(S)NOWPAC

Generated by Doxygen 1.8.11

Contents

1 Hierarchical Index 1

1.1 Class Hierarchy 1

2 Class Index 3

2.1 Class List 3

3 Class Documentation 5

3.1 BasisForMinimumFrobeniusNormModel Class Reference 5

3.2 BasisForSurrogateModelBaseClass Class Reference 5

3.2.1 Detailed Description 6

3.2.2 Constructor & Destructor Documentation 6

3.2.2.1 BasisForSurrogateModelBaseClass(int n) 6

3.2.3 Member Data Documentation 6

3.2.3.1 nb_basis_functions 6

3.3 BlackBoxBaseClass Class Reference 7

3.4 BlackboxData Struct Reference 7

3.5 CholeskyFactorization Class Reference 7

3.5.1 Detailed Description 8

3.5.2 Member Function Documentation 8

3.5.2.1 compute(Eigen::MatrixXd &, int &, double &, int) 8

3.6 GaussianProcess Class Reference 8

3.6.1 Detailed Description 9

3.6.2 Constructor & Destructor Documentation 9

3.6.2.1 GaussianProcess(int) 9

3.6.3 Member Function Documentation 9

3.6.3.1 build(std::vector< std::vector< double > > const &, Eigen::VectorXd const &, Eigen::VectorXd const &) 9

3.6.3.2 estimate_hyper_parameters(std::vector< std::vector< double > > const &, Eigen::VectorXd const &, Eigen::VectorXd const &) 10

3.6.3.3 evaluate(std::vector< double > const &, double &, double &) 10

3.6.3.4 evaluate_kernel(std::vector< double > const &, std::vector< double > const &) . 10

3.6.3.5 update(std::vector< double > const &, double &, double &) 11

3.7 GaussianProcessBaseClass Class Reference 11

3.7.1 Detailed Description 11

3.8 GaussianProcessKernelBaseClass Class Reference 12

3.8.1 Detailed Description 12

3.9 GaussianProcessSupport Class Reference 12

3.10 ImprovePoisedness Class Reference 12

3.10.1 Detailed Description 13

3.10.2 Constructor & Destructor Documentation 13

3.10.2.1 ImprovePoisedness(BasisForSurrogateModelBaseClass &, double, int, double &, int) 13

3.10.3 Member Function Documentation 13

3.10.3.1 improve_poisedness(int, BlackboxData &) 13

3.10.3.2 replace_node(int, BlackboxData const &, std::vector< double > const &) 14

3.11 ImprovePoisednessBaseClass Class Reference 14

3.12 MinimumFrobeniusNormModel Class Reference 15

3.13 NOWPAC< TSurrogateModel, TBasisForSurrogateModel > Class Template Reference 15

3.14 QuadraticMinimization Class Reference 16

3.14.1 Detailed Description 16

3.14.2 Constructor & Destructor Documentation 16

3.14.2.1 QuadraticMinimization(int) 16

3.14.3 Member Function Documentation 16

3.14.3.1 minimize(Eigen::VectorXd &, Eigen::VectorXd const &, Eigen::MatrixXd const &) 16

3.15 QuadraticMonomial Class Reference 17

3.15.1 Detailed Description 17

3.15.2 Constructor & Destructor Documentation 17
3.15.2.1 QuadraticMonomial(int dim_input) 17
3.15.3 Member Function Documentation 18
3.15.3.1 evaluate_monomial(int, std::vector< double > const &) 18
3.16 RegularizedMinimumFrobeniusNormModel Class Reference 18
3.17 SubproblemData< TSurrogateModel, TSubproblemOptimization > Struct Template Reference . . 19
3.18 SubproblemDefinitions< TSurrogateModel, TSubproblemOptimization > Class Template Reference 19
3.19 SubproblemOptimization< TSurrogateModel > Class Template Reference 19
3.20 SurrogateModelBaseClass Class Reference 20
3.21 TriangularMatrixOperations Class Reference 20
3.21.1 Detailed Description 21
3.21.2 Constructor & Destructor Documentation 21
3.21.2.1 TriangularMatrixOperations(int) 21
3.21.3 Member Function Documentation 21
3.21.3.1 backward_substitution(Eigen::MatrixXd const &, Eigen::VectorXd &) 21
3.21.3.2 compute_large_norm_solution(Eigen::MatrixXd const &, Eigen::VectorXd &) . . . 22
3.21.3.3 forward_substitution(Eigen::MatrixXd const &, Eigen::VectorXd &) 22
3.22 VectorOperations Class Reference 22
3.22.1 Detailed Description 23
3.22.2 Member Function Documentation 23
3.22.2.1 add(double, std::vector< double > const &, std::vector< double > &) 23
3.22.2.2 diff_norm(std::vector< double > const &, std::vector< double > const &) 23
3.22.2.3 dot_product(std::vector< double > const &, std::vector< double > const &) . . . 23
3.22.2.4 minus(std::vector< double > const &, std::vector< double > const &, std::vector< double > &) 24
3.22.2.5 rescale(double, std::vector< double > const &, std::vector< double > const &, std::vector< double > &) 24
3.22.2.6 scale(double, std::vector< double > const &, std::vector< double > &) 24
3.22.2.7 set_zero(std::vector< double > &) 25

