FNN++

1

Generated by Doxygen 1.8.9.1

Sun May 10 2015 22:16:51

Contents

Chapter 1

Namespace Index

1.1 Namespace List

Here is a list of all documented namespaces with brief descriptions:

Chapter 2

Hierarchical Index

2.1 Class Hierarchy

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

Chapter 3

Class Index

3.1 Class List

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

Chapter 4

Namespace Documentation

4.1 fnn Namespace Reference

Different sigmoid activation functions.

Classes

- struct DataPoint
- class DataSet
- class Experiment
- class FNNTrainer

  The Trainer for the Network
- class Log

  A log.
- class Loggable

  A loggable.
- class LogManager

  Manager for logs.
- class Math

  The main mathematics helper class for FNNLIB
- class Network

  The main class of operation on the functional neural networks.
- class Sigmoid
- class WeightSurface

4.1.1 Detailed Description

Different sigmoid activation functions.

==========

William Guss, 4/6/2015.

Chapter 5

Class Documentation

5.1 fnn::DataPoint Struct Reference

Public Attributes

- std::function< double(double)> **input**
- std::function< double(double)> **desired**

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

- FNN++/FNNLib/FNNDataPoint.h

5.2 fnn::DataSet Class Reference

Public Member Functions

- DataSet ()
  Constructor that initializes the class
- virtual void Load ()=0
  Loads the DataSet
- void Shuffle ()
  Shuffles this DataSet
- std::vector< double(double)> calculateErrors (Network &nn, double step=-1)
  Calculates the errors.
- double calcError (Network &nn, double step=-1)
  Calculates the error.
- int size ()
  Gets the size.
- DataPoint & DataSet::operator[ ] (int index)
  Array indexer operator.

5.2.1 Constructor & Destructor Documentation

5.2.1.1 fnn::DataSet::DataSet (void)

Constructor that initializes the class

==========

==========

Phillip Kuznetsov, 5/6/2015.

Constructor that initializes the class.

Phillip Kuznetsov, 5/6/2015.

5.2.2 Member Function Documentation

5.2.2.1 double fnn::DataSet::calcError (Network & *nn,* double *step* = −1)

Calculates the error.

Phillip Kuznetsov, 5/6/2015.

Parameters

nn	The nn.
step	Amount to increment by.

Returns

The calculated error.

5.2.2.2 std::vector<double(double)> fnn::DataSet::calculateErrors (Network & *nn,* double *step* = −1)

Calculates the errors.

Phillip Kuznetsov, 5/6/2015.

Parameters

n	The Network to process.
step	Amount to increment by.

Returns

The calculated errors.

5.2.2.3 DataPoint& fnn::DataSet::DataSet::operator[ ] (int *index*)

Array indexer operator.

Phillip Kuznetsov, 5/6/2015.

Parameters

index	Zero-based index of the.

Returns

The indexed value from DataPoints.

5.2.2.4 virtual void fnn::DataSet::Load ( ) [pure virtual]

Loads the DataSet

Phillip Kuznetsov, 5/6/2015.

5.2.2.5 void fnn::DataSet::Shuffle ( )

Shuffles this DataSet

Phillip Kuznetsov, 5/6/2015.

5.2.2.6 int fnn::DataSet::size (void)

Gets the size.

Phillip Kuznetsov, 5/6/2015.

Returns

An int size of the DataSet

Phillip Kuznetsov, 5/6/2015.

Returns

An int size of the DataSet.

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

- FNN++/FNNLib/FNNDataSet.h
- FNN++/FNNLib/FNNDataSet.cpp

5.3 fnn::Experiment Class Reference

Public Member Functions

- Experiment (DataSet &trainingSet, DataSet &testingSet)

  Constructor.
- virtual void Run ()=0

  Runs this object.
- bool RunAsThread ()

  Executes as thread operation.
- void toggleThread ()

  Toggles whether the thread is running or not

5.3.1 Constructor & Destructor Documentation

5.3.1.1 fnn::Experiment::Experiment (DataSet & *trainingSet,* DataSet & *testingSet*)

Constructor.

