electromagnetic problems I. Couckuyt, F. Declercq, T. Dhaene, H. Rogier, L. Knockaert, International Journal of RF and Microwave Computer-Aided Engineering (RFMiCA-E), Special Issue on Advances in Design Optimization of Microwave/RF Circuits and Systems, Vol. 20, No. 5, pp. 492-501, September 2010.

Contact : ivo.couckuyt@ugent.be - http://sumo.intec.ugent.be/?q=ooDACE Signature [LB UB] = getBounds(this)

8.76 /home/ilm/projecten/meta/scripts/ooDACE/optimizers/@Optimizer/getHint.m File Reference

8.76.1 Detailed Description

Authors

Ivo Couckuyt

Version

1.4 ($Revision$)

Date

$LastChangedDate$

Contact : ivo.couckuyt@ugent.be - http://sumo.intec.ugent.be/?q=ooDACE Signature value = getHint(this, key)

8.77 /home/ilm/projecten/meta/scripts/ooDACE/optimizers/@Optimizer/getInitialPopulation.m File Reference

8.77.1 Detailed Description

Authors

Ivo Couckuyt

Version

1.4 ($Revision$)

Date

$LastChangedDate$
Copyright 2010-2013

This file is part of the ooDACE toolbox and you can redistribute it and/or modify it under the terms of the GNU Affero General Public License version 3 as published by the Free Software Foundation. With the additional provision that a commercial license must be purchased if the ooDACE toolbox is used, modified, or extended in a commercial setting. For details see the included LICENSE.txt file. When referring to the ooDACE toolbox please make reference to the corresponding publications:

- Blind Kriging: Implementation and performance analysis I. Couckuyt, A. Forrester, D. Gorissen, F. De Turck, T. Dhaene, Advances in Engineering Software, Vol. 49, pp. 1-13, July 2012.
- Surrogate-based infill optimization applied to electromagnetic problems I. Couckuyt, F. Declercq, T. Dhaene, H. Rogier, L. Knockaert, International Journal of RF and Microwave Computer-Aided Engineering (RFMiCAE), Special Issue on Advances in Design Optimization of Microwave/RF Circuits and Systems, Vol. 20, No. 5, pp. 492-501, September 2010.

Contact : ivo.couckuyt@ugent.be - http://sumo.intec.ugent.be/?q=ooDACE Signature pop = getInitialPopulation(this)

8.78 /home/ilm/projecten/meta/scripts/ooDACE/optimizers/@Optimizer/getInputDimension.m File Reference

8.78.1 Detailed Description

Authors

Ivo Couckuyt

Version

1.4 ($Revision$)

Date

$LastChangedDate$
Copyright 2010-2013

This file is part of the ooDACE toolbox and you can redistribute it and/or modify it under the terms of the GNU Affero General Public License version 3 as published by the Free Software Foundation. With the additional provision that a commercial license must be purchased if the ooDACE toolbox is used, modified, or extended in a commercial setting. For details see the included LICENSE.txt file. When referring to the ooDACE toolbox please make reference to the corresponding publications:

- Blind Kriging: Implementation and performance analysis I. Couckuyt, A. Forrester, D. Gorissen, F. De Turck, T. Dhaene, Advances in Engineering Software, Vol. 49, pp. 1-13, July 2012.
- Surrogate-based infill optimization applied to electromagnetic problems I. Couckuyt, F. Declercq, T. Dhaene, H. Rogier, L. Knockaert, International Journal of RF and Microwave Computer-Aided Engineering (RFMiCAE), Special Issue on Advances in Design Optimization of Microwave/RF Circuits and Systems, Vol. 20, No. 5, pp. 492-501, September 2010.

Contact : ivo.couckuyt@ugent.be - http://sumo.intec.ugent.be/?q=ooDACE Signature nvars = getInputDimension(this)

8.79 /home/ilm/projecten/meta/scripts/ooDACE/optimizers/@Optimizer/getOutputDimension.m File Reference

8.79.1 Detailed Description

Authors

Ivo Couckuyt

Version

1.4 ($Revision$)

Date

$LastChangedDate$
 Signature nobjectives = getOutputDimension(this)

8.80 /home/ilm/projecten/meta/scripts/ooDACE/optimizers/@Optimizer/Optimizer.m File Reference

Classes

- class Optimizer

  Abstract base class for an optimizer.

