Project

Generated by Doxygen 1.8.17

Chapter 1

Hierarchical Index

1.1 Class Hierarchy

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

Chapter 2

Class Index

2.1 Class List

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

Chapter 3

File Index

3.1 File List

Here is a list of all files with brief descriptions:

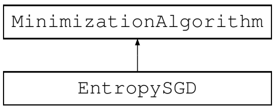
Chapter 4

Class Documentation

4.1 EntropySGD Class Reference

```
#include <EntropySGD.h>
```

Inheritance diagram for EntropySGD:



Public Member Functions

- EntropySGD (unsigned int, OptimizationFunction ∗, Eigen::VectorXd, unsigned int, unsigned int, unsigned int, double, double, double, double, double, double, double, double)
- EntropySGD (unsigned int, unsigned int, unsigned int, unsigned int, double, double, double, double, double, double, double, double)
- EntropySGD (OptimizationFunction ∗, Eigen::VectorXd, unsigned int, unsigned int, unsigned int, double, double, double, double, double, double, double, double)
- EntropySGD (OptimizationFunction ∗, unsigned int, unsigned int, unsigned int, double, double, double, double, double, double, double, double)
- EntropySGD (unsigned int, OptimizationFunction ∗, Eigen::VectorXd)
- EntropySGD (unsigned int)
- EntropySGD (OptimizationFunction ∗, Eigen::VectorXd)
- EntropySGD (OptimizationFunction ∗)
- void set_parameters (unsigned int, unsigned int, unsigned int, double, double, double, double, double, double, double, double)
- void solve () override
- double get_epochs () const override

    Returns the number of epochs, so the number of visits to the whole dataset.

Additional Inherited Members

4.1.1 Detailed Description

This class is derived from the abstract class MinimizationAlgorithm. This implements the Entropy SGD algorithm.

4.1.2 Constructor & Destructor Documentation

4.1.2.1 EntropySGD() [1/8]

```
EntropySGD::EntropySGD (
            unsigned int dim,
            OptimizationFunction * F_min,
            Eigen::VectorXd start,
            unsigned int max_ep,
            unsigned int mb,
            unsigned int L_it,
            double tol_f,
            double b,
            double g0,
            double g1,
            double y,
            double x,
            double a,
            double r )
```

4.1.2.2 EntropySGD() [2/8]

```
EntropySGD::EntropySGD (
            unsigned int dim,
            unsigned int max_ep,
            unsigned int mb,
            unsigned int L_it,
            double tol_f,
            double b,
            double g0,
            double g1,
            double y,
            double x,
            double a,
            double r )
```

4.1.2.3 EntropySGD() [3/8]

```
EntropySGD::EntropySGD (
            OptimizationFunction * F_min,
            Eigen::VectorXd start,
            unsigned int max_ep,
            unsigned int mb,
            unsigned int L_it,
            double tol_f,
            double b,
            double g0,
            double g1,
            double y,
            double x,
            double a,
            double r )
```

4.1.2.4 EntropySGD() [4/8]

```
EntropySGD::EntropySGD (
            OptimizationFunction * F_min,
            unsigned int max_ep,
            unsigned int mb,
            unsigned int L_it,
            double tol_f,
            double b,
            double g0,
            double g1,
            double y,
            double x,
            double a,
            double r )
```

4.1.2.5 EntropySGD() [5/8]

```
EntropySGD::EntropySGD (
            unsigned int dim,
            OptimizationFunction * F_min,
            Eigen::VectorXd start )
```

4.1.2.6 EntropySGD() [6/8]

```
EntropySGD::EntropySGD (
            unsigned int dim )
```

4.1.2.7 EntropySGD() [7/8]

```
EntropySGD::EntropySGD (
            OptimizationFunction * F_min,
            Eigen::VectorXd start )
```

4.1.2.8 EntropySGD() [8/8]

```
EntropySGD::EntropySGD (
            OptimizationFunction * F_min )
```

4.1.3 Member Function Documentation

4.1.3.1 get_epochs()

```
double EntropySGD::get_epochs ( ) const  [override], [virtual]
```

Returns the number of epochs, so the number of visits to the whole dataset.

Implements MinimizationAlgorithm.

4.1.3.2 set_parameters()

```
void EntropySGD::set_parameters (
            unsigned int max_ep,
            unsigned int mb,
            unsigned int L_it,
            double tol_f,
            double b,
            double g0,
            double g1,
            double y,
            double x,
            double a,
            double r )
```

Set parameters for the algorithm. This re-writes the values that were previously stored.

4.1.3.3 solve()

```
void EntropySGD::solve ( )  [override], [virtual]
```

Method that runs the algorithm. The final value of the parameters that minimizes the function is stored in min_point, which will be accessible with the getter method.

Implements MinimizationAlgorithm.