Phillip Kuznetsov, 5/8/2015.

Parameters

trainingSet	Set the training belongs to.
testingSet	Set the testing belongs to.

5.3.2 Member Function Documentation

5.3.2.1 virtual void fnn::Experiment::Run ( ) [pure virtual]

Runs this object.

Phillip Kuznetsov, 5/8/2015.

5.3.2.2 bool fnn::Experiment::RunAsThread ( )

Executes as thread operation.

Phillip Kuznetsov, 5/8/2015.

Returns

true if it succeeds, false if it fails.

5.3.2.3 void fnn::Experiment::toggleThread ( )

Toggles whether the thread is running or not

Toggles whether the thread is running or not.

Phillip Kuznetsov, 5/8/2015.

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

- FNN++/FNNLib/FNNExperiment.h
- FNN++/FNNLib/FNNExperiment.cpp

5.4 fnn::FNNTrainer Class Reference

The Trainer for the Network

`#include <FNNTrainer.h>`

Public Member Functions

- FNNTrainer (Network &network, fnn::DataSet &trainingSet, fnn::DataSet &testingSet)

  FNNTrainer that takes a 2D vector for the trainingSet and testingSet.
- int Train (int epochs, double minError, std::vector< double > &learningParameters, bool nudging=false)

  Trains the network according to the parameters
- double Bound (double val, double min, double max)

  Bounds a double to a range of min and max

5.4.1 Detailed Description

The Trainer for the Network

==========

Phillip Kuznetsov, 5/6/2015.

5.4.2 Constructor & Destructor Documentation

5.4.2.1 fnn::FNNTrainer::FNNTrainer (Network & *network,* fnn::DataSet & *trainingSet,* fnn::DataSet & *testingSet*)

FNNTrainer that takes a 2D vector for the trainingSet and testingSet.

==========

Phillip Kuznetsov, 5/6/2015.

Parameters

network	The Network being operated on.
trainingSet	The training dataset.
testingSet	The traiing dataset.

5.4.3 Member Function Documentation

5.4.3.1 double fnn::FNNTrainer::Bound (double *val,* double *min,* double *max*)

Bounds a double to a range of min and max

Bounds a double to a range of min and max.

Phillip Kuznetsov, 5/6/2015.

Parameters

val	The value.
min	The minimum.
max	The maximum.

Returns

A double.

5.4.3.2 int fnn::FNNTrainer::Train (int *epochs,* double *minError,* std::vector< double > & *learningParameters,* bool *nudging* = `false`)

Trains the network according to the parameters

==========

Phillip Kuznetsov, 5/6/2015.

Parameters

epochs	The epochs.
minError	The minimum error.
learning↩ Parameters	Options for controlling the learning.
nudging	Whether the trainer uses nudging

An int.

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

- FNN++/FNNLib/FNNTrainer.h
- FNN++/FNNLib/FNNTrainer.cpp

5.5 fnn::Log Class Reference

A log.

`#include <FNNLog.h>`

Public Member Functions

- Log ()

  Default constructor.
- Log (string name, bool verbose)

  Constructs the log with a name and a vocality.
- void Push (string message)

  Pushes an object onto this log.
- list< string > * GetContent ()

  Gets the content of the log.
- string GetName ()

  Gets the name.
- bool IsVerbose ()

  Query if this object is verbose.

5.5.1 Detailed Description

A log.

==========

William, 4/29/2015.

5.5.2 Constructor & Destructor Documentation

5.5.2.1 fnn::Log::Log (void)

Default constructor.

==========

William, 4/29/2015.

5.5.2.2 fnn::Log::Log (string *name*, bool *verbose*)

Constructs the log with a name and a vocality.

==========

==========

William, 4/29/2015.

Parameters

name	The name.

true to vocal.

Constructs the log with a name and a vocality.

William, 4/29/2015.

Parameters

name	The name.

true to vocal.

Constructs the log.

William, 4/29/2015.

Parameters

name	The name.

true to verbose.

5.5.3 Member Function Documentation

5.5.3.1 list< string > * fnn::Log::GetContent (void)

Gets the content of the log.