8.80.1 Detailed Description

Authors

Ivo Couckuyt

Version

1.4 ($Revision$)

Date

$LastChangedDate$

8.81 /home/ilm/projecten/meta/scripts/ooDACE/optimizers/@Optimizer/setBounds.m File Reference

8.81.1 Detailed Description

Authors

Ivo Couckuyt

Version

1.4 ($Revision$)

Date

$LastChangedDate$

8.82 /home/ilm/projecten/meta/scripts/ooDACE/optimizers/@Optimizer/setDimensions.m File Reference

8.82.1 Detailed Description

Authors

Ivo Couckuyt

Version

1.4 ($Revision$)

Date

$LastChangedDate$
 Signature this = setDimensions(this,inDim,outDim)

8.83 /home/ilm/projecten/meta/scripts/ooDACE/optimizers/@Optimizer/setHint.m File Reference

8.83.1 Detailed Description

Authors

Ivo Couckuyt

Version

1.4 ($Revision$)

Date

$LastChangedDate$

- Blind Kriging: Implementation and performance analysis I. Couckuyt, A. Forrester, D. Gorissen, F. De Turck, T. Dhaene, Advances in Engineering Software, Vol. 49, pp. 1-13, July 2012.
- Surrogate-based infill optimization applied to electromagnetic problems I. Couckuyt, F. Declercq, T. Dhaene, H. Rogier, L. Knockaert, International Journal of RF and Microwave Computer-Aided Engineering (RFMiCA-E), Special Issue on Advances in Design Optimization of Microwave/RF Circuits and Systems, Vol. 20, No. 5, pp. 492-501, September 2010.

Contact : ivo.couckuyt@ugent.be - http://sumo.intec.ugent.be/?q=ooDACE Signature this = setHint(this, key, value)

8.84 /home/ilm/projecten/meta/scripts/ooDACE/optimizers/@Optimizer/setInitialPopulation.m File Reference

8.84.1 Detailed Description

Authors

Ivo Couckuyt

Version

1.4 ($Revision$)

Date

$LastChangedDate$

Contact : ivo.couckuyt@ugent.be - http://sumo.intec.ugent.be/?q=ooDACE Signature this = setInitialPopulation(this,pop)

8.85 /home/ilm/projecten/meta/scripts/ooDACE/optimizers/@SQPLabOptimizer/simulator.m File Reference

8.85.1 Detailed Description

Authors

Ivo Couckuyt

Version

1.4 ($Revision$)

Date

$LastChangedDate$
Copyright 2010-2013

This file is part of the ooDACE toolbox and you can redistribute it and/or modify it under the terms of the GNU Affero General Public License version 3 as published by the Free Software Foundation. With the additional provision that a commercial license must be purchased if the ooDACE toolbox is used, modified, or extended in a commercial setting. For details see the included LICENSE.txt file. When referring to the ooDACE toolbox please make reference to the corresponding publications:

- Blind Kriging: Implementation and performance analysis I. Couckuyt, A. Forrester, D. Gorissen, F. De Turck, T. Dhaene, Advances in Engineering Software, Vol. 49, pp. 1-13, July 2012.
- Surrogate-based infill optimization applied to electromagnetic problems I. Couckuyt, F. Declercq, T. Dhaene, H. Rogier, L. Knockaert, International Journal of RF and Microwave Computer-Aided Engineering (RFMiCA-E), Special Issue on Advances in Design Optimization of Microwave/RF Circuits and Systems, Vol. 20, No. 5, pp. 492-501, September 2010.