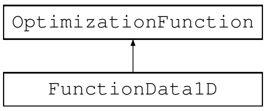
The documentation for this class was generated from the following files:

- C:/Users/SteDale/Documents/Uni/Magistrale/PACS/project/Code_C_final/CodeDoxygen/Headers/EntropySGD.h
- C:/Users/SteDale/Documents/Uni/Magistrale/PACS/project/Code_C_final/CodeDoxygen/Sources/EntropySGD.cpp

4.2 FunctionData1D Class Reference

```
#include <FunctionData1D.h>
```

Inheritance diagram for FunctionData1D:



Public Member Functions

- FunctionData1D ()=default
- FunctionData1D (unsigned int state_dim)
- FunctionData1D (unsigned int, std::function< double(const Eigen::VectorXd &, double)>, std::function< Eigen::VectorXd(const Eigen::VectorXd &, double)>)
- void make_dataset (unsigned int, double, double)
- double evaluate (const Eigen::VectorXd &) override
- Eigen::VectorXd stochastic_gradient (const Eigen::VectorXd &, unsigned int) override
- Eigen::VectorXd gradient (const Eigen::VectorXd &)
- void set_f (std::function< double(const Eigen::VectorXd &, double)>)

  Sets the base function.
- void set_df (std::function< Eigen::VectorXd(const Eigen::VectorXd &, double)>)

  Sets the gradient of the base function.
- void set_dataset (FunctionData1D ∗)
- void set_dataset (std::shared_ptr< std::vector< double >>)

  Sets the dataset using an already existing dataset.
- std::shared_ptr< std::vector< double > > get_dataset ()
- unsigned int get_data_dim () const override

4.2.1 Detailed Description

This class is derived from the abstract class OptimizationFunction. This function is based on a dataset made of scalar values (double), and provides also a method for the evaluation of the gradient.

4.2.2 Constructor & Destructor Documentation

4.2.2.1 FunctionData1D() [1/3]

```
FunctionData1D::FunctionData1D ( )  [default]
```

4.2.2.2 FunctionData1D() [2/3]

```
FunctionData1D::FunctionData1D (
             unsigned int state_dim )  [inline]
```

4.2.2.3 FunctionData1D() [3/3]

```
FunctionData1D::FunctionData1D (
             unsigned int state_dim,
             std::function< double(const Eigen::VectorXd &, double)> f,
             std::function< Eigen::VectorXd(const Eigen::VectorXd &, double)> df )
```

4.2.3 Member Function Documentation

4.2.3.1 evaluate()

```
double FunctionData1D::evaluate (
             const Eigen::VectorXd & x )  [override], [virtual]
```

This method evaluates the function, by summing the value of the base function on the dataset, according to the given value of the parameters (the argument x).

Implements OptimizationFunction.

4.2.3.2 get_data_dim()

```
unsigned int FunctionData1D::get_data_dim ( ) const  [inline], [override], [virtual]
```

Implements OptimizationFunction.

4.2.3.3 get_dataset()

```
std::shared_ptr<std::vector<double> > FunctionData1D::get_dataset ( )  [inline]
```

4.2.3.4 gradient()

```
Eigen::VectorXd FunctionData1D::gradient (
            const Eigen::VectorXd & x )
```

This method is not overridden from the base class. It computes the whole gradient by summing the gradient computed for ALL the instances in the dataset.

4.2.3.5 make_dataset()

```
void FunctionData1D::make_dataset (
            unsigned int data_dimention,
            double data_min,
            double data_max )
```

This method builds the dataset according to the parameters: the number of elements, the min and the max value. The dataset is build in a randomic way using a uniform distribution.

4.2.3.6 set_dataset() [1/2]

```
void FunctionData1D::set_dataset (
            FunctionData1D * other_function )
```

Sets the dataset, by copying the pointer to the dataset of another function. (The values are not copied)

4.2.3.7 set_dataset() [2/2]

```
void FunctionData1D::set_dataset (
            std::shared_ptr< std::vector< double >> new_dataset )
```

Sets the dataset using an already existing dataset.

4.2.3.8 set_df()

```
void FunctionData1D::set_df (
            std::function< Eigen::VectorXd(const Eigen::VectorXd &, double)> df )
```

Sets the gradient of the base function.

4.2.3.9 set_f()

```
void FunctionData1D::set_f (
            std::function< double(const Eigen::VectorXd &, double)> f )
```

Sets the base function.

4.2.3.10 stochastic_gradient()

```
Eigen::VectorXd FunctionData1D::stochastic_gradient (
             const Eigen::VectorXd & x,
             unsigned int mini_batch_size )  [override], [virtual]
```

Returns the value of the stochastic gradient at the point x given as a parameter. The stochastic gradient is computed by summing the value of te gradient only for some of the instances in the dataset.

Implements OptimizationFunction.

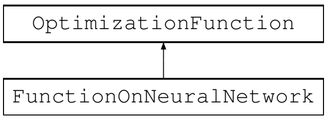
The documentation for this class was generated from the following files:

- C:/Users/SteDale/Documents/Uni/Magistrale/PACS/project/Code_C_final/CodeDoxygen/Headers/FunctionData1D.h
- C:/Users/SteDale/Documents/Uni/Magistrale/PACS/project/Code_C_final/CodeDoxygen/Sources/FunctionData1D.cpp

4.3 FunctionOnNeuralNetwork Class Reference

```
#include <FunctionOnNeuralNetwork.h>
```

Inheritance diagram for FunctionOnNeuralNetwork:



Public Member Functions

- FunctionOnNeuralNetwork ()=default
- FunctionOnNeuralNetwork (OpenNN::LossIndex ∗)
- void set_network_pointer (OpenNN::NeuralNetwork ∗)

  Sets the pointer to an exisitng OpenNN::NeuralNetwork.
- void set_dataset_pointer (OpenNN::DataSet ∗)

  Sets the pointer to an existing OpenNN::DataSet.
- double evaluate (const Eigen::VectorXd &) override

  Evaluates the Loss index function for the passed values for the parameters of the neural network.
- Eigen::VectorXd stochastic_gradient (const Eigen::VectorXd &, unsigned int) override

  Returns the stochastic gradient using the tools of the OpenNN library.
- unsigned int get_data_dim () const override

  Returns the dimention of the training dataset.
- OpenNN::NeuralNetwork ∗ get_neural_network_pointer () const

  Returns a pointer to the OpenNN::NeuralNetwork used.