==========

William, 4/29/2015.

null if it fails, else the content.

5.5.3.2 string fnn::Log::GetName (void)

Gets the name.

==========

William, 4/29/2015.

The name.

5.5.3.3 bool fnn::Log::IsVerbose (void)

Query if this object is verbose.

==========

William, 4/29/2015.

true if verbose, false if not.

5.5.3.4 void fnn::Log::Push (string *message*)

Pushes an object onto this log.

==========

William, 4/29/2015.

The message to push.

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

- FNN++/FNNLib/FNNLog.h
- FNN++/FNNLib/FNNLog.cpp

5.6 fnn::Loggable Class Reference

A loggable.

`#include <FNNLoggable.h>`

Inheritance diagram for fnn::Loggable:

Public Member Functions

- Loggable ()
  Default constructor.
- void Initialize (LogManager *lm, string name, bool verbose)
  Constructs the logger.
- void Log (string log, string message, bool verbose=false)
  Logs a message to a specific log.
- void AddLog (string name, bool verbose)
  Adds a log.
- void SetVerbose (bool verbose)
  Sets a verbose.
- bool IsVerbose ()
  Query if this object is verbose.
- std::vector< fnn::Log * > GetLogs ()
  Gets the logs.
- string GetName ()
  Gets the name.

5.6.1 Detailed Description

A loggable.

==========

William, 4/29/2015.

5.6.2 Constructor & Destructor Documentation

5.6.2.1 fnn::Loggable::Loggable (void)

Default constructor.

==========

William, 5/2/2015.

5.6.3 Member Function Documentation

5.6.3.1 void fnn::Loggable::AddLog (string *name,* bool *verbose*)

Adds a log.

==========

William, 4/29/2015.

The name.

5.6.3.2 std::vector< fnn::Log * > fnn::Loggable::GetLogs (void)

Gets the logs.

==========

William, 4/29/2015.

null if it fails, else the logs.

5.6.3.3 string fnn::Loggable::GetName (void)

Gets the name.

==========

William, 4/29/2015.

The name.

5.6.3.4 void fnn::Loggable::Initialize (LogManager * *lm,* string *name,* bool *verbose*)

Constructs the logger.

==========

==========

William, 4/29/2015.

Parameters

lm	[in,out] If non-null, the lm.
name	The name.

true to verbose.

Constructs the logger.

William, 4/29/2015.

The name.

Initializes this object.

William, 5/2/2015.

Parameters

lm	[in,out] If non-null, the lm.
name	The name.

true to verbose.

5.6.3.5 bool fnn::Loggable::IsVerbose (void)

Query if this object is verbose.

==========

William, 4/29/2015.

true if verbose, false if not.

5.6.3.6 void fnn::Loggable::Log (string *log,* string *message,* bool *verbose =* `false`)

Logs a message to a specific log.

==========

William, 4/29/2015.

Parameters

log	The log.

The message.

==========

William, 4/29/2015.

Parameters

log	The log.
message	The message.

Whether or not (upon creation of a new log the log is verbose)

5.6.3.7 void fnn::Loggable::SetVerbose (bool *verbose*)

Sets a verbose.

==========

William, 4/29/2015.

true to verbose.

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

- FNN++/FNNLib/FNNLoggable.h
- FNN++/FNNLib/FNNLoggable.cpp

5.7 fnn::LogManager Class Reference

Manager for logs.

`#include <FNNLogManager.h>`

Public Member Functions

- LogManager ()

  Default constructor.
- void Register (Loggable *logger, string loggerName, bool verbose)

  Registers a log with the LogManager.
- void Save (string directory)

  Saves the set of all loggers under a directory and a sub directory. Consider is main logger property.
- void Print (Loggable *logger, string log, string message)

  TODO: Consider adding loading functionality.

5.7.1 Detailed Description

Manager for logs.

==========

William, 4/29/2015.

5.7.2 Constructor & Destructor Documentation

5.7.2.1 fnn::LogManager::LogManager (void)

Default constructor.

==========

==========

William, 4/29/2015.