Contact : ivo.couckuyt@ugent.be - http://sumo.intec.ugent.be/?q=ooDACE Signature [outdic,f,ci,ce,cs,g,ai,ae] = simulator(indic, x, func, this, varargin)

8.86 /home/ilm/projecten/meta/scripts/ooDACE/optimizers/@SQPLabOptimizer/SQPLabOptimizer.m File Reference

Classes

- class SQPLabOptimizer
  Wrapper around the SQPLab optimization package.

8.86.1 Detailed Description

Authors

Ivo Couckuyt

Version

1.4 ($Revision$)

Date

$LastChangedDate$
Copyright 2010-2013

This file is part of the ooDACE toolbox and you can redistribute it and/or modify it under the terms of the GNU Affero General Public License version 3 as published by the Free Software Foundation. With the additional provision that a commercial license must be purchased if the ooDACE toolbox is used, modified, or extended in a commercial setting. For details see the included LICENSE.txt file. When referring to the ooDACE toolbox please make reference to the corresponding publications:

- Blind Kriging: Implementation and performance analysis I. Couckuyt, A. Forrester, D. Gorissen, F. De Turck, T. Dhaene, Advances in Engineering Software, Vol. 49, pp. 1-13, July 2012.
- Surrogate-based infill optimization applied to electromagnetic problems I. Couckuyt, F. Declercq, T. Dhaene, H. Rogier, L. Knockaert, International Journal of RF and Microwave Computer-Aided Engineering (RFMiCA-E), Special Issue on Advances in Design Optimization of Microwave/RF Circuits and Systems, Vol. 20, No. 5, pp. 492-501, September 2010.

Contact : ivo.couckuyt@ugent.be - http://sumo.intec.ugent.be/?q=ooDACE Signature SQP-LabOptimizer

8.87 /home/ilm/projecten/meta/scripts/ooDACE/plotKrigingModel.m File Reference

Functions

- function plotKrigingModel (var k, var LB, var UB)

  Generate some plots of a Kriging model.

8.87.1 Detailed Description

Authors

Ivo Couckuyt

Version

1.4 ($Revision$)

Date

$LastChangedDate$
Copyright 2010-2013

This file is part of the ooDACE toolbox and you can redistribute it and/or modify it under the terms of the GNU Affero General Public License version 3 as published by the Free Software Foundation. With the additional provision that a commercial license must be purchased if the ooDACE toolbox is used, modified, or extended in a commercial setting. For details see the included LICENSE.txt file. When referring to the ooDACE toolbox please make reference to the corresponding publications:

- Blind Kriging: Implementation and performance analysis I. Couckuyt, A. Forrester, D. Gorissen, F. De Turck, T. Dhaene, Advances in Engineering Software, Vol. 49, pp. 1-13, July 2012.
- Surrogate-based infill optimization applied to electromagnetic problems I. Couckuyt, F. Declercq, T. Dhaene, H. Rogier, L. Knockaert, International Journal of RF and Microwave Computer-Aided Engineering (RFMiCA-E), Special Issue on Advances in Design Optimization of Microwave/RF Circuits and Systems, Vol. 20, No. 5, pp. 492-501, September 2010.

Contact : ivo.couckuyt@ugent.be - http://sumo.intec.ugent.be/?q=ooDACE Signature h = plotKrigingModel(k, LB, UB)

8.87.2 Function Documentation

8.87.2.1 function plotKrigingModel (var *k,* var *LB,* var *UB*)

Generate some plots of a Kriging model.

Only for 1D/2D models.

Parameters

k	Kriging model
LB	lowerbound of input parameters (optional)
UB	upperbound of input parameters (optional)

Return values

h	handles to the figure windows

8.88 /home/ilm/projecten/meta/scripts/ooDACE/predictor.m File Reference

Functions

- function predictor (var points, var krige)

  Calculates prediction of a kriging model.

