Fractal: Developer Manual

Generated by Doxygen 1.8.8

Tue May 19 2015 23:07:41

Contents

Chapter 1

Overview

1.1 About this manual

This manual is automatically generated by Doxygen.

1.2 Author

Kyuyeon Hwang (Signal Processing Systems Laboratory, Seoul National University, Seoul, Korea)

1.3 License

Chapter 2

Namespace Index

2.1 Namespace List

Here is a list of all namespaces with brief descriptions:

Chapter 3

Hierarchical Index

3.1 Class Hierarchy

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

Chapter 4

Class Index

4.1 Class List

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

Chapter 5

File Index

5.1 File List

Here is a list of all files with brief descriptions:

Chapter 6

Namespace Documentation

6.1 fractal Namespace Reference

The topmost namespace of libfractal.

Namespaces

- basicLayers
- cudaKernels

Classes

- class AutoOptimizer
- class BackpropArgs
- class ClassificationEvaluator
- class Connection
- class DataSet
- class DataStream
- class Engine
- class EvaluateArgs
- class Evaluator
- class InitWeightParam
- class Layer
- class LayerParam
- class Matrix
- class Mem
- class Optimizer
- class PEvent
- class Pipe
- class Probe
- class PStream
- class RegressionEvaluator
- class Rnn

  A network structure container.
- class Stream

Typedefs

- typedef float FLOAT
- typedef std::tuple
  < std::string, unsigned long > PortMap
- typedef std::list< PortMap > PortMapList

Enumerations

- enum ActType {
  ACT_BIAS, ACT_SIGMOID, ACT_TANH, ACT_SOFTPLUS,
  ACT_RECTLINEAR, ACT_LINEAR, ACT_ONE_MINUS_LINEAR, ACT_INVERSE,
  ACT_SOFTMAX }
- enum StateType { AGG_DONTCARE, AGG_SUM, AGG_MULT }

Variables

- const FLOAT NO_STATE_PENALTY = (FLOAT) -1
- static const long UNTOUCHED = -1
- static const long TOUCHED = -2
- static const long SCC_DETERMINED = -3

6.1.1 Detailed Description

The topmost namespace of libfractal.

Contains all libfractal classes.

6.1.2 Typedef Documentation

6.1.2.1 typedef float fractal::FLOAT

Definition at line 34 of file FractalCommon.h.

6.1.2.2 typedef std::tuple<std::string, unsigned long> fractal::PortMap

Definition at line 32 of file PortMap.h.

6.1.2.3 typedef std::list<PortMap> fractal::PortMapList

Definition at line 33 of file PortMap.h.

6.1.3 Enumeration Type Documentation

6.1.3.1 enum fractal::ActType

Enumerator

ACT_BIAS

ACT_SIGMOID

ACT_TANH

ACT_SOFTPLUS

ACT_RECTLINEAR
ACT_LINEAR
ACT_ONE_MINUS_LINEAR
ACT_INVERSE
ACT_SOFTMAX

Definition at line 34 of file Layer.h.

6.1.3.2 enum fractal::StateType

Enumerator

AGG_DONTCARE
AGG_SUM
AGG_MULT

Definition at line 35 of file Layer.h.

6.1.4 Variable Documentation

6.1.4.1 const FLOAT fractal::NO_STATE_PENALTY = (FLOAT) -1

Definition at line 27 of file Layer.cc.

6.1.4.2 const long fractal::SCC_DETERMINED = -3 [static]

Definition at line 36 of file Rnn.cc.

6.1.4.3 const long fractal::TOUCHED = -2 [static]

Definition at line 35 of file Rnn.cc.

6.1.4.4 const long fractal::UNTOUCHED = -1 [static]

Definition at line 34 of file Rnn.cc.