4.3.1 Detailed Description

This class is derived from the abstract class OptimizationFunction. In particular this class uses the classes defined in the open library opennn for the implementation of neural networks.

4.3.2 Constructor & Destructor Documentation

4.3.2.1 FunctionOnNeuralNetwork() [1/2]

```
FunctionOnNeuralNetwork::FunctionOnNeuralNetwork ( )  [default]
```

4.3.2.2 FunctionOnNeuralNetwork() [2/2]

```
FunctionOnNeuralNetwork::FunctionOnNeuralNetwork (
                OpenNN::LossIndex * loss_index_ptr )
```

4.3.3 Member Function Documentation

4.3.3.1 evaluate()

```
double FunctionOnNeuralNetwork::evaluate (
                const Eigen::VectorXd & parameters_eigen )  [override], [virtual]
```

Evaluates the Loss index function for the passed values for the parameters of the neural network.

Implements OptimizationFunction.

4.3.3.2 get_data_dim()

```
unsigned int FunctionOnNeuralNetwork::get_data_dim ( ) const  [inline], [override], [virtual]
```

Returns the dimention of the training dataset.

Implements OptimizationFunction.

4.3.3.3 get_neural_network_pointer()

```
OpenNN::NeuralNetwork* FunctionOnNeuralNetwork::get_neural_network_pointer ( ) const  [inline]
```

Returns a pointer to the OpenNN::NeuralNetwork used.

4.3.3.4 set_dataset_pointer()

```
void FunctionOnNeuralNetwork::set_dataset_pointer (
             OpenNN::DataSet * new_data )
```

Sets the pointer to an existing OpenNN::DataSet.

4.3.3.5 set_network_pointer()

```
void FunctionOnNeuralNetwork::set_network_pointer (
             OpenNN::NeuralNetwork * new_network )
```

Sets the pointer to an exisitng OpenNN::NeuralNetwork.

4.3.3.6 stochastic_gradient()

```
Eigen::VectorXd FunctionOnNeuralNetwork::stochastic_gradient (
             const Eigen::VectorXd & parameters,
             unsigned int batch_dimention )  [override], [virtual]
```

Returns the stochastic gradient using the tools of the OpenNN library.

Implements OptimizationFunction.

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

- C:/Users/SteDale/Documents/Uni/Magistrale/PACS/project/Code_C_final/CodeDoxygen/Headers/FunctionOnNeuralNetwork.h
- C:/Users/SteDale/Documents/Uni/Magistrale/PACS/project/Code_C_final/CodeDoxygen/Sources/FunctionOnNeuralNetwork.c

4.4 GradientDescent Class Reference

```
#include <GradientDescent.h>
```

Public Member Functions

- GradientDescent (unsigned int, FunctionData1D ∗, Eigen::VectorXd)
- GradientDescent (unsigned int)
- GradientDescent (FunctionData1D ∗, Eigen::VectorXd)
- GradientDescent (FunctionData1D ∗)
- GradientDescent (unsigned int, FunctionData1D ∗, Eigen::VectorXd, double, double, double, double, double)
- GradientDescent (unsigned int, double, double, double, double, double)
- GradientDescent (FunctionData1D ∗, Eigen::VectorXd, double, double, double, double, double)
- GradientDescent (FunctionData1D ∗, double, double, double, double, double)
- void set_state_dim (unsigned int)
- void set_state_dim_no_matter_what (unsigned int)
- void set_function (FunctionData1D ∗)
- void set_function_no_matter_what (FunctionData1D ∗)
- void set_starting_point (Eigen::VectorXd)
- void set_parameters (double, double, double, double, double)
- Eigen::VectorXd get_min () const

  Returns the final value of the parameters, found and the end of the algorithm.
- unsigned int get_iterations () const

  Returns the iterations needed for the minimization algorithm.
- double get_final_value () const
- double get_epochs () const

  Returns the number of epochs, so the number of visits to the whole dataset.
- double get_computation_time () const

  Returns the computation time for the minimization, in milliseconds.
- void solve ()

4.4.1 Detailed Description

This class implements the Gradient Descent algorithm for the minimization of functions. This class was not derived from the class MinimizationAlgorithm because the function thet is minimized needs to provide also a method for the evaluation of the gradient.

4.4.2 Constructor & Destructor Documentation

4.4.2.1 GradientDescent() [1/8]

```
GradientDescent::GradientDescent (
            unsigned int dim,
            FunctionData1D * F_min,
            Eigen::VectorXd start )
```

4.4.2.2 GradientDescent() [2/8]

```
GradientDescent::GradientDescent (
            unsigned int dim )
```

4.4.2.3 GradientDescent() [3/8]

```
GradientDescent::GradientDescent (
            FunctionData1D * F_min,
            Eigen::VectorXd start )
```

4.4.2.4 GradientDescent() [4/8]

```
GradientDescent::GradientDescent (
            FunctionData1D * F_min )
```

4.4.2.5 GradientDescent() [5/8]

```
GradientDescent::GradientDescent (
            unsigned int dim,
            FunctionData1D * F_min,
            Eigen::VectorXd start,
            double tol_t,
            double tol_f,
            double t,
            double b,
            double a )
```