8.88.1 Detailed Description

Authors

Ivo Couckuyt

Version

1.4 ($Revision$)

Date

$LastChangedDate$
Copyright 2010-2013

This file is part of the ooDACE toolbox and you can redistribute it and/or modify it under the terms of the GNU Affero General Public License version 3 as published by the Free Software Foundation. With the additional provision that a commercial license must be purchased if the ooDACE toolbox is used, modified, or extended in a commercial setting. For details see the included LICENSE.txt file. When referring to the ooDACE toolbox please make reference to the corresponding publications:

- Blind Kriging: Implementation and performance analysis I. Couckuyt, A. Forrester, D. Gorissen, F. De Turck, T. Dhaene, Advances in Engineering Software, Vol. 49, pp. 1-13, July 2012.
- Surrogate-based infill optimization applied to electromagnetic problems I. Couckuyt, F. Declercq, T. Dhaene, H. Rogier, L. Knockaert, International Journal of RF and Microwave Computer-Aided Engineering (RFMiCA-E), Special Issue on Advances in Design Optimization of Microwave/RF Circuits and Systems, Vol. 20, No. 5, pp. 492-501, September 2010.

Contact : ivo.couckuyt@ugent.be - http://sumo.intec.ugent.be/?q=ooDACE Signature [y, or1, or2, dmse] = predictor(points, krige)

8.88.2 Function Documentation

8.88.2.1 function predictor (var *points,* var *krige*)

Calculates prediction of a kriging model.

DACE toolbox compatible interface to ooDACE (wrapper)

Parameters

points	input points matrix
krige	kriging model

Return values

y	prediction values
or1	
or2	
dmse	

8.89 /home/ilm/projecten/meta/scripts/ooDACE/runBlindKrigingExamples.m File Reference

Functions

- function runBlindKrigingExamples ()

  Fits blind kriging models to some datasets.

8.89.1 Detailed Description

Authors

Ivo Couckuyt

Version

1.4 ($Revision$)

Date

$LastChangedDate$
Copyright 2010-2013

This file is part of the ooDACE toolbox and you can redistribute it and/or modify it under the terms of the GNU Affero General Public License version 3 as published by the Free Software Foundation. With the additional provision that a commercial license must be purchased if the ooDACE toolbox is used, modified, or extended in a commercial setting. For details see the included LICENSE.txt file. When referring to the ooDACE toolbox please make reference to the corresponding publications:

- Blind Kriging: Implementation and performance analysis I. Couckuyt, A. Forrester, D. Gorissen, F. De Turck, T. Dhaene, Advances in Engineering Software, Vol. 49, pp. 1-13, July 2012.
- Surrogate-based infill optimization applied to electromagnetic problems I. Couckuyt, F. Declercq, T. Dhaene, H. Rogier, L. Knockaert, International Journal of RF and Microwave Computer-Aided Engineering (RFMiCA-E), Special Issue on Advances in Design Optimization of Microwave/RF Circuits and Systems, Vol. 20, No. 5, pp. 492-501, September 2010.

Contact : ivo.couckuyt@ugent.be - http://sumo.intec.ugent.be/?q=ooDACE Signature runBlindKrigingExamples()

8.89.2 Function Documentation

8.89.2.1 function runBlindKrigingExamples ( )

Fits blind kriging models to some datasets.

8.90 /home/ilm/projecten/meta/scripts/ooDACE/runRegressionTests.m File Reference

Functions

- function runRegressionTests (var idx, var regressionTestsDir, var saveResults)

  Runs the regression test suite.
- function cmp_combinedRelative (var a, var b)

8.90.1 Detailed Description

Authors

Ivo Couckuyt

Version

1.4 ($Revision$)

Date

$LastChangedDate$
Copyright 2010-2013

This file is part of the ooDACE toolbox and you can redistribute it and/or modify it under the terms of the GNU Affero General Public License version 3 as published by the Free Software Foundation. With the additional provision that a commercial license must be purchased if the ooDACE toolbox is used, modified, or extended in a commercial setting. For details see the included LICENSE.txt file. When referring to the ooDACE toolbox please make reference to the corresponding publications:

- Blind Kriging: Implementation and performance analysis I. Couckuyt, A. Forrester, D. Gorissen, F. De Turck, T. Dhaene, Advances in Engineering Software, Vol. 49, pp. 1-13, July 2012.
- Surrogate-based infill optimization applied to electromagnetic problems I. Couckuyt, F. Declercq, T. Dhaene, H. Rogier, L. Knockaert, International Journal of RF and Microwave Computer-Aided Engineering (RFMiCA-E), Special Issue on Advances in Design Optimization of Microwave/RF Circuits and Systems, Vol. 20, No. 5, pp. 492-501, September 2010.