6.2 fractal::basicLayers Namespace Reference

Functions

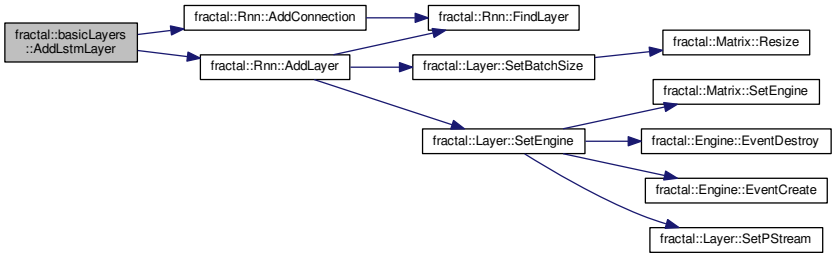
- void AddLstmLayer (Rnn &rnn, const std::string name, const std::string biasLayer, const unsigned long delayAmount, const unsigned long size, const InitWeightParam &initWeightParam, const FLOAT initForget↩ GateBias)

6.2.1 Function Documentation

6.2.1.1 void fractal::basicLayers::AddLstmLayer (Rnn & *rnn,* const std::string *name,* const std::string *biasLayer,* const unsigned long *delayAmount,* const unsigned long *size,* const InitWeightParam & *initWeightParam,* const FLOAT *initForgetGateBias*)

Definition at line 29 of file BasicLayers.cc.

Here is the call graph for this function:



6.3 fractal::cudaKernels Namespace Reference

Functions

- template<class T >
  void MemSet (T *_x, const T val, const unsigned long n, const cudaStream_t stream)
- template<class T >
  void ElemMult (const T *_x, const T *_y, T *_z, const unsigned long n, const cudaStream_t stream)
- template<class T >
  void Add (const T *_x, const T *_y, T *_z, const unsigned long n, const cudaStream_t stream)
- template<class T >
  void FuncSigmoid (const T *_x, T *_y, const unsigned long n, const cudaStream_t stream)
- template<class T >
  void FuncTanh (const T *_x, T *_y, const unsigned long n, const cudaStream_t stream)
- template<class T >
  void FuncSoftplus (const T *_x, T *_y, const unsigned long n, const cudaStream_t stream)
- template<class T >
  void FuncRectLinear (const T *_x, T *_y, const unsigned long n, const cudaStream_t stream)
- template<class T >
  void FuncSoftmax (const T *_x, T *_y, const unsigned long layerSize, const unsigned long batchSize, const cudaStream_t stream)
- template<class T >
  void FuncBoundRange (const T *_x, T *_y, const T min, const T max, const unsigned long n, const cuda↩Stream_t stream)
- template<class T >
  void FuncSigmoidDeriv (const T *_x, T *_y, const unsigned long n, const cudaStream_t stream)
- template<class T >
  void FuncTanhDeriv (const T *_x, T *_y, const unsigned long n, const cudaStream_t stream)
- template<class T >
  void FuncSoftplusDeriv (const T *_x, T *_y, const unsigned long n, const cudaStream_t stream)
- template<class T >
  void FuncRectLinearDeriv (const T *_x, T *_y, const unsigned long n, const cudaStream_t stream)
- template<class T >
  void Rmsprop (T *_newDerivs, const T *_derivs, T *_msDeriv, const T decayRate, const unsigned long n, const cudaStream_t stream)
- template<class T >
  void Adadelta (T *_deltas, const T *_derivs, T *_msDeriv, T *_msDelta, const T learningRate, const T decayRate, const unsigned long n, const cudaStream_t stream)

6.3.1 Function Documentation

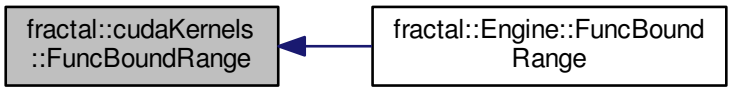
6.3.1.1 template<class T > void fractal::cudaKernels::Adadelta (T * *_deltas*, const T * *_derivs*, T * *_msDeriv*, T * *_msDelta*, const T *learningRate*, const T *decayRate*, const unsigned long *n*, const cudaStream_t *stream*)

6.3.1.2 template<class T > void fractal::cudaKernels::Add (const T * *_x*, const T * *_y*, T * *_z*, const unsigned long *n*, const cudaStream_t *stream*)

6.3.1.3 template<class T > void fractal::cudaKernels::ElemMult (const T * *_x*, const T * *_y*, T * *_z*, const unsigned long *n*, const cudaStream_t *stream*)

6.3.1.4 template<class T > void fractal::cudaKernels::FuncBoundRange (const T * *_x*, T * *_y*, const T *min*, const T *max*, const unsigned long *n*, const cudaStream_t *stream*)

Here is the caller graph for this function:



6.3.1.5 template<class T > void fractal::cudaKernels::FuncRectLinear (const T * *_x*, T * *_y*, const unsigned long *n*, const cudaStream_t *stream*)