4.4.2.6 GradientDescent() [6/8]

```
GradientDescent::GradientDescent (
            unsigned int dim,
            double tol_t,
            double tol_f,
            double t,
            double b,
            double a )
```

4.4.2.7 GradientDescent() [7/8]

```
GradientDescent::GradientDescent (
            FunctionData1D * F_min,
            Eigen::VectorXd start,
            double tol_t,
            double tol_f,
            double t,
            double b,
            double a )
```

4.4.2.8 GradientDescent() [8/8]

```
GradientDescent::GradientDescent (
            FunctionData1D * F_min,
            double tol_t,
            double tol_f,
            double t,
            double b,
            double a )
```

4.4.3 Member Function Documentation

4.4.3.1 get_computation_time()

```
double GradientDescent::get_computation_time ( ) const
```

Returns the computation time for the minimization, in milliseconds.

4.4.3.2 get_epochs()

```
double GradientDescent::get_epochs ( ) const
```

Returns the number of epochs, so the number of visits to the whole dataset.

4.4.3.3 get_final_value()

```
double GradientDescent::get_final_value ( ) const
```

Returns the final value of the function found at the end of the algorithm. This value of the function corresponds to the value computed at min_point.

4.4.3.4 get_iterations()

```
unsigned int GradientDescent::get_iterations ( ) const
```

Returns the iterations needed for the minimization algorithm.

4.4.3.5 get_min()

```
Eigen::VectorXd GradientDescent::get_min ( ) const
```

Returns the final value of the parameters, found and the end of the algorithm.

4.4.3.6 set_function()

```
void GradientDescent::set_function (
            FunctionData1D * F_min )
```

This method changes the pointer to the function to be optimized, but only if the dimention of the new function is equal to the dimention of the state.

4.4.3.7 set_function_no_matter_what()

```
void GradientDescent::set_function_no_matter_what (
            FunctionData1D * F_min )
```

This method changes the pointer to the FunctionData1D, even if the dimention of the state and the dimention of the new function are not coherent. In this case the new dimention is set to the dimention of the state of the new function.

4.4.3.8 set_parameters()

```
void GradientDescent::set_parameters (
            double tol_t,
            double tol_f,
            double t,
            double b,
            double a )
```

Set parameters for the algorithm. This re-writes the values that were previously stored.

4.4.3.9 set_starting_point()

```
void GradientDescent::set_starting_point (
            Eigen::VectorXd start )
```

Method to set the starting point of the algorithm. The dimention of the starting point must match with the dimention of the state.

4.4.3.10 set_state_dim()

```
void GradientDescent::set_state_dim (
            unsigned int dim )
```

This method sets the dimention of the state, but only if the function to be optimized is not already set.

4.4.3.11 set_state_dim_no_matter_what()

```
void GradientDescent::set_state_dim_no_matter_what (
            unsigned int dim )
```

This algorithm changes the dimention of the state, even if the new dimention is not coherent with the dimention of the FunctionData1D. In that case the pointer to the FunctionData1D is set as NULL.

4.4.3.12 solve()

```
void GradientDescent::solve ( )
```

Method that runs the algorithm. The final value of the parameters that minimizes the function is stored in min_point, which will be accessible with the getter method.

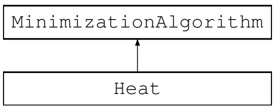
The documentation for this class was generated from the following files:

- C:/Users/SteDale/Documents/Uni/Magistrale/PACS/project/Code_C_final/CodeDoxygen/Headers/GradientDescent.h
- C:/Users/SteDale/Documents/Uni/Magistrale/PACS/project/Code_C_final/CodeDoxygen/Sources/GradientDescent.cpp

4.5 Heat Class Reference

```
#include <Heat.h>
```

Inheritance diagram for Heat:



Public Member Functions

- Heat (unsigned int, OptimizationFunction ∗, Eigen::VectorXd, unsigned int, unsigned int, unsigned int, double, double, double, double)
- Heat (unsigned int, unsigned int, unsigned int, unsigned int, double, double, double, double)
- Heat (OptimizationFunction ∗, Eigen::VectorXd, unsigned int, unsigned int, unsigned int, double, double, double, double)
- Heat (OptimizationFunction ∗, unsigned int, unsigned int, unsigned int, double, double, double, double)
- Heat (unsigned int, OptimizationFunction ∗, Eigen::VectorXd)
- Heat (unsigned int)
- Heat (OptimizationFunction ∗, Eigen::VectorXd)
- Heat (OptimizationFunction ∗)
- void set_parameters (unsigned int, unsigned int, unsigned int, double, double, double, double)
- void solve () override
- double get_epochs () const override

    Returns the number of epochs, so the number of visits to the whole dataset.

Additional Inherited Members

4.5.1 Detailed Description

This class is derived from the abstract class MinimizationAlgorithm. This implements the Heat SGD algorithm.