Default constructor.

William, 4/29/2015.

5.7.3 Member Function Documentation

5.7.3.1 void fnn::LogManager::Print (Loggable * *logger,* string *log,* string *message*)

TODO: Consider adding loading functionality.

Prints a message to the verbose log.

==========

William, 5/3/2015.

Parameters

logger	[in,out] If non-null, the logger.
log	The log.

The message.

5.7.3.2 void fnn::LogManager::Register (Loggable * *logger,* string *loggerName,* bool *verbose*)

Registers a log with the LogManager.

==========

William, 4/29/2015.

Parameters

logger	[in,out] The logger.
loggerName	Name of the logger.

true to verbose.

5.7.3.3 void fnn::LogManager::Save (string *dir*)

Saves the set of all loggers under a directory and a sub directory. Consider is main logger property.

==========

William, 4/29/2015.

The directory to which to save.

==========

William, 4/29/2015.

The directory to which to save.

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

- FNN++/FNNLib/FNNLogManager.h
- FNN++/FNNLib/FNNLogManager.cpp

5.8 fnn::Math Class Reference

The main mathematics helper class for FNNLIB

`#include <FNNMath.h>`

Static Public Member Functions

- static double NIntegrate (std::function< double(double)> &f, double a, double b, double eps)

  Numerically integrates any integrable function on a compact Hausdorff space. Integration occurs using Simpson's rule.
- static double NIntegrate (std::function< double(double)> &f, double a, double b)

  Numerically integrates any integrable function using Simpson's rule with auto scaling.

- static double UniformReal (double min=0.0, double max=0.0)

  Uniform real.
- static double GaussianReal (double mean=0.0, double dev=1.0)

  Gaussian real.
- static std::vector< double > PolyMult (std::vector< double > &poly1, std::vector< double > &poly2)

  A polynomial multiplication helper
- static std::function< double(double)> LERP (std::vector< std::vector< double >> &data)

  A linear interpolation algorithm
- static std::function< double(double)> LagrangeInterpolation (std::vector< std::vector< double >> &data)

  A polynomial interpolation algorithm using the Lagrange Interpolation Polynomial according to http://en.wikipedia.org/wiki/Polynomial_interpolation
- static int Factorial (int n)

  Factorial implementation.
- static std::vector< double > GaussJordan (std::vector< std::vector< double >> &matrix)

  Gauss Jordan elimination for matrices.
- static std::function< double(double)> SSpline (std::vector< std::vector< double >> &data)

  A simple spline interpolation algorithm as described in http://www.geos.ed.ac.uk/~yliu23/docs/lect_spline.pdf*. Makes the assumption that the second derivative at the boundaries is equal to 0.*
- static void DataSort (std::vector< std::vector< double >> &data)

  Data sort algorthim to sort by x-values of the data. Useful for the interpolation algorithms.
- static double Mean (std::vector< double > &data)

  Determines the mean of the input vector.
- static double StdDev (std::vector< double > &data)

  Standard deviation of the data.

5.8.1 Detailed Description

The main mathematics helper class for FNNLIB

==========

William Guss, 4/6/2015.

5.8.2 Member Function Documentation

5.8.2.1 void fnn::Math::DataSort (std::vector< std::vector< double >> & *data*) [static]

Data sort algorthim to sort by x-values of the data. Useful for the interpolation algorithms.

==========

Phillip Kuznetsov, 4/29/2015.

Parameters

data	2D vector of input data points. Each row is a point.

==========

Phillip Kuznetsov, 4/29/2015.

Parameters

data	2D vector of input data points. Each row is a point.

A 2D vector of the same points sorted.

5.8.2.2 int fnn::Math::Factorial (int *n*) [static]

Factorial implementation.

==========

William, 4/26/2015.

Parameters

n	The int to process.

An int.

5.8.2.3 double fnn::Math::GaussianReal (double *mean* = 0.0, double *dev* = 1.0) [static]

Gaussian real.

==========

William Guss, 4/11/2015.

Parameters

mean	The mean.

A double.