Index 27

Chapter 1

Hierarchical Index

1.1 Class Hierarchy

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

- BasisForSurrogateModelBaseClass . . . 5
  - BasisForMinimumFrobeniusNormModel . . . 5
- BlackBoxBaseClass . . . 7
- BlackboxData . . . 7
- CholeskyFactorization . . . 7
  - GaussianProcess . . . 8
  - QuadraticMinimization . . . 16
    - ImprovePoisedness . . . 12
- GaussianProcessKernelBaseClass . . . 12
  - GaussianProcessBaseClass . . . 11
    - GaussianProcess . . . 8
- ImprovePoisednessBaseClass . . . 14
  - ImprovePoisedness . . . 12
- QuadraticMonomial . . . 17
  - BasisForMinimumFrobeniusNormModel . . . 5
- SubproblemData< TSurrogateModel, TSubproblemOptimization > . . . 19
- SurrogateModelBaseClass . . . 20
  - MinimumFrobeniusNormModel . . . 15
  - RegularizedMinimumFrobeniusNormModel . . . 18
- TriangularMatrixOperations . . . 20
  - GaussianProcess . . . 8
  - QuadraticMinimization . . . 16
- VectorOperations . . . 22
  - BasisForMinimumFrobeniusNormModel . . . 5
  - GaussianProcessSupport . . . 12
  - MinimumFrobeniusNormModel . . . 15
  - NOWPAC< TSurrogateModel, TBasisForSurrogateModel > . . . 15
  - QuadraticMinimization . . . 16
  - RegularizedMinimumFrobeniusNormModel . . . 18
  - SubproblemDefinitions< TSurrogateModel, TSubproblemOptimization > . . . 19
  - SubproblemDefinitions< TSurrogateModel, SubproblemOptimization > . . . 19
  - SubproblemOptimization< TSurrogateModel > . . . 19

Chapter 2

Class Index

2.1 Class List

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

BasisForMinimumFrobeniusNormModel . . . 5

BasisForSurrogateModelBaseClass

Base class for definiton of surrogate model . . . 5

BlackBoxBaseClass . . . 7

BlackboxData . . . 7

CholeskyFactorization

Cholesky factorization . . . 7

GaussianProcess

Gaussian process regression . . . 8

GaussianProcessBaseClass

Gaussian process regression . . . 11

GaussianProcessKernelBaseClass

Interface for Gaussian kernel defintion . . . 12

GaussianProcessSupport . . . 12

ImprovePoisedness

Improve poisedness of interpolation nodes . . . 12

ImprovePoisednessBaseClass . . . 14

MinimumFrobeniusNormModel . . . 15

NOWPAC< TSurrogateModel, TBasisForSurrogateModel > . . . 15

QuadraticMinimization

Quadratic minimization in unit ball . . . 16

QuadraticMonomial

Quadratic monomials . . . 17

RegularizedMinimumFrobeniusNormModel . . . 18

SubproblemData< TSurrogateModel, TSubproblemOptimization > . . . 19

SubproblemDefinitions< TSurrogateModel, TSubproblemOptimization > . . . 19

SubproblemOptimization< TSurrogateModel > . . . 19

SurrogateModelBaseClass . . . 20

TriangularMatrixOperations

Forward and backward substiutions with lower triangular matrices . . . 20

VectorOperations

Vector operations . . . 22

Chapter 3

Class Documentation

3.1 BasisForMinimumFrobeniusNormModel Class Reference

Public Member Functions

- **BasisForMinimumFrobeniusNormModel** (int, double &)
- void **set_nb_nodes** (int)
- std::vector< double > & **evaluate** (std::vector< double > const &)
- double **evaluate** (std::vector< double > const &, int)
- std::vector< double > & **gradient** (std::vector< double > const &, int)
- void **compute_basis_coefficients** (BlackboxData const &)
- void **get_mat_vec_representation** (int, Eigen::VectorXd &, Eigen::MatrixXd &)
- void **compute_mat_vec_representation** (int)

Additional Inherited Members

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

- /Users/Florian/home/sandbox/surrogate_models_cpp/include/BasisForMinimumFrobeniusNormModel.hpp
- /Users/Florian/home/sandbox/surrogate_models_cpp/src/BasisForMinimumFrobeniusNormModel.cpp

3.2 BasisForSurrogateModelBaseClass Class Reference

Base class for definiton of surrogate model.

```
#include <BasisForSurrogateModelBaseClass.hpp>
```

Public Member Functions

- BasisForSurrogateModelBaseClass (int n)

  Constructor.
- ∼BasisForSurrogateModelBaseClass ()

  Destructor.
- virtual void **get_mat_vec_representation** (int, Eigen::VectorXd &, Eigen::MatrixXd &)=0
- virtual void **compute_basis_coefficients** (BlackboxData const &)=0
- virtual std::vector< double > & **evaluate** (std::vector< double > const &)=0
- virtual std::vector< double > & **gradient** (std::vector< double > const &, int)=0
- virtual double **evaluate** (std::vector< double > const &, int)=0
- int **dimension** ()

Protected Attributes

- int dim

  Number of arguments of surrogate model.
- int nb_basis_functions

  Number of basic basis functions.
- std::vector< Eigen::VectorXd > basis_coefficients

  Coefficients of surrogate basis functions in terms of basic basis function.