Contact : ivo.couckuyt@ugent.be - http://sumo.intec.ugent.be/?q=ooDACE Signature runRegressionTests(idx, regressionTestsDir, saveResults)

8.90.2 Function Documentation

8.90.2.1 function cmp_combinedRelative (var *a,* var *b*)

8.90.2.2 function runRegressionTests (var *idx,* var *regressionTestsDir,* var *saveResults*)

Runs the regression test suite.

Parameters

idx	vector indicating the tests to run. Default: 1:5 (optional)
regressionTestsDir	directory of results (optional)
saveResults	saves the results to regressionTestsDir. Default: false (optional)

Note

Tests should be deterministic, no fixed random state needed (yet)
The symbolic expression for the regression, correlation function and getExpression can not be compared directly

8.91 /home/ilm/projecten/meta/scripts/ooDACE/startup.m File Reference

Functions

- function startup ()

  Initializes the ooDACE toolbox.

8.91.1 Detailed Description

Authors

Ivo Couckuyt

Version

1.4 ($Revision$)

Date

$LastChangedDate$
Copyright 2010-2013

This file is part of the ooDACE toolbox and you can redistribute it and/or modify it under the terms of the GNU Affero General Public License version 3 as published by the Free Software Foundation. With the additional provision that a commercial license must be purchased if the ooDACE toolbox is used, modified, or extended in a commercial setting. For details see the included LICENSE.txt file. When referring to the ooDACE toolbox please make reference to the corresponding publications:

- Blind Kriging: Implementation and performance analysis I. Couckuyt, A. Forrester, D. Gorissen, F. De Turck, T. Dhaene, Advances in Engineering Software, Vol. 49, pp. 1-13, July 2012.
- Surrogate-based infill optimization applied to electromagnetic problems I. Couckuyt, F. Declercq, T. Dhaene, H. Rogier, L. Knockaert, International Journal of RF and Microwave Computer-Aided Engineering (RFMiCA-E), Special Issue on Advances in Design Optimization of Microwave/RF Circuits and Systems, Vol. 20, No. 5, pp. 492-501, September 2010.

Contact : ivo.couckuyt@ugent.be - http://sumo.intec.ugent.be/?q=ooDACE Signature startup()

8.91.2 Function Documentation

8.91.2.1 function startup ( )

Initializes the ooDACE toolbox.

Setup the toolbox path. Needs to be called once before using the ooDACE toolbox.

8.92 /home/ilm/projecten/meta/scripts/ooDACE/tools/averageEuclideanError.m File Reference

Functions

- function averageEuclideanError (var a, var b)

  Computes the average euclidean error (AEE) between a (true) and b (predicted).

8.92.1 Detailed Description

Authors

Ivo Couckuyt

Version

1.4 ($Revision$)

Date

$LastChangedDate$
 Signature e = averageEuclideanError(a,b)

8.92.2 Function Documentation

8.92.2.1 function averageEuclideanError (var *a*, var *b*)

Computes the average euclidean error (AEE) between a (true) and b (predicted).

From: Rong Li and Zhanlue Zhao, Evaluation of estimation algorithms part I: incomprehensive measures of performance, IEEE Transactions on Aerospace and Electronic Systems, vol. 42, no. 4, pp. 1340-1358, 2006

8.93 /home/ilm/projecten/meta/scripts/ooDACE/tools/buildVandermondeMatrix.m File Reference

Functions

- function buildVandermondeMatrix (var samples, var degrees, var baseFunctions)

  Build multidimensional Vandermonde like matrix for interpolation and/or evaluation of multidimensional polynomials.