4.5.2 Constructor & Destructor Documentation

4.5.2.1 Heat() [1/8]

```
Heat::Heat (
            unsigned int dim,
            OptimizationFunction * F_min,
            Eigen::VectorXd start,
            unsigned int max_ep,
            unsigned int mb,
            unsigned int L_it,
            double tol_f,
            double g0,
            double g1,
            double x )
```

4.5.2.2 Heat() [2/8]

```
Heat::Heat (
            unsigned int dim,
            unsigned int max_ep,
            unsigned int mb,
            unsigned int L_it,
            double tol_f,
            double g0,
            double g1,
            double x )
```

4.5.2.3 Heat() [3/8]

```
Heat::Heat (
            OptimizationFunction * F_min,
            Eigen::VectorXd start,
            unsigned int max_ep,
            unsigned int mb,
            unsigned int L_it,
            double tol_f,
            double g0,
            double g1,
            double x )
```

4.5.2.4 Heat() [4/8]

```
Heat::Heat (
            OptimizationFunction * F_min,
            unsigned int max_ep,
            unsigned int mb,
            unsigned int L_it,
            double tol_f,
            double g0,
            double g1,
            double x )
```

4.5.2.5 Heat() [5/8]

```
Heat::Heat (
            unsigned int dim,
            OptimizationFunction * F_min,
            Eigen::VectorXd start )
```

4.5.2.6 Heat() [6/8]

```
Heat::Heat (
            unsigned int dim )
```

4.5.2.7 Heat() [7/8]

```
Heat::Heat (
            OptimizationFunction * F_min,
            Eigen::VectorXd start )
```

4.5.2.8 Heat() [8/8]

```
Heat::Heat (
            OptimizationFunction * F_min )
```

4.5.3 Member Function Documentation

4.5.3.1 get_epochs()

```
double Heat::get_epochs ( ) const  [override], [virtual]
```

Returns the number of epochs, so the number of visits to the whole dataset.

Implements MinimizationAlgorithm.

4.5.3.2 set_parameters()

```
void Heat::set_parameters (
            unsigned int max_ep,
            unsigned int mb,
            unsigned int L_it,
            double tol_f,
            double g0,
            double g1,
            double x )
```

Set parameters for the algorithm. This re-writes the values that were previously stored.

4.5.3.3 solve()

```
void Heat::solve ( )  [override], [virtual]
```

Method that runs the algorithm. The final value of the parameters that minimizes the function is stored in min_point, which will be accessible with the getter method.

Implements MinimizationAlgorithm.

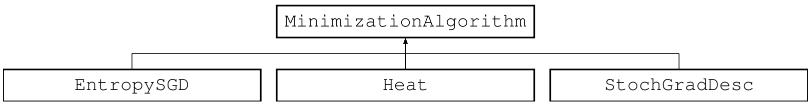
The documentation for this class was generated from the following files:

- C:/Users/SteDale/Documents/Uni/Magistrale/PACS/project/Code_C_final/CodeDoxygen/Headers/Heat.h
- C:/Users/SteDale/Documents/Uni/Magistrale/PACS/project/Code_C_final/CodeDoxygen/Sources/Heat.cpp

4.6 MinimizationAlgorithm Class Reference

```
#include <MinimizationAlgorithm.h>
```

Inheritance diagram for MinimizationAlgorithm:



Public Member Functions

- MinimizationAlgorithm (unsigned int, OptimizationFunction ∗, Eigen::VectorXd)
- MinimizationAlgorithm (unsigned int)
- MinimizationAlgorithm (OptimizationFunction ∗, Eigen::VectorXd)
- MinimizationAlgorithm (OptimizationFunction ∗)
- void set_state_dim (unsigned int)
- void set_state_dim_no_matter_what (unsigned int)
- void set_function (OptimizationFunction ∗)
- void set_function_no_matter_what (OptimizationFunction ∗)
- void set_starting_point (Eigen::VectorXd)
- Eigen::VectorXd get_min () const

  Returns the final value of the parameters, found and the end of the algorithm.
- unsigned int get_iterations () const

  Returns the iterations needed for the minimization algorithm.
- double get_final_value () const
- double get_computation_time () const

  Returns the computation time for the minimization, in milliseconds.
- virtual void solve ()=0
- virtual double get_epochs () const =0

Protected Attributes

- unsigned int state_dim =1
- OptimizationFunction ∗ F_minimization =NULL
- Eigen::VectorXd min_point
- Eigen::VectorXd starting_point
- unsigned int iterations =0
- std::vector< double > Function_values_sequence
- double comp_time =0.0

4.6.1 Detailed Description

This is an abstract class, used as a basis for the development of algorithms for the minimization of functions that depend on a dataset. In particular this class contains a pointer to the abstract class OptimizationFunction. Any class derived from that would possibly be minimized by an algorith, derived from this class.

4.6.2 Constructor & Destructor Documentation

4.6.2.1 MinimizationAlgorithm() [1/4]

```
MinimizationAlgorithm::MinimizationAlgorithm (
            unsigned int dim,
            OptimizationFunction * F_min,
            Eigen::VectorXd start )
```

4.6.2.2 MinimizationAlgorithm() [2/4]

```
MinimizationAlgorithm::MinimizationAlgorithm (
            unsigned int dim )
```

4.6.2.3 MinimizationAlgorithm() [3/4]

```
MinimizationAlgorithm::MinimizationAlgorithm (
            OptimizationFunction * F_min,
            Eigen::VectorXd start )
```

4.6.2.4 MinimizationAlgorithm() [4/4]

```
MinimizationAlgorithm::MinimizationAlgorithm (
            OptimizationFunction * F_min )
```

4.6.3 Member Function Documentation

4.6.3.1 get_computation_time()

```
double MinimizationAlgorithm::get_computation_time ( ) const
```

Returns the computation time for the minimization, in milliseconds.