Here is the caller graph for this function:

6.3.1.6 template<class T > void fractal::cudaKernels::FuncRectLinearDeriv (const T * *_x*, T * *_y*, const unsigned long *n*, const cudaStream_t *stream*)

6.3.1.7 template<class T > void fractal::cudaKernels::FuncSigmoid (const T * *_x*, T * *_y*, const unsigned long *n*, const cudaStream_t *stream*)

Here is the caller graph for this function:

6.3.1.8 template<class T > void fractal::cudaKernels::FuncSigmoidDeriv (const T * _x, T * _y, const unsigned long n, const cudaStream_t stream)

6.3.1.9 template<class T > void fractal::cudaKernels::FuncSoftmax (const T * _x, T * _y, const unsigned long layerSize, const unsigned long batchSize, const cudaStream_t stream)

6.3.1.10 template<class T > void fractal::cudaKernels::FuncSoftplus (const T * _x, T * _y, const unsigned long n, const cudaStream_t stream)

Here is the caller graph for this function:

6.3.1.11 template<class T > void fractal::cudaKernels::FuncSoftplusDeriv (const T * _x, T * _y, const unsigned long n, const cudaStream_t stream)

6.3.1.12 template<class T > void fractal::cudaKernels::FuncTanh (const T * _x, T * _y, const unsigned long n, const cudaStream_t stream)

Here is the caller graph for this function:

6.3.1.13 template<class T > void fractal::cudaKernels::FuncTanhDeriv (const T * _x, T * _y, const unsigned long n, const cudaStream_t stream)

6.3.1.14 template<class T > void fractal::cudaKernels::MemSet (T * _x, const T val, const unsigned long n, const cudaStream_t stream)

6.3.1.15 template<class T > void fractal::cudaKernels::Rmsprop (T * _newDerivs, const T * _derivs, T * _msDeriv, const T decayRate, const unsigned long n, const cudaStream_t stream)

Chapter 7

Class Documentation

7.1 fractal::AutoOptimizer Class Reference

`#include <AutoOptimizer.h>`

Public Member Functions

- AutoOptimizer ()
- virtual ~AutoOptimizer ()
- void Optimize (Rnn &rnn, Stream &trainStream, Stream &evalStream, Evaluator &evaluator, const PortMapList &inputPorts, const PortMapList &outputPorts, const unsigned long nTrainFramePerEpoch, const unsigned long nEvalFramePerEpoch, const unsigned long windowSize, const unsigned long stepSize)
- void SetWorkspacePath (const std::string &path)
- void SetInitLearningRate (const FLOAT val)
- void SetMinLearningRate (const FLOAT val)
- void SetMomentum (const FLOAT val)
- void SetRmsprop (const bool val)
- void SetAdadelta (const bool val)
- void SetRmsDecayRate (const FLOAT val)
- void SetMaxRetryCount (const unsigned long val)
- void SetLearningRateDecayRate (const FLOAT val)
- void SetLambdaLoss (std::function< double(Evaluator &)> lambda)
- void SetLambdaPostEval (std::function< void(Evaluator &)> lambda)
- const std::string & GetWorkspacePath ()
- const FLOAT GetInitLearningRate ()
- const FLOAT GetMinLearningRate ()
- const FLOAT GetMomentum ()
- const bool GetRmsprop ()
- const bool GetAdadelta ()
- const FLOAT GetRmsDecayRate ()
- const unsigned long GetMaxRetryCount ()
- const FLOAT GetLearningRateDecayRate ()

Protected Attributes

- std::function< double(Evaluator &)> lambdaLoss
- std::function< void(Evaluator &)> lambdaPostEval
- std::string workspacePath
- FLOAT initLearningRate

- FLOAT minLearningRate
- FLOAT momentum
- bool rmsprop
- bool adadelta
- FLOAT rmsDecayRate
- FLOAT learningRateDecayRate
- unsigned long maxRetryCount

7.1.1 Detailed Description

Definition at line 32 of file AutoOptimizer.h.

7.1.2 Constructor & Destructor Documentation

7.1.2.1 fractal::AutoOptimizer::AutoOptimizer ( )

Definition at line 28 of file AutoOptimizer.cc.

7.1.2.2 virtual fractal::AutoOptimizer::~AutoOptimizer ( ) [inline], [virtual]

Definition at line 36 of file AutoOptimizer.h.