3.2.1 Detailed Description

Base class for definiton of surrogate model.

Defines the required structure of surrogate models to work with NOWPAC

3.2.2 Constructor & Destructor Documentation

3.2.2.1 BasisForSurrogateModelBaseClass::BasisForSurrogateModelBaseClass (int *n*) [inline]

Constructor.

Constructor to set number of arguments (dimension) of the basis

Parameters

dim_input	Number of arguments (dimension)

3.2.3 Member Data Documentation

3.2.3.1 int BasisForSurrogateModelBaseClass::nb_basis_functions [protected]

Number of basic basis functions.

Number of basis function, for example quadratic monomials, for surrogate basis functions

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

- /Users/Florian/home/sandbox/surrogate_models_cpp/include/BasisForSurrogateModelBaseClass.hpp

3.3 BlackBoxBaseClass Class Reference

Public Member Functions

- virtual void **evaluate** (std::vector< double > const &x, std::vector< double > &vals, void ∗param)
- virtual void **evaluate** (std::vector< double > const &x, std::vector< double > &vals, std::vector< double > &noise, void ∗param)

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

- /Users/Florian/home/sandbox/surrogate_models_cpp/include/BlackBoxBaseClass.hpp

3.4 BlackboxData Struct Reference

Public Attributes

- int **max_nb_nodes**
- int **best_index**
- std::vector< std::vector< double > > **nodes**
- std::vector< std::vector< double > > **values**
- std::vector< std::vector< double > > **noise**
- std::vector< int > **surrogate_nodes_index**

The documentation for this struct was generated from the following file:

- /Users/Florian/home/sandbox/surrogate_models_cpp/include/BlackBoxData.hpp

3.5 CholeskyFactorization Class Reference

Cholesky factorization.

```
#include <CholeskyFactorization.hpp>
```

Protected Member Functions

- void compute (Eigen::MatrixXd &, int &, double &, int)

  Computes Cholesky factorization.

3.5.1 Detailed Description

Cholesky factorization.

Computes the Cholesky factorization of a matrix M if M is positive definite, i.e. M = LL' with a lower triangular matrix L. If M is positive defineite the matrix M is over-written by L and p and offset are set to zero.
If M is not positive definite with a zero eigenvalue, a partial Cholesky decomposition in the upper left (p-1)x(p-1) block of M is returned.
Additionally, p and offset are set such that M + rho ep ep' (with the p-th canonical unit vector ep) has a zero eigenvalue.

3.5.2 Member Function Documentation

3.5.2.1 void CholeskyFactorization::compute (Eigen::MatrixXd & *L*, int & *p*, double & *offset*, int *n*) `[protected]`

Computes Cholesky factorization.

Parameters

L	matrix M to be factorizied on input and the factorized matrix L on output
p	index of the diagonal element such that M+offset ep ep^T has a zero eigenvalue
offset	offset to shift non-positive eigenvalue of M to zero.
n	dimension of the matrix L

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

- /Users/Florian/home/sandbox/surrogate_models_cpp/include/CholeskyFactorization.hpp
- /Users/Florian/home/sandbox/surrogate_models_cpp/src/CholeskyFactorization.cpp

3.6 GaussianProcess Class Reference

Gaussian process regression.

`#include <GaussianProcess.hpp>`

Public Member Functions

- GaussianProcess (int)

  Constructor.
- ∼GaussianProcess ()

  Destructor.
- void estimate_hyper_parameters (std::vector< std::vector< double > > const &, Eigen::VectorXd const &, Eigen::VectorXd const &)

  Estimation of hyper parameters.
- void build (std::vector< std::vector< double > > const &, Eigen::VectorXd const &, Eigen::VectorXd const &)

  Build the Gaussian process.
- void update (std::vector< double > const &, double &, double &)

  Update the Gaussian process.
- void evaluate (std::vector< double > const &, double &, double &)

  Evaluate Gaussian process.

Protected Member Functions

- double evaluate_kernel (std::vector< double > const &, std::vector< double > const &)

  Evaluation of Gaussian process kernel.
- double d_evaluate_kernel (std::vector< double > const &, std::vector< double > const &, int)

  Evaluation of the derivative of the Gaussina process kernel.

3.6.1 Detailed Description

Gaussian process regression.

Computes a Gaussian process of given data points, function evaluations and noise estimates.