4.6.3.2 get_epochs()

```
virtual double MinimizationAlgorithm::get_epochs ( ) const  [pure virtual]
```

Implemented in EntropySGD, Heat, and StochGradDesc.

4.6.3.3 get_final_value()

```
double MinimizationAlgorithm::get_final_value ( ) const
```

Returns the final value of the function found at the end of the algorithm. This value of the function corresponds to the value computed at min_point.

4.6.3.4 get_iterations()

```
unsigned int MinimizationAlgorithm::get_iterations ( ) const
```

Returns the iterations needed for the minimization algorithm.

4.6.3.5 get_min()

```
Eigen::VectorXd MinimizationAlgorithm::get_min ( ) const
```

Returns the final value of the parameters, found and the end of the algorithm.

4.6.3.6 set_function()

```
void MinimizationAlgorithm::set_function (
             OptimizationFunction * F_min )
```

This method changes the pointer to the function to be optimized, but only if the dimention of the new function is equal to the dimention of the state.

4.6.3.7 set_function_no_matter_what()

```
void MinimizationAlgorithm::set_function_no_matter_what (
             OptimizationFunction * F_min )
```

This method changes the pointer to the OptimizationFunction, even if the dimention of the state and the dimention of the new function are not coherent. In this case the new dimention is set to the dimention of the state of the new function.

4.6.3.8 set_starting_point()

```
void MinimizationAlgorithm::set_starting_point (
             Eigen::VectorXd start )
```

Method to set the starting point of the algorithm. The dimention of the starting point must match with the dimention of the state.

4.6.3.9 set_state_dim()

```
void MinimizationAlgorithm::set_state_dim (
             unsigned int dim )
```

This method sets the dimention of the state, but only if the function to be optimized is not already set.

4.6.3.10 set_state_dim_no_matter_what()

```
void MinimizationAlgorithm::set_state_dim_no_matter_what (
             unsigned int dim )
```

This algorithm changes the dimention of the state, even if the new dimention is not coherent with the dimention of the OptimizationFunction. In that case the pointer to the OptimizationFunction is set as NULL.

4.6.3.11 solve()

```
virtual void MinimizationAlgorithm::solve ( )  [pure virtual]
```

Implemented in EntropySGD, Heat, and StochGradDesc.

4.6.4 Member Data Documentation

4.6.4.1 comp_time

```
double MinimizationAlgorithm::comp_time =0.0  [protected]
```

4.6.4.2 F_minimization

```
OptimizationFunction* MinimizationAlgorithm::F_minimization =NULL  [protected]
```

4.6.4.3 Function_values_sequence

```
std::vector<double> MinimizationAlgorithm::Function_values_sequence  [protected]
```

4.6.4.4 iterations

```
unsigned int MinimizationAlgorithm::iterations =0  [protected]
```

4.6.4.5 min_point

```
Eigen::VectorXd MinimizationAlgorithm::min_point  [protected]
```

4.6.4.6 starting_point

```
Eigen::VectorXd MinimizationAlgorithm::starting_point  [protected]
```

4.6.4.7 state_dim

```
unsigned int MinimizationAlgorithm::state_dim =1  [protected]
```

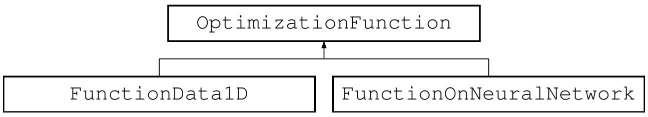
The documentation for this class was generated from the following files:

- C:/Users/SteDale/Documents/Uni/Magistrale/PACS/project/Code_C_final/CodeDoxygen/Headers/MinimizationAlgorithm.h
- C:/Users/SteDale/Documents/Uni/Magistrale/PACS/project/Code_C_final/CodeDoxygen/Sources/MinimizationAlgorithm.cpp

4.7 OptimizationFunction Class Reference

```
#include <OptimizationFunction.h>
```

Inheritance diagram for OptimizationFunction:



Public Member Functions

- OptimizationFunction ()=default
- OptimizationFunction (unsigned int dim)
- unsigned int get_state_dim () const
- void set_state_dim (unsigned int dim)

    Re-writes the previous value of the state_dim;.
- virtual double evaluate (const Eigen::VectorXd &)=0
- virtual Eigen::VectorXd stochastic_gradient (const Eigen::VectorXd &, unsigned int)=0
- virtual unsigned int get_data_dim () const =0

4.7.1 Detailed Description

This is an abstract class that is used as a model for functions to be minimized by minimization algorithms. In particular the classes derived from this class must provide a stochastic_gradient method for the evaluation of the stochastic gradient, given a state (a set of parameters) and a dimention for the mini batch.

4.7.2 Constructor & Destructor Documentation

4.7.2.1 OptimizationFunction() [1/2]

```
OptimizationFunction::OptimizationFunction ( )  [default]
```

4.7.2.2 OptimizationFunction() [2/2]

```
OptimizationFunction::OptimizationFunction (
            unsigned int dim )  [inline], [explicit]
```

4.7.3 Member Function Documentation

4.7.3.1 evaluate()

```
virtual double OptimizationFunction::evaluate (
            const Eigen::VectorXd &  )  [pure virtual]
```

Implemented in FunctionData1D, and FunctionOnNeuralNetwork.

4.7.3.2 get_data_dim()

```
virtual unsigned int OptimizationFunction::get_data_dim ( ) const  [pure virtual]
```

Implemented in FunctionData1D, and FunctionOnNeuralNetwork.