7.1.3 Member Function Documentation

7.1.3.1 const bool fractal::AutoOptimizer::GetAdadelta ( ) [inline]

Definition at line 61 of file AutoOptimizer.h.

7.1.3.2 const FLOAT fractal::AutoOptimizer::GetInitLearningRate ( ) [inline]

Definition at line 57 of file AutoOptimizer.h.

7.1.3.3 const FLOAT fractal::AutoOptimizer::GetLearningRateDecayRate ( ) [inline]

Definition at line 64 of file AutoOptimizer.h.

7.1.3.4 const unsigned long fractal::AutoOptimizer::GetMaxRetryCount ( ) [inline]

Definition at line 63 of file AutoOptimizer.h.

7.1.3.5 const FLOAT fractal::AutoOptimizer::GetMinLearningRate ( ) [inline]

Definition at line 58 of file AutoOptimizer.h.

7.1.3.6 const FLOAT fractal::AutoOptimizer::GetMomentum ( ) [inline]

Definition at line 59 of file AutoOptimizer.h.

7.1.3.7 const FLOAT fractal::AutoOptimizer::GetRmsDecayRate ( ) [inline]

Definition at line 62 of file AutoOptimizer.h.

7.1.3.8 const bool fractal::AutoOptimizer::GetRmsprop ( ) [inline]

Definition at line 60 of file AutoOptimizer.h.

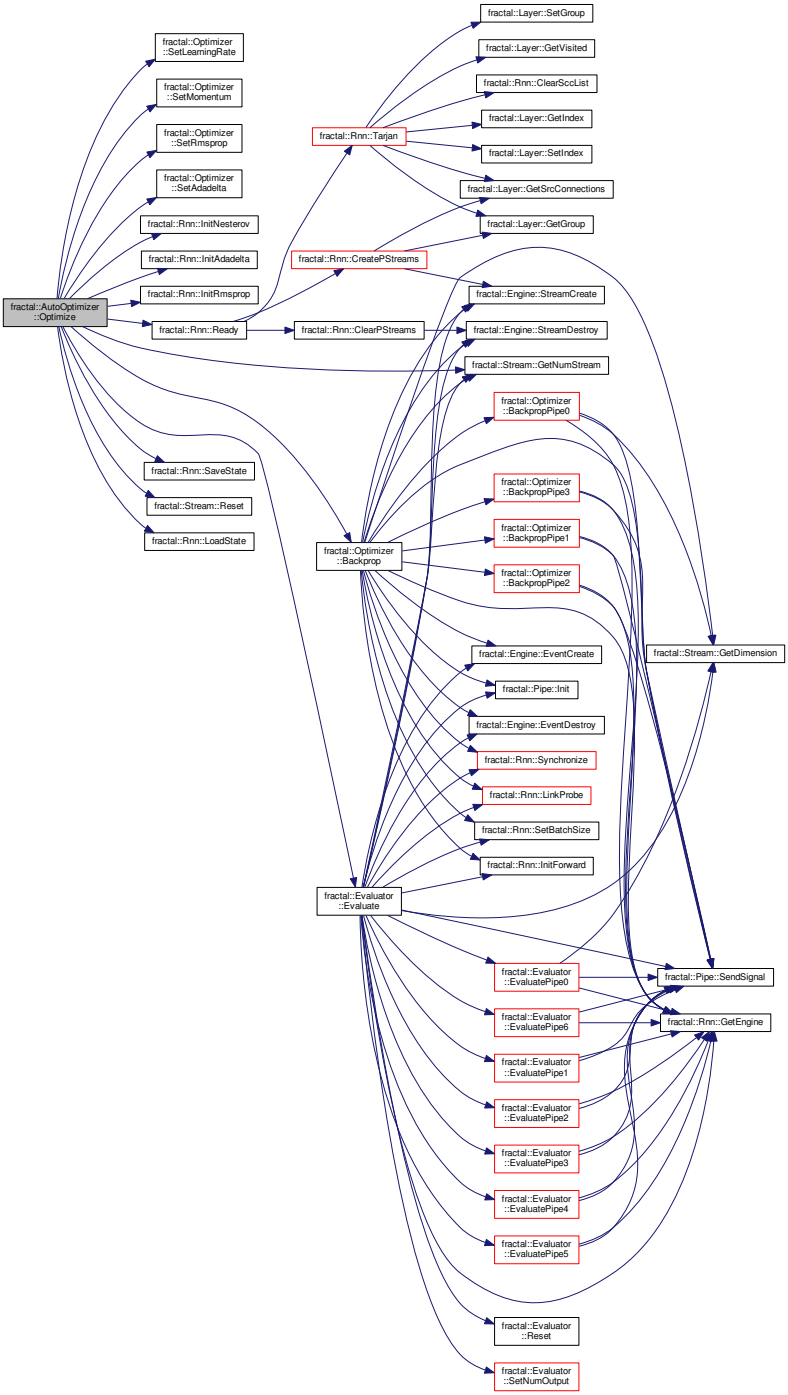
7.1.3.9 const std::string& fractal::AutoOptimizer::GetWorkspacePath ( ) [inline]