See also

GaussianProcessBaseClass
CholeskyFactorization
TriangularMatrixOperations

3.6.2 Constructor & Destructor Documentation

3.6.2.1 GaussianProcess::GaussianProcess (int *n*)

Constructor.

Class constructor.

Parameters

n	dimension of the Gaussian process.

3.6.3 Member Function Documentation

3.6.3.1 void GaussianProcess::build (std::vector< std::vector< double > > const & *nodes,* Eigen::VectorXd const & *values,* Eigen::VectorXd const & *noise*) [virtual]

Build the Gaussian process.

Computes the Gaussian process
Requires the estimation of hyper parameters

Parameters

nodes	regression points
function	values
noise	in function values

See also

estimate_hyper_parameters

Implements GaussianProcessBaseClass.

3.6.3.2 void GaussianProcess::estimate_hyper_parameters (std::vector< std::vector< double > > const & *nodes*, Eigen::VectorXd const & *values*, Eigen::VectorXd const & *noise*) [virtual]

Estimation of hyper parameters.

Estimates the hyper parameters of the Gaussian process.
The hyper parameters are the variance and the length scale parameters in the exponential kernel.

Parameters

nodes	regression points
function	values
noise	in function values

Implements GaussianProcessKernelBaseClass.

3.6.3.3 void GaussianProcess::evaluate (std::vector< double > const & *x*, double & *mean*, double & *variance*) [virtual]

Evaluate Gaussian process.

Computes the mean and variance of the Gaussian process.
Requires the building of the Gaussian process.

Parameters

x	point at which the Gaussian process is evaluated
mean	mean of the Gaussian process at point x
variance	variance of the Gaussina process at point x

See also

build

Implements GaussianProcessBaseClass.

3.6.3.4 double GaussianProcess::evaluate_kernel (std::vector< double > const & *x*, std::vector< double > const & *y*) [protected], [virtual]

Evaluation of Gaussian process kernel.

Evaluates the square exponential kernel.

Implements GaussianProcessKernelBaseClass.

3.6.3.5 void GaussianProcess::update (std::vector< double > const & *x*, double & *value*, double & *noise*) [virtual]

Update the Gaussian process.

Includees a new point into the Gaussian process

Parameters

x	new point to be included into the Gaussian process
value	new function value at new point
noise	new noise estimate at new function value

Implements GaussianProcessBaseClass.

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

- /Users/Florian/home/sandbox/surrogate_models_cpp/include/GaussianProcess.hpp
- /Users/Florian/home/sandbox/surrogate_models_cpp/src/GaussianProcess.cpp

3.7 GaussianProcessBaseClass Class Reference

Gaussian process regression.

```
#include <GaussianProcessBaseClass.hpp>
```

Public Member Functions

- GaussianProcessBaseClass ()

  Constructor.
- ∼GaussianProcessBaseClass ()

  Destructor.
- virtual void build (std::vector< std::vector< double > > const &, Eigen::VectorXd const &, Eigen::VectorXd const &)=0

  Build the Gaussian process.
- virtual void update (std::vector< double > const &, double &, double &)=0

  Update the Gaussian process.
- virtual void evaluate (std::vector< double > const &, double &, double &)=0

  Evaluate Gaussian process.

3.7.1 Detailed Description

Gaussian process regression.

Interface for Gaussian process regression

See also

GaussianProcessKernelBaseClasss

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

- /Users/Florian/home/sandbox/surrogate_models_cpp/include/GaussianProcessBaseClass.hpp

3.8 GaussianProcessKernelBaseClass Class Reference

Interface for Gaussian kernel defintion.

```
#include <GaussianProcessKernelBaseClass.hpp>
```

Public Member Functions

- virtual void estimate_hyper_parameters (std::vector< std::vector< double > > const &, Eigen::VectorXd const &, Eigen::VectorXd const &)=0

  Virtual member function for the estimation of hyper parameters of the kernel.
- virtual double evaluate_kernel (std::vector< double > const &, std::vector< double > const &)=0

  Virtual membber function for the evaluation of the kernel.

3.8.1 Detailed Description

Interface for Gaussian kernel defintion.

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

- /Users/Florian/home/sandbox/surrogate_models_cpp/include/GaussianProcessKernelBaseClass.hpp

3.9 GaussianProcessSupport Class Reference

Public Member Functions

- void **initialize** (const int, const int, double &, Eigen::VectorXd const &, int)
- void **smooth_data** (BlackboxData &)

Additional Inherited Members

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

- /Users/Florian/home/sandbox/surrogate_models_cpp/include/GaussianProcessSupport.hpp
- /Users/Florian/home/sandbox/surrogate_models_cpp/src/GaussianProcessSupport.cpp

3.10 ImprovePoisedness Class Reference

Improve poisedness of interpolation nodes.

```
#include <ImprovePoisedness.hpp>
```

Public Member Functions

- ImprovePoisedness (BasisForSurrogateModelBaseClass &, double, int, double &, int)

  Constructor.
- ∼ImprovePoisedness ()

  Destructor.
- int replace_node (int, BlackboxData const &, std::vector< double > const &)

  Find node to be replaced by better poised node.
- void improve_poisedness (int, BlackboxData &)

  Improves poisedness of interpolation nodes.