4.7.3.3 get_state_dim()

```
unsigned int OptimizationFunction::get_state_dim ( ) const  [inline]
```

Returns the dimention of the state, or the number of the parameters for the optimization.

4.7.3.4 set_state_dim()

```
void OptimizationFunction::set_state_dim (
            unsigned int dim )  [inline]
```

Re-writes the previous value of the state_dim;.

4.7.3.5 stochastic_gradient()

```
virtual Eigen::VectorXd OptimizationFunction::stochastic_gradient (
             const Eigen::VectorXd & ,
             unsigned int  )  [pure virtual]
```

Implemented in FunctionData1D, and FunctionOnNeuralNetwork.

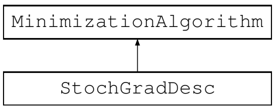
The documentation for this class was generated from the following file:

- C:/Users/SteDale/Documents/Uni/Magistrale/PACS/project/Code_C_final/CodeDoxygen/Headers/OptimizationFunction.h

4.8 StochGradDesc Class Reference

`#include <StochGradDesc.h>`

Inheritance diagram for StochGradDesc:



Public Member Functions

- StochGradDesc (unsigned int, OptimizationFunction ∗, Eigen::VectorXd, unsigned int, unsigned int, double, double, double, double)
- StochGradDesc (unsigned int, unsigned int, unsigned int, double, double, double, double)
- StochGradDesc (OptimizationFunction ∗, Eigen::VectorXd, unsigned int, unsigned int, double, double, double, double)
- StochGradDesc (OptimizationFunction ∗, unsigned int, unsigned int, double, double, double, double)
- StochGradDesc (unsigned int, OptimizationFunction ∗, Eigen::VectorXd)
- StochGradDesc (unsigned int)
- StochGradDesc (OptimizationFunction ∗, Eigen::VectorXd)
- StochGradDesc (OptimizationFunction ∗)
- void set_parameters (unsigned int, unsigned int, double, double, double, double)
- void solve () override
- double get_epochs () const override

  Returns the number of epochs, so the number of visits to the whole dataset.

Additional Inherited Members

4.8.1 Detailed Description

This class is derived from the abstract class MinimizationAlgorithm. This implements the Stochastic Gradient Descent algorithm.

4.8.2 Constructor & Destructor Documentation

4.8.2.1 StochGradDesc() [1/8]

```
StochGradDesc::StochGradDesc (
            unsigned int dim,
            OptimizationFunction * F_min,
            Eigen::VectorXd start,
            unsigned int max_it,
            unsigned int mb,
            double tol_t,
            double tol_f,
            double b,
            double t )
```

4.8.2.2 StochGradDesc() [2/8]

```
StochGradDesc::StochGradDesc (
            unsigned int dim,
            unsigned int max_it,
            unsigned int mb,
            double tol_t,
            double tol_f,
            double b,
            double t )
```

4.8.2.3 StochGradDesc() [3/8]

```
StochGradDesc::StochGradDesc (
            OptimizationFunction * F_min,
            Eigen::VectorXd start,
            unsigned int max_it,
            unsigned int mb,
            double tol_t,
            double tol_f,
            double b,
            double t )
```

4.8.2.4 StochGradDesc() [4/8]

```
StochGradDesc::StochGradDesc (
            OptimizationFunction * F_min,
            unsigned int max_it,
            unsigned int mb,
            double tol_t,
            double tol_f,
            double b,
            double t )
```

4.8.2.5 StochGradDesc() [5/8]

```
StochGradDesc::StochGradDesc (
            unsigned int dim,
            OptimizationFunction * F_min,
            Eigen::VectorXd start )
```

4.8.2.6 StochGradDesc() [6/8]

```
StochGradDesc::StochGradDesc (
            unsigned int dim )
```

4.8.2.7 StochGradDesc() [7/8]

```
StochGradDesc::StochGradDesc (
            OptimizationFunction * F_min,
            Eigen::VectorXd start )
```

4.8.2.8 StochGradDesc() [8/8]

```
StochGradDesc::StochGradDesc (
            OptimizationFunction * F_min )
```

4.8.3 Member Function Documentation

4.8.3.1 get_epochs()

```
double StochGradDesc::get_epochs ( ) const  [override], [virtual]
```

Returns the number of epochs, so the number of visits to the whole dataset.

Implements MinimizationAlgorithm.

4.8.3.2 set_parameters()

```
void StochGradDesc::set_parameters (
             unsigned int max_it,
             unsigned int mb,
             double tol_t,
             double tol_f,
             double b,
             double t )
```

Set parameters for the algorithm. This re-writes the values that were previously stored.

4.8.3.3 solve()

```
void StochGradDesc::solve ( )  [override], [virtual]
```

Method that runs the algorithm. The final value of the parameters that minimizes the function is stored in min_point, which will be accessible with the getter method.

Implements MinimizationAlgorithm.