5.8.2.4 std::vector< double > fnn::Math::GaussJordan (std::vector< std::vector< double >> & *matrix*) [static]

Gauss Jordan elimination for matrices.

==========

Phillip Kuznetsov, 4/29/2015.

Parameters

matrix	The systems of equation augmented matrix.

A vector of the variable values solved by completed Gauss-Jordan elimination.

==========

Phillip Kuznetsov, 4/29/2015.

Parameters

matrix	The systems of equation augmented matrix. Passes by reference.

A vector of the variable values solved by completed Gauss-Jordan elimination.

5.8.2.5 std::function< double(double)> fnn::Math::LagrangeInterpolation (std::vector< std::vector< double >> & *data*) [static]

A polynomial interpolation algorithm using the Lagrange Interpolation Polynomial according to http://en.wikipedia.org/wiki/Polynomial_interpolation

==========

Phillip Kuznetsov, 4/19/2015.

Parameters

data	2D vector of input data points. Each row is a point.

A polynomial interpolation function

5.8.2.6 std::function< double(double)> fnn::Math::LERP (std::vector< std::vector< double >> & *data*) [static]

A linear interpolation algorithm

==========

Phillip Kuznetsov, 4/18/2015.

Parameters

data	2D vector of input data points

A linear interpolation function

5.8.2.7 double fnn::Math::Mean (std::vector< double > & *data*) [static]

Determines the mean of the input vector.

Phillip Kuznetsov, 5/6/2015.

Parameters

data	The data.

Returns

The mean of the data

Phillip Kuznetsov, 5/6/2015.

Parameters

data	The data.

Returns

The mean of the data.

5.8.2.8 double fnn::Math::NIntegrate (std::function< double(double)> & *f*, double *a*, double *b*, double *eps*) [static]

Numerically integrates any integrable function on a compact Hausdorff space. Integration occurs using Simpson's rule.

==========

William Guss, 4/6/2015.

Parameters

f	The function to integrate.
a	The lower bound of integration.
b	The upper bound of integration.
eps	The step-size of integration using Simpson's rule.

The result.

==========

William Guss, 4/6/2015.

Parameters

f	The function to integrate.
a	The lower bound of integration.
b	The upper bound of integration.
eps	The step-size of integration using Simpson's rule.

The result.

5.8.2.9 double fnn::Math::NIntegrate (std::function< double(double)> & *f,* double *a,* double *b*) [static]

Numerically integrates any integrable function using Simpson's rule with auto scaling.

==========

William Guss, 4/6/2015.

Parameters

f	The function to integrate.
a	The lower bound of integration.
b	The upper bound of integration.

The result.

5.8.2.10 std::vector< double > fnn::Math::PolyMult (std::vector< double > & *poly1,* std::vector< double > & *poly2*) [static]

A polynomial multiplication helper

==========

Phillip Kuznetsov, 4/19/2015.

Parameters

poly1	A vector of the coefficients. Each index is the power which x is raised to

Parameters

poly2	A vector of the second polynomial coefficients./param>

A vector of coefficients for hte polynomial

5.8.2.11 std::function< double(double)> fnn::Math::SSpline (std::vector< std::vector< double >> & *data*) [static]

A simple spline interpolation algorithm as described in http://www.geos.ed.ac.uk/~yliu23/docs/lect_spline.pdf. Makes the assumption that the second derivative at the boundaries is equal to 0.

==========

Phillip Kuznetsov, 4/29/2015.

Parameters

data	2D vector of input data points. Each row is a point.

A polynomial interpolation function

==========

Phillip Kuznetsov, 4/29/2015.

2D vector of input data points. Each row is a point.

k: the vector of three coordinates that the current spline is working with. Dimensions are (x,y)

The coefficents of the equation d^2f(k-1)/dx^2 coef[0] +d^2f(k)/dx^2 coef[1] +d^2f(k+1)/dx^2 coef[2]

The coefs index counter. The for loop goes through the row and fills them with the three coefficients of the equation set.

This is why we initialized the size to data.size()+1.

5.8.2.12 double fnn::Math::StdDev (std::vector< double > & *data*) [static]

Standard deviation of the data.