Additional Inherited Members

3.10.1 Detailed Description

Improve poisedness of interpolation nodes.

3.10.2 Constructor & Destructor Documentation

3.10.2.1 ImprovePoisedness::ImprovePoisedness (BasisForSurrogateModelBaseClass & *B*, double *poisedness_threshold*, int *m*, double & *rad*, int *verbose*)

Constructor.

Set parameters required for the improvement of the poisedness of interploation nodes

Parameters

B	basis for surrogate model
poisedness_threshold	threshold for poisedness constant
m	maximal number of interpolation nodes
rad	radius arround current best point of ball that contains well poised points
verbose	switch output on (verbose = 3) or off (verbose = 0)

See also

BlackboxData

3.10.3 Member Function Documentation

3.10.3.1 void ImprovePoisedness::improve_poisedness (int *reference_node*, BlackboxData & *evaluations*) [virtual]

Improves poisedness of interpolation nodes.

Improves poisedness of interpolation nodes by maximizing the absolute value of basis functions. Nodes to replace existing interpolation nodes are computed and appended to the list of nodes,

See also

BlackboxData
The index of nodes to reduce the poisedness value are indicated in evaluations
BlackboxData

Parameters

reference_node	index of node that is not replaced
evaluations	structure containing interpolation nodes,

See also

BlackboxData

Implements ImprovePoisednessBaseClass.

3.10.3.2 int ImprovePoisedness::replace_node (int *reference_node*, BlackboxData const & *evaluations*, std::vector< double > const & *new_node*) [virtual]

Find node to be replaced by better poised node.

Finds a node to replace another interpolation node to improve poisedness

Parameters

reference_node	index of node that is not replaced
evaluations	interpolation nodes,

See also

BlackboxData

Parameters

new_node	new node to replace an existing interpolation node

Implements ImprovePoisednessBaseClass.

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

- /Users/Florian/home/sandbox/surrogate_models_cpp/include/ImprovePoisedness.hpp
- /Users/Florian/home/sandbox/surrogate_models_cpp/src/ImprovePoisedness.cpp

3.11 ImprovePoisednessBaseClass Class Reference

Public Member Functions

- **ImprovePoisednessBaseClass** (double threshold_for_poisedness_constant_input, BasisForSurrogate↩ ModelBaseClass &basis_input)
- virtual int **replace_node** (int, BlackboxData const &, std::vector< double > const &)=0
- virtual void **improve_poisedness** (int, BlackboxData &)=0

Protected Attributes

- BasisForSurrogateModelBaseClass ∗ **basis**
- double **poisedness_constant**
- double **threshold_for_poisedness_constant**
- std::vector< bool > **index_of_changed_nodes**

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

- /Users/Florian/home/sandbox/surrogate_models_cpp/include/ImprovePoisednessBaseClass.hpp

3.12 MinimumFrobeniusNormModel Class Reference

Public Member Functions

- **MinimumFrobeniusNormModel** (BasisForMinimumFrobeniusNormModel &)
- double **evaluate** (std::vector< double > const &)
- std::vector< double > & **gradient** (std::vector< double > const &)
- void **set_function_values** (std::vector< double > const &, std::vector< double > const &, std::vector< int > const &)

Additional Inherited Members

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

- /Users/Florian/home/sandbox/surrogate_models_cpp/include/MinimumFrobeniusNormModel.hpp
- /Users/Florian/home/sandbox/surrogate_models_cpp/src/MinimumFrobeniusNormModel.cpp

3.13 NOWPAC< TSurrogateModel, TBasisForSurrogateModel > Class Template Reference

Public Member Functions

- **NOWPAC** (int)
- void **set_blackbox** (BlackBoxBaseClass &, int)
- void **set_blackbox** (BlackBoxBaseClass &)
- int **optimize** (std::vector< double > &, double &)
- void **set_option** (std::string const &, int const &)
- void **set_option** (std::string const &, double const &)
- void **set_option** (std::string const &, bool const &)
- void **set_option** (std::string const &, Eigen::VectorXd const &)
- void **void_user_data** (void ∗)
- void **set_lower_bounds** (Eigen::VectorXd const &)
- void **set_upper_bounds** (Eigen::VectorXd const &)
- void **set_trustregion** (double const &)
- void **set_trustregion** (double const &, double const &)
- void **set_max_trustregion** (double const &)
- void **set_max_number_evaluations** (int const &)

Additional Inherited Members

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

- /Users/Florian/home/sandbox/surrogate_models_cpp/src/NOWPAC.cpp

3.14 QuadraticMinimization Class Reference

Quadratic minimization in unit ball.

```
#include <QuadraticMinimization.hpp>
```

Public Member Functions

- QuadraticMinimization (int)

  Constructor.
- ∼QuadraticMinimization ()

  Destructor.
- void minimize (Eigen::VectorXd &, Eigen::VectorXd const &, Eigen::MatrixXd const &)

  Solve quadratic minimization in unit ball.