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

- C:/Users/SteDale/Documents/Uni/Magistrale/PACS/project/Code_C_final/CodeDoxygen/Headers/StochGradDesc.h
- C:/Users/SteDale/Documents/Uni/Magistrale/PACS/project/Code_C_final/CodeDoxygen/Sources/StochGradDesc.cpp

Chapter 5

File Documentation

5.1 C:/Users/SteDale/Documents/Uni/Magistrale/PACS/project/Code_C↩ _final/CodeDoxygen/Headers/EntropySGD.h File Reference

```
#include "OptimizationFunction.h"
#include "MinimizationAlgorithm.h"
#include <Eigen/Dense>
```

Classes

- class EntropySGD

5.2 C:/Users/SteDale/Documents/Uni/Magistrale/PACS/project/Code_C↩ _final/CodeDoxygen/Headers/FunctionData1D.h File Reference

```
#include "OptimizationFunction.h"
#include <Eigen/Dense>
#include <stdlib.h>
#include <functional>
#include <memory>
#include <vector>
```

Classes

- class FunctionData1D

5.3 C:/Users/SteDale/Documents/Uni/Magistrale/PACS/project/Code_C↩_final/CodeDoxygen/Headers/FunctionOnNeuralNetwork.h File Reference

```
#include "../Headers/opennn_headers/opennn.h"
#include "../Headers/OptimizationFunction.h"
```

Classes

- class FunctionOnNeuralNetwork

5.4 C:/Users/SteDale/Documents/Uni/Magistrale/PACS/project/Code_C↩_final/CodeDoxygen/Headers/GradientDescent.h File Reference

```
#include "FunctionData1D.h"
#include <Eigen/Dense>
#include <vector>
```

Classes

- class GradientDescent

5.5 C:/Users/SteDale/Documents/Uni/Magistrale/PACS/project/Code_C↩_final/CodeDoxygen/Headers/Heat.h File Reference

```
#include "OptimizationFunction.h"
#include "MinimizationAlgorithm.h"
#include <Eigen/Dense>
```

Classes

- class Heat

5.6 C:/Users/SteDale/Documents/Uni/Magistrale/PACS/project/Code_C↩_final/CodeDoxygen/Headers/MinimizationAlgorithm.h File Reference

```
#include "OptimizationFunction.h"
#include <Eigen/Dense>
#include <vector>
```

Classes

- class MinimizationAlgorithm

5.7 C:/Users/SteDale/Documents/Uni/Magistrale/PACS/project/Code_C↩_final/CodeDoxygen/Headers/OptimizationFunction.h File Reference

```
#include <Eigen/Dense>
```

Classes

- class OptimizationFunction

5.8 C:/Users/SteDale/Documents/Uni/Magistrale/PACS/project/Code_C↩_final/CodeDoxygen/Headers/StochGradDesc.h File Reference

```
#include "OptimizationFunction.h"
#include "MinimizationAlgorithm.h"
#include <Eigen/Dense>
```

Classes

- class StochGradDesc

5.9 C:/Users/SteDale/Documents/Uni/Magistrale/PACS/project/Code_C↩_final/CodeDoxygen/Sources/EntropySGD.cpp File Reference

```
#include "../Headers/OptimizationFunction.h"
#include "../Headers/MinimizationAlgorithm.h"
#include "../Headers/EntropySGD.h"
#include <Eigen/Dense>
#include <iostream>
#include <cmath>
#include <random>
#include <chrono>
```

5.10 C:/Users/SteDale/Documents/Uni/Magistrale/PACS/project/Code_↩ C_final/CodeDoxygen/Sources/FunctionData1D.cpp File Reference

```
#include "../Headers/OptimizationFunction.h"
#include "../Headers/FunctionData1D.h"
#include <Eigen/Dense>
#include <stdlib.h>
#include <iostream>
#include <functional>
#include <memory>
#include <random>
```

5.11 C:/Users/SteDale/Documents/Uni/Magistrale/PACS/project/Code_↩ C_final/CodeDoxygen/Sources/FunctionOnNeuralNetwork.cpp File Reference

```
#include "../Headers/OptimizationFunction.h"
#include "../Headers/FunctionOnNeuralNetwork.h"
#include <opennn.h>
#include <Eigen/Dense>
#include <stdlib.h>
#include <random>
```

5.12 C:/Users/SteDale/Documents/Uni/Magistrale/PACS/project/Code_↩ C_final/CodeDoxygen/Sources/GradientDescent.cpp File Reference

```
#include "../Headers/FunctionData1D.h"
#include "../Headers/GradientDescent.h"
#include <Eigen/Dense>
#include <iostream>
#include <cmath>
#include <chrono>
```

5.13 C:/Users/SteDale/Documents/Uni/Magistrale/PACS/project/Code_↩ C_final/CodeDoxygen/Sources/Heat.cpp File Reference

```
#include "../Headers/OptimizationFunction.h"
#include "../Headers/MinimizationAlgorithm.h"
#include "../Headers/Heat.h"
```

```
#include <Eigen/Dense>
#include <iostream>
#include <cmath>
#include <random>
#include <chrono>
```

5.14 C:/Users/SteDale/Documents/Uni/Magistrale/PACS/project/Code_↩ C_final/CodeDoxygen/Sources/MinimizationAlgorithm.cpp File Reference

```
#include "../Headers/MinimizationAlgorithm.h"
#include "../Headers/OptimizationFunction.h"
#include <Eigen/Dense>
#include <iostream>
```

5.15 C:/Users/SteDale/Documents/Uni/Magistrale/PACS/project/Code_↩ C_final/CodeDoxygen/Sources/StochGradDesc.cpp File Reference

```
#include "../Headers/OptimizationFunction.h"
#include "../Headers/MinimizationAlgorithm.h"
#include "../Headers/StochGradDesc.h"
#include <Eigen/Dense>
#include <iostream>
#include <cmath>
#include <chrono>
```