The population standard deviation of the data.

Phillip Kuznetsov, 5/6/2015.

Parameters

data	The data.

Returns

The standard deviation of the data

Phillip Kuznetsov, 5/6/2015.

Parameters

data	The data.

Returns

The standard deviation of the data.

5.8.2.13 double fnn::Math::UniformReal (double *min* = 0.0, double *max* = 0.0) [static]

Uniform real.

==========

William Guss, 4/11/2015.

Parameters

min	The minimum.
max	The maximum.

A double.

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

- FNN++/FNNLib/FNNMath.h
- FNN++/FNNLib/FNNMath.cpp

5.9 fnn::Network Class Reference

The main class of operation on the functional neural networks.

`#include <FNNNetwork.h>`

Inheritance diagram for fnn::Network:

Public Member Functions

- Network ()

  Constructs a functional neural network..
- std::function< double(double)> FeedForward (std::function< double(double)>)

  Runs the network using the fast feedforward algorithm. The algorithm caches following that described in the paper.
- double BackPropagate (std::function< double(double)>)

  Back propagate using the Super Pro Algo developed by William Guss and Patrick Chen.
- void SetActivation (Sigmoid activator)

  Sets an activation.
- void AddLayer (int x, int y)

  Adds a layer to 'y'.
- double Train (DataPoint &dp, std::vector< double > learningParameters)

  Trains the network
- void NudgeWeights ()

  Nudge weights.

Public Attributes

- Sigmoid Activator

  The primary activator type for the neural network.

5.9.1 Detailed Description

The main class of operation on the functional neural networks.

==========

William, 4/9/2015.

5.9.2 Constructor & Destructor Documentation

5.9.2.1 fnn::Network::Network ( )

Constructs a functional neural network..

==========

==========

William, 4/10/2015.

The layer count.

Constructs a functional neural network.

William, 4/10/2015.

The layer count.

5.9.3 Member Function Documentation

5.9.3.1 void fnn::Network::AddLayer (int *x*, int *y*)

Adds a layer to 'y'.

==========

William Guss, 4/12/2015.

Parameters

x	The x coordinate.

The y coordinate.

5.9.3.2 double fnn::Network::BackPropagate (std::function< double(double)>)

Back propagate using the Super Pro Algo developed by William Guss and Patrick Chen.

==========

William Guss, 5/6/2015.

Parameters

	The desired function delta .

The total integrated error over the last interval.

5.9.3.3 std::function< double(double)> fnn::Network::FeedForward (std::function< double(double)>)

Runs the network using the fast feedforward algorithm. The algorithm caches following that described in the paper.

==========

William, 4/10/2015.

Parameters

	The input, .

The ouput, F[].

5.9.3.4 void fnn::Network::NudgeWeights (void)

Nudge weights.

Phillip Kuznetsov, 5/8/2015.

5.9.3.5 void fnn::Network::SetActivation (Sigmoid *activator*)

Sets an activation.

==========

William Guss, 4/11/2015.

The activator.

5.9.3.6 double fnn::Network::Train (DataPoint & *dp,* std::vector< double > *learningParameters*)

Trains the network

Trains the network.

Phillip Kuznetsov, 5/8/2015.

Parameters

ds	The current datapoint
learning↩ Parameters	Options for controlling the learning.

Returns

A double.

Phillip Kuznetsov, 5/8/2015.

Parameters

dp	The current datapoint.
learning↩ Parameters	Options for controlling the learning.

Returns

A double.

5.9.4 Member Data Documentation

5.9.4.1 Sigmoid fnn::Network::Activator

The primary activator type for the neural network.

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

- FNN++/FNNLib/FNNNetwork.h
- FNN++/FNNLib/FNNNetwork.cpp

5.10 fnn::Sigmoid Class Reference

Public Member Functions

- Sigmoid (std::function< double(double)> f, std::function< double(double)> fprime)

  Constructs a sigmoid function
- Sigmoid ()

  Default constructor.
- double prime (double x)

  Evaluates the derivative of the activation function.
- double operator() (double x)

  Evaluates the sigmoid.