Additional Inherited Members

3.14.1 Detailed Description

Quadratic minimization in unit ball.

3.14.2 Constructor & Destructor Documentation

3.14.2.1 QuadraticMinimization::QuadraticMinimization (int *n*)

Constructor.

Contructor to set dimension of quadratic optimizatin problem

Parameters

n	dimension of quadratic optimization problem

3.14.3 Member Function Documentation

3.14.3.1 void QuadraticMinimization::minimize (Eigen::VectorXd & *y*, Eigen::VectorXd const & *g*, Eigen::MatrixXd const & *H*)

Solve quadratic minimization in unit ball.

Solves the quadratic minimization y = argmin g'x + 0.5x'Hx subject to ||x|| <= 1.
Implementation of algorithm from More/Sorensen, Computing a trust region step (1983).

Parameters

y	solution of quadratic optimization problem
g	gradient of quadratic opjective function
H	hessian of quadratic objective function

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

- /Users/Florian/home/sandbox/surrogate_models_cpp/include/QuadraticMinimization.hpp
- /Users/Florian/home/sandbox/surrogate_models_cpp/src/QuadraticMinimization.cpp

3.15 QuadraticMonomial Class Reference

Quadratic monomials.

```
#include <QuadraticMonomial.hpp>
```

Public Member Functions

- QuadraticMonomial (int dim_input)

  Constructor.

Protected Member Functions

- double evaluate_monomial (int, std::vector< double > const &)

  Evaluation of monomials.

3.15.1 Detailed Description

Quadratic monomials.

Evaluates quadratic monomials in dim dimensions. Monimial 0 = 1 Monomial 1 ... dim = x_i Monomial dim+1 ... 2*dim = 0.5*x_i^2 Monomial 2*dim+1 ... 3*dim = x_1*x_2 ... x_1*x_n Monomial 3*dim+1 ... 4*dim-1 = x_2*x_3 ... x_2*x_n ... Monoial (dim^2 + 3*dim +2)/2 = x_{n-1}x_n

3.15.2 Constructor & Destructor Documentation

3.15.2.1 QuadraticMonomial::QuadraticMonomial (int *dim_input*) [inline]

Constructor.

Parameters

dim_input	Dimension of monomials

3.15.3 Member Function Documentation

3.15.3.1 double QuadraticMonomial::evaluate_monomial (int *basis_number,* std::vector< double > const & *x*) [protected]

Evaluation of monomials.

Evaluation of monomial number p

Parameters

p	Number of monomial as discribed
x	Point where monomial is evaluated

See also

QuadraticMonomial

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

- /Users/Florian/home/sandbox/surrogate_models_cpp/include/QuadraticMonomial.hpp
- /Users/Florian/home/sandbox/surrogate_models_cpp/src/QuadraticMonomial.cpp

3.16 RegularizedMinimumFrobeniusNormModel Class Reference

Public Member Functions

- **RegularizedMinimumFrobeniusNormModel** (BasisForMinimumFrobeniusNormModel &)
- double **evaluate** (std::vector< double > const &)
- std::vector< double > & **gradient** (std::vector< double > const &)
- void **set_function_values** (std::vector< double > const &, std::vector< double > const &, std::vector< int > const &)

Static Public Member Functions

- static double **regularization_objective** (std::vector< double > const &, std::vector< double > &, void ∗)

Additional Inherited Members

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

- /Users/Florian/home/sandbox/surrogate_models_cpp/include/RegularizedMinimumFrobeniusNormModel.↩ hpp
- /Users/Florian/home/sandbox/surrogate_models_cpp/src/RegularizedMinimumFrobeniusNormModel.cpp

3.17 SubproblemData< TSurrogateModel, TSubproblemOptimization > Struct Template Reference

Public Attributes

- TSubproblemOptimization< TSurrogateModel > ∗ **me**
- VectorOperations ∗ **vo**
- int **constraint_number**

The documentation for this struct was generated from the following file:

- /Users/Florian/home/sandbox/surrogate_models_cpp/include/SubproblemDefinitions.hpp

3.18 SubproblemDefinitions< TSurrogateModel, TSubproblemOptimization > Class Template Reference

Static Public Member Functions

- static double **opt_trial_point_obj** (std::vector< double > const &, std::vector< double > &, void ∗)
- static double **opt_criticality_measure_obj** (std::vector< double > const &, std::vector< double > &, void ∗)
- static double **opt_restore_feasibility_obj** (std::vector< double > const &, std::vector< double > &, void ∗)
- static double **trustregion_constraint** (std::vector< double > const &, std::vector< double > &, void ∗)
- static double **constraints_for_subproblems** (std::vector< double > const &, std::vector< double > &, void ∗)

Additional Inherited Members

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

- /Users/Florian/home/sandbox/surrogate_models_cpp/include/SubproblemDefinitions.hpp