8.93.1 Detailed Description

Authors

Ivo Couckuyt

Version

1.4 ($Revision$)

Date

$LastChangedDate$

Contact : ivo.couckuyt@ugent.be - http://sumo.intec.ugent.be/?q=ooDACE Signature [m dm] = buildVandermondeMatrix(samples, degrees, baseFunctions)

8.93.2 Function Documentation

8.93.2.1 function buildVandermondeMatrix (var *samples,* var *degrees,* var *baseFunctions*)

Build multidimensional Vandermonde like matrix for interpolation and/or evaluation of multidimensional polynomials.

8.94 /home/ilm/projecten/meta/scripts/ooDACE/tools/cfix.m File Reference

Functions

- function cfix (var x, var d, var err)

  This function "fixes" a cell array.

8.94.1 Detailed Description

Authors

Ivo Couckuyt

Version

1.4 ($Revision$)

Date

$LastChangedDate$

- Blind Kriging: Implementation and performance analysis I. Couckuyt, A. Forrester, D. Gorissen, F. De Turck, T. Dhaene, Advances in Engineering Software, Vol. 49, pp. 1-13, July 2012.
- Surrogate-based infill optimization applied to electromagnetic problems I. Couckuyt, F. Declercq, T. Dhaene, H. Rogier, L. Knockaert, International Journal of RF and Microwave Computer-Aided Engineering (RFMiCA-E), Special Issue on Advances in Design Optimization of Microwave/RF Circuits and Systems, Vol. 20, No. 5, pp. 492-501, September 2010.

Contact : ivo.couckuyt@ugent.be - http://sumo.intec.ugent.be/?q=ooDACE Signature y = cfix(x,d,err)

8.94.2 Function Documentation

8.94.2.1 function cfix (var *x,* var *d,* var *err*)

This function "fixes" a cell array.

Either a constant, a one element cell array or a length 'd' cell array can be passed to this function. The function will return a length 'd' cell array, duplicating the input if necessary

8.95 /home/ilm/projecten/meta/scripts/ooDACE/tools/makeEvalGrid.m File Reference

Functions

- function makeEvalGrid (var gridpoints, var gridsize)

  Low-level procedure, makes a 'prod(gridsize)' by 'length(gridsize)' array, where each row is a different vector, where the i'th element is a number out of 'gridpoints{i}'.

8.95.1 Detailed Description

Authors

Ivo Couckuyt

Version

1.4 ($Revision$)

Date

$LastChangedDate$
Copyright 2010-2013

This file is part of the ooDACE toolbox and you can redistribute it and/or modify it under the terms of the GNU Affero General Public License version 3 as published by the Free Software Foundation. With the additional provision that a commercial license must be purchased if the ooDACE toolbox is used, modified, or extended in a commercial setting. For details see the included LICENSE.txt file. When referring to the ooDACE toolbox please make reference to the corresponding publications:

- Blind Kriging: Implementation and performance analysis I. Couckuyt, A. Forrester, D. Gorissen, F. De Turck, T. Dhaene, Advances in Engineering Software, Vol. 49, pp. 1-13, July 2012.
- Surrogate-based infill optimization applied to electromagnetic problems I. Couckuyt, F. Declercq, T. Dhaene, H. Rogier, L. Knockaert, International Journal of RF and Microwave Computer-Aided Engineering (RFMiCA-E), Special Issue on Advances in Design Optimization of Microwave/RF Circuits and Systems, Vol. 20, No. 5, pp. 492-501, September 2010.

Contact : ivo.couckuyt@ugent.be - http://sumo.intec.ugent.be/?q=ooDACE Signature evalgrid = makeEvalGrid(gridpoints, gridsize)

8.95.2 Function Documentation

8.95.2.1 function makeEvalGrid (var *gridpoints,* var *gridsize*)

Low-level procedure, makes a 'prod(gridsize)' by 'length(gridsize)' array, where each row is a different vector, where the i'th element is a number out of 'gridpoints{i}'.

The 'gridsize' parameter may be omitted, in that case it's generated from 'gridpoints'. This function produces grids in the expected order, which is now also used for gridded dataset files. Use this instead of makeEvalGridInverted if possible.