Static Public Member Functions

- static Sigmoid Linear ()

  Gets the linear sigmoid activation.
- static Sigmoid Logistic ()

  The logistic activation function.
- static Sigmoid Tanh ()

  Gets the hyperbolic tangent activation function.

5.10.1 Constructor & Destructor Documentation

5.10.1.1 fnn::Sigmoid::Sigmoid (std::function< double(double)> *g*, std::function< double(double)> *gprime*)

Constructs a sigmoid function

Default constructor. Do nothing.t

==========

William, 4/10/2015.

Parameters

f	The std::function<double(double)> to process.

The fprime.

==========

William Guss, 4/12/2015.

5.10.1.2 fnn::Sigmoid::Sigmoid ( )

Default constructor.

Default constructor. Do nothing.t

==========

William Guss, 4/12/2015.

5.10.2 Member Function Documentation

5.10.2.1 fnn::Sigmoid fnn::Sigmoid::Linear (void) [static]

Gets the linear sigmoid activation.

==========

William, 4/10/2015.

A Sigmoid.

5.10.2.2 fnn::Sigmoid fnn::Sigmoid::Logistic (void) [static]

The logistic activation function.

==========

William, 4/10/2015.

A Sigmoid.

5.10.2.3 double fnn::Sigmoid::operator() (double *x*)

Evaluates the sigmoid.

==========

William, 4/10/2015.

Parameters

x	The x coordinate.

The result of the operation.

5.10.2.4 double fnn::Sigmoid::prime (double *x*)

Evaluates the derivative of the activation function.

==========

==========

William, 4/10/2015.

Parameters

x	The x coordinate.

A double.

Evaluates the derivative of the activation function.

William, 4/10/2015.

Parameters

x	The x coordinate.

A double.

5.10.2.5 fnn::Sigmoid fnn::Sigmoid::Tanh (void) [static]

Gets the hyperbolic tangent activation function.

==========

William, 4/10/2015.

A Sigmoid.

==========

William, 4/10/2015. </remaqrks>

A Sigmoid.

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

- FNN++/FNNLib/FNNSigmoid.h
- FNN++/FNNLib/FNNSigmoid.cpp

5.11 fnn::WeightSurface Class Reference

Public Member Functions

- WeightSurface (int x, int y)
  Default constructor.
- double operator() (double i, double j)
  Function call operator.
- double GetCoefficient (int x, int y)
  Gets a coefficient.
- void Nudge ()
  Nudges the weight surface coefficients.
- int GetSizeX ()
  Gets size x coordinate.
- int GetSizeY ()
  Gets size y coordinate.

5.11.1 Constructor & Destructor Documentation

5.11.1.1 fnn::WeightSurface::WeightSurface (int *x*, int *y*)

Default constructor.

==========

==========

William Guss, 4/11/2015.

Default constructor.

William Guss, 4/11/2015.

Parameters

x	The x depth.

The y depth.

5.11.2 Member Function Documentation

5.11.2.1 double fnn::WeightSurface::GetCoefficient (int *x*, int *y*)

Gets a coefficient.

==========

William Guss, 4/11/2015.

Parameters

x	The x coordinate.
y	The y coordinate.

The coefficient.

5.11.2.2 int fnn::WeightSurface::GetSizeX (void)

Gets size x coordinate.

==========

William Guss, 4/11/2015.

The size x coordinate.

5.11.2.3 int fnn::WeightSurface::GetSizeY (void)

Gets size y coordinate.

==========

William Guss, 4/11/2015.

The size y coordinate.

5.11.2.4 void fnn::WeightSurface::Nudge (void)

Nudges the weight surface coefficients.

==========

William Guss, 4/11/2015.

5.11.2.5 double fnn::WeightSurface::operator() (double *i,* double *j*)

Function call operator.

==========

William Guss, 4/11/2015.

Parameters

i	Zero-based index of the .
j	The double to process.

The result of the operation.

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

- FNN++/FNNLib/FNNWeightSurface.h
- FNN++/FNNLib/FNNWeightSurface.cpp

Index