3.19 SubproblemOptimization< TSurrogateModel > Class Template Reference

Public Member Functions

- **SubproblemOptimization** (std::vector< TSurrogateModel > &, double &, Eigen::VectorXd &)
- double **compute_criticality_measure** (std::vector< double > &)
- double **compute_trial_point** (std::vector< double > &)
- double **restore_feasibility** (std::vector< double > &)
- void **set_lower_bounds** (Eigen::VectorXd &)
- void **set_upper_bounds** (Eigen::VectorXd &)

Public Attributes

- VectorOperations **vo**
- std::vector< double > **best_point**
- Eigen::VectorXd ∗ **inner_boundary_constant**
- double ∗ **delta**
- Eigen::VectorXd **feasibility_thresholds**
- std::vector< TSurrogateModel > ∗ **surrogate_models**
- std::vector< double > **criticality_gradient**

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

- /Users/Florian/home/sandbox/surrogate_models_cpp/include/SubproblemOptimization.hpp

3.20 SurrogateModelBaseClass Class Reference

Public Member Functions

- **SurrogateModelBaseClass** (BasisForSurrogateModelBaseClass &basis_input)
- virtual double **evaluate** (std::vector< double > const &)=0
- virtual std::vector< double > & **gradient** (std::vector< double > const &)=0
- virtual void **set_function_values** (std::vector< double > const &, std::vector< double > const &, std::vector< int > const &)=0
- int **dimension** ()

Protected Attributes

- std::vector< double > **model_gradient**
- std::vector< double > **function_values**
- BasisForSurrogateModelBaseClass ∗ **basis**

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

- /Users/Florian/home/sandbox/surrogate_models_cpp/include/SurrogateModelBaseClass.hpp

3.21 TriangularMatrixOperations Class Reference

Forward and backward substiutions with lower triangular matrices.

```
#include <TriangularMatrixOperations.hpp>
```

Public Member Functions

- TriangularMatrixOperations (int)

  Constructor.
- ∼TriangularMatrixOperations ()

  Destructor.
- void forward_substitution (Eigen::MatrixXd const &, Eigen::VectorXd &)

  Forward substituion.
- void backward_substitution (Eigen::MatrixXd const &, Eigen::VectorXd &)

  Backward substituion.
- void compute_large_norm_solution (Eigen::MatrixXd const &, Eigen::VectorXd &)

  Look behind algorithm (Cline et al. 1982)

3.21.1 Detailed Description

Forward and backward substiutions with lower triangular matrices.

Forward and backward substituion with a lower triangular system matrix. The vector of the right-hand side is overwritten with the solution of the linear system.

3.21.2 Constructor & Destructor Documentation

3.21.2.1 TriangularMatrixOperations::TriangularMatrixOperations (int *n*)

Constructor.

Set dimension of linear systems

Parameters

n	dimension of linear systems

3.21.3 Member Function Documentation

3.21.3.1 void TriangularMatrixOperations::backward_substitution (Eigen::MatrixXd const & *L*, Eigen::VectorXd & *x*)

Backward substituion.

Solves the linear system L'y = x for a lower triangular matrix L The input vextor x contains the right-hand side on input and the solution on output

Parameters

L	Lower triangular matrix
x	Vector of right-hand side on input, solution vector on output

3.21.3.2 void TriangularMatrixOperations::compute_large_norm_solution (Eigen::MatrixXd const & *L*, Eigen::VectorXd & *y*)

Look behind algorithm (Cline et al. 1982)

Computes an approximation of the largest norm solution y of the linear system Ly = p for a vector $||p|| = 1$. Algorithm implemented from Cline et al., Generalizing the LINPACK condition estimator, 1982.

Parameters

L	lower triangular matrix
y	on output the solution an approximation to the largest norm solution

3.21.3.3 void TriangularMatrixOperations::forward_substitution (Eigen::MatrixXd const & *L*, Eigen::VectorXd & *x*)

Forward substituion.

Solves the linear system Ly = x for a lower triangular matrix L The input vextor x contains the right-hand side on input and the solution on output

Parameters

L	Lower triangular matrix
x	Vector of right-hand side on input, solution vector on output

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

- /Users/Florian/home/sandbox/surrogate_models_cpp/include/TriangularMatrixOperations.hpp
- /Users/Florian/home/sandbox/surrogate_models_cpp/src/TriangularMatrixOperations.cpp

3.22 VectorOperations Class Reference

Vector operations.

```
#include <VectorOperations.hpp>
```

Public Member Functions

- void set_zero (std::vector< double > &)

  Setting vector to zero.
- void scale (double, std::vector< double > const &, std::vector< double > &)

  Scaling of vector.
- void add (double, std::vector< double > const &, std::vector< double > &)

  Adding scaled vector.
- void minus (std::vector< double > const &, std::vector< double > const &, std::vector< double > &)

  Substracting two vectors.
- void rescale (double, std::vector< double > const &, std::vector< double > const &, std::vector< double > &)

Rescaling and shifting vector.

- double diff_norm (std::vector< double > const &, std::vector< double > const &)

  Norm of difference of vectors.