Example: makeEvalGrid({ [-1, .5], [-1 0 1], [.2 .3] }, [2 3 2]) ans = -1.0000 -1.0000 0.2000 -1.0000 -1.0000 0.3000 -1.0000 0.0000 0.2000 -1.0000 0.0000 0.3000 -1.0000 1.0000 0.2000 -1.0000 1.0000 0.3000 0.5000 -1.0000 0.2000 0.5000 -1.0000 0.3000 0.5000 0.0000 0.2000 0.5000 0.0000 0.3000 0.5000 1.0000 0.2000 0.5000 1.0000 0.3000

8.96 /home/ilm/projecten/meta/scripts/ooDACE/tools/makeGrid.m File Reference

Functions

- function makeGrid (var sizes)

  Construct a matrix of size `prod(sizes) by length(sizes)" where the rows represent all gridpoints on a` *sizes" sized grid.*

8.96.1 Detailed Description

Authors

Ivo Couckuyt

Version

1.4 ($Revision$)

Date

$LastChangedDate$
Copyright 2010-2013

This file is part of the ooDACE toolbox and you can redistribute it and/or modify it under the terms of the GNU Affero General Public License version 3 as published by the Free Software Foundation. With the additional provision that a commercial license must be purchased if the ooDACE toolbox is used, modified, or extended in a commercial setting. For details see the included LICENSE.txt file. When referring to the ooDACE toolbox please make reference to the corresponding publications:

- Blind Kriging: Implementation and performance analysis I. Couckuyt, A. Forrester, D. Gorissen, F. De Turck, T. Dhaene, Advances in Engineering Software, Vol. 49, pp. 1-13, July 2012.
- Surrogate-based infill optimization applied to electromagnetic problems I. Couckuyt, F. Declercq, T. Dhaene, H. Rogier, L. Knockaert, International Journal of RF and Microwave Computer-Aided Engineering (RFMiCA-E), Special Issue on Advances in Design Optimization of Microwave/RF Circuits and Systems, Vol. 20, No. 5, pp. 492-501, September 2010.

Contact : ivo.couckuyt@ugent.be - http://sumo.intec.ugent.be/?q=ooDACE Signature g = makeGrid(sizes)

8.96.2 Function Documentation

8.96.2.1 function makeGrid (var *sizes*)

Construct a matrix of size `prod(sizes) by length(sizes)" where the rows represent all gridpoints on a`sizes" sized grid.

Example:

makeGrid([3 1 2 2])

ans = 1 1 1 1 1 1 1 2 1 1 2 1 1 1 2 2 2 1 1 1 2 1 1 2 2 1 2 1 2 1 2 2 3 1 1 1 3 1 1 2 3 1 2 1 3 1 2 2

8.97 /home/ilm/projecten/meta/scripts/ooDACE/tools/mergeStruct.m File Reference

Functions

- function mergeStruct (var s1, var s2, var destFieldExist)
  Copies field of s2 over to s1.

8.97.1 Detailed Description

Authors

Ivo Couckuyt

Version

1.4 ($Revision$)

Date

$LastChangedDate$
Copyright 2010-2013

This file is part of the ooDACE toolbox and you can redistribute it and/or modify it under the terms of the GNU Affero General Public License version 3 as published by the Free Software Foundation. With the additional provision that a commercial license must be purchased if the ooDACE toolbox is used, modified, or extended in a commercial setting. For details see the included LICENSE.txt file. When referring to the ooDACE toolbox please make reference to the corresponding publications:

- Blind Kriging: Implementation and performance analysis I. Couckuyt, A. Forrester, D. Gorissen, F. De Turck, T. Dhaene, Advances in Engineering Software, Vol. 49, pp. 1-13, July 2012.
- Surrogate-based infill optimization applied to electromagnetic problems I. Couckuyt, F. Declercq, T. Dhaene, H. Rogier, L. Knockaert, International Journal of RF and Microwave Computer-Aided Engineering (RFMiCAE), Special Issue on Advances in Design Optimization of Microwave/RF Circuits and Systems, Vol. 20, No. 5, pp. 492-501, September 2010.

Contact : `ivo.couckuyt@ugent.be` - `http://sumo.intec.ugent.be/?q=ooDACE` Signature o = mergeStruct(s1, s2, destFieldExist)

8.97.2 Function Documentation

8.97.2.1 function mergeStruct (var *s1,* var *s2,* var *destFieldExist*)

Copies field of s2 over to s1.

Parameters

s1	destination structure
s2	source structure
destFieldExist	• -1: always copy • false: only copy when destination field does NOT exist • true: only copy when destination field exist

8.98 /home/ilm/projecten/meta/scripts/ooDACE/tools/plotScatteredData.m File Reference

Functions

- function plotScatteredData (var varargin)

  A nice, easy way to create a surf plot of scatterd data.

8.98.1 Detailed Description

Authors

Ivo Couckuyt

Version

1.4 ($Revision$)

Date

$LastChangedDate$
Copyright 2010-2013

This file is part of the ooDACE toolbox and you can redistribute it and/or modify it under the terms of the GNU Affero General Public License version 3 as published by the Free Software Foundation. With the additional provision that a commercial license must be purchased if the ooDACE toolbox is used, modified, or extended in a commercial setting. For details see the included LICENSE.txt file. When referring to the ooDACE toolbox please make reference to the corresponding publications:

- Blind Kriging: Implementation and performance analysis I. Couckuyt, A. Forrester, D. Gorissen, F. De Turck, T. Dhaene, Advances in Engineering Software, Vol. 49, pp. 1-13, July 2012.
- Surrogate-based infill optimization applied to electromagnetic problems I. Couckuyt, F. Declercq, T. Dhaene, H. Rogier, L. Knockaert, International Journal of RF and Microwave Computer-Aided Engineering (RFMiCA-E), Special Issue on Advances in Design Optimization of Microwave/RF Circuits and Systems, Vol. 20, No. 5, pp. 492-501, September 2010.

Contact : ivo.couckuyt@ugent.be - http://sumo.intec.ugent.be/?q=ooDACE Signature h = plotScatteredData(varargin)

8.98.2 Function Documentation

8.98.2.1 function plotScatteredData (var *varargin*)

A nice, easy way to create a surf plot of scatterd data.

8.99 /home/ilm/projecten/meta/scripts/ooDACE/tools/powerBase.m File Reference

Functions

- function powerBase (var x, var d)

  Powers of all the variables as a base for the interpolant.

8.99.1 Detailed Description

Authors

Ivo Couckuyt

Version

1.4 ($Revision$)

Date

$LastChangedDate$
Copyright 2010-2013

This file is part of the ooDACE toolbox and you can redistribute it and/or modify it under the terms of the GNU Affero General Public License version 3 as published by the Free Software Foundation. With the additional provision that a commercial license must be purchased if the ooDACE toolbox is used, modified, or extended in a commercial setting. For details see the included LICENSE.txt file. When referring to the ooDACE toolbox please make reference to the corresponding publications:

- Blind Kriging: Implementation and performance analysis I. Couckuyt, A. Forrester, D. Gorissen, F. De Turck, T. Dhaene, Advances in Engineering Software, Vol. 49, pp. 1-13, July 2012.
- Surrogate-based infill optimization applied to electromagnetic problems I. Couckuyt, F. Declercq, T. Dhaene, H. Rogier, L. Knockaert, International Journal of RF and Microwave Computer-Aided Engineering (RFMiCA-E), Special Issue on Advances in Design Optimization of Microwave/RF Circuits and Systems, Vol. 20, No. 5, pp. 492-501, September 2010.

Contact : ivo.couckuyt@ugent.be - http://sumo.intec.ugent.be/?q=ooDACE Signature base = powerBase(x, d)

8.99.2 Function Documentation

8.99.2.1 function powerBase (var *x*, var *d*)

Powers of all the variables as a base for the interpolant.

Index