- double dot_product (std::vector< double > const &, std::vector< double > const &)

  Dot product of two vectors.

3.22.1 Detailed Description

Vector operations.

3.22.2 Member Function Documentation

3.22.2.1 void VectorOperations::add (double *s*, std::vector< double > const & *v*, std::vector< double > & *w*)

Adding scaled vector.

Computes w = w + s v

Parameters

s	scaling factor
v	input vector
w	input and output vector

3.22.2.2 double VectorOperations::diff_norm (std::vector< double > const & *v1*, std::vector< double > const & *v2*)

Norm of difference of vectors.

Computes the 2 norm of the difference of two vectors

Parameters

v1	input vector
v2	input vector

Returns

norm of v1-v2

3.22.2.3 double VectorOperations::dot_product (std::vector< double > const & *v1*, std::vector< double > const & *v2*)

Dot product of two vectors.

Computes the dot product of two vectors

Parameters

v1	input vector
v2	input vector

Returns

dot product of v1 and v2

3.22.2.4 void VectorOperations::minus (std::vector< double > const & *v1*, std::vector< double > const & *v2*, std::vector< double > & *w*)

Substracting two vectors.

Computes w = v1 - v2

Parameters

v1	input vector
v2	input vector
output	vector

3.22.2.5 void VectorOperations::rescale (double *s*, std::vector< double > const & *v1*, std::vector< double > const & *v2*, std::vector< double > & *w*)

Rescaling and shifting vector.

Computes w = (v1 - v2)s

Parameters

s	scaling factor
v1	vector to be rescaled and shifted
v2	reference vector
w	output vector

3.22.2.6 void VectorOperations::scale (double *s*, std::vector< double > const & *v*, std::vector< double > & *w*)

Scaling of vector.

Computes w = s v

Parameters

s	scaling factor
v	input vector to be scaled
w	scaled vector s v

3.22.2.7 void VectorOperations::set_zero (std::vector< double > & *v*)

Setting vector to zero.

Sets vector v to zero

Parameters

v	on output a vector with elements zero

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

- /Users/Florian/home/sandbox/surrogate_models_cpp/include/VectorOperations.hpp
- /Users/Florian/home/sandbox/surrogate_models_cpp/src/VectorOperations.cpp

Index

add
    VectorOperations, 23

backward_substitution
    TriangularMatrixOperations, 21
BasisForMinimumFrobeniusNormModel, 5
BasisForSurrogateModelBaseClass, 5
    BasisForSurrogateModelBaseClass, 6
    nb_basis_functions, 6
BlackBoxBaseClass, 7
BlackboxData, 7
build
    GaussianProcess, 9

CholeskyFactorization, 7
    compute, 8
compute
    CholeskyFactorization, 8
compute_large_norm_solution
    TriangularMatrixOperations, 21

diff_norm
    VectorOperations, 23
dot_product
    VectorOperations, 23

estimate_hyper_parameters
    GaussianProcess, 10
evaluate
    GaussianProcess, 10
evaluate_kernel
    GaussianProcess, 10
evaluate_monomial
    QuadraticMonomial, 18

forward_substitution
    TriangularMatrixOperations, 22

GaussianProcess, 8
    build, 9
    estimate_hyper_parameters, 10
    evaluate, 10
    evaluate_kernel, 10
    GaussianProcess, 9
    update, 10
GaussianProcessBaseClass, 11
GaussianProcessKernelBaseClass, 12
GaussianProcessSupport, 12

improve_poisedness
    ImprovePoisedness, 13
ImprovePoisedness, 12
    improve_poisedness, 13
    ImprovePoisedness, 13
    replace_node, 14
ImprovePoisednessBaseClass, 14

minimize
    QuadraticMinimization, 16
MinimumFrobeniusNormModel, 15
minus
    VectorOperations, 24

NOWPAC< TSurrogateModel, TBasisForSurrogate↩
        Model >, 15
nb_basis_functions
    BasisForSurrogateModelBaseClass, 6

QuadraticMinimization, 16
    minimize, 16
    QuadraticMinimization, 16
QuadraticMonomial, 17
    evaluate_monomial, 18
    QuadraticMonomial, 17

RegularizedMinimumFrobeniusNormModel, 18
replace_node
    ImprovePoisedness, 14
rescale
    VectorOperations, 24

scale
    VectorOperations, 24
set_zero
    VectorOperations, 25
SubproblemData< TSurrogateModel, TSubproblem↩
        Optimization >, 19
SubproblemDefinitions< TSurrogateModel, T↩
        SubproblemOptimization >, 19
SubproblemOptimization< TSurrogateModel >, 19
SurrogateModelBaseClass, 20

TriangularMatrixOperations, 20
    backward_substitution, 21
    compute_large_norm_solution, 21
    forward_substitution, 22
    TriangularMatrixOperations, 21

update
    GaussianProcess, 10

VectorOperations, 22

add, 23
diff_norm, 23
dot_product, 23
minus, 24
rescale, 24
scale, 24
set_zero, 25