Definition at line 56 of file AutoOptimizer.h.

7.1.3.10 void fractal::AutoOptimizer::Optimize (Rnn & *rnn,* Stream & *trainStream,* Stream & *evalStream,* Evaluator & *evaluator,* const PortMapList & *inputPorts,* const PortMapList & *outputPorts,* const unsigned long *nTrainFramePerEpoch,* const unsigned long *nEvalFramePerEpoch,* const unsigned long *windowSize,* const unsigned long *stepSize*)

Definition at line 65 of file AutoOptimizer.cc.

Here is the call graph for this function:



7.1.3.11 void fractal::AutoOptimizer::SetAdadelta (const bool *val*) [inline]

Definition at line 48 of file AutoOptimizer.h.

7.1.3.12 void fractal::AutoOptimizer::SetInitLearningRate (const FLOAT *val*) [inline]

Definition at line 44 of file AutoOptimizer.h.

7.1.3.13 void fractal::AutoOptimizer::SetLambdaLoss (std::function< double(Evaluator &)> *lambda*) [inline]

Definition at line 53 of file AutoOptimizer.h.

7.1.3.14 void fractal::AutoOptimizer::SetLambdaPostEval (std::function< void(Evaluator &)> *lambda*) [inline]

Definition at line 54 of file AutoOptimizer.h.

7.1.3.15 void fractal::AutoOptimizer::SetLearningRateDecayRate (const FLOAT *val*) [inline]

Definition at line 51 of file AutoOptimizer.h.

7.1.3.16 void fractal::AutoOptimizer::SetMaxRetryCount (const unsigned long *val*) [inline]

Definition at line 50 of file AutoOptimizer.h.

7.1.3.17 void fractal::AutoOptimizer::SetMinLearningRate (const FLOAT *val*) [inline]

Definition at line 45 of file AutoOptimizer.h.

7.1.3.18 void fractal::AutoOptimizer::SetMomentum (const FLOAT *val*) [inline]

Definition at line 46 of file AutoOptimizer.h.

7.1.3.19 void fractal::AutoOptimizer::SetRmsDecayRate (const FLOAT *val*) [inline]

Definition at line 49 of file AutoOptimizer.h.

7.1.3.20 void fractal::AutoOptimizer::SetRmsprop (const bool *val*) [inline]

Definition at line 47 of file AutoOptimizer.h.

7.1.3.21 void fractal::AutoOptimizer::SetWorkspacePath (const std::string & *path*) [inline]

Definition at line 43 of file AutoOptimizer.h.

7.1.4 Member Data Documentation

7.1.4.1 bool fractal::AutoOptimizer::adadelta [protected]

Definition at line 78 of file AutoOptimizer.h.

7.1.4.2 FLOAT fractal::AutoOptimizer::initLearningRate [protected]

Definition at line 73 of file AutoOptimizer.h.

7.1.4.3 std::function<double (Evaluator &)> fractal::AutoOptimizer::lambdaLoss [protected]

Definition at line 68 of file AutoOptimizer.h.

7.1.4.4 std::function<void (Evaluator &)> fractal::AutoOptimizer::lambdaPostEval [protected]

Definition at line 69 of file AutoOptimizer.h.

7.1.4.5 FLOAT fractal::AutoOptimizer::learningRateDecayRate [protected]

Definition at line 81 of file AutoOptimizer.h.

7.1.4.6 unsigned long fractal::AutoOptimizer::maxRetryCount [protected]

Definition at line 83 of file AutoOptimizer.h.

7.1.4.7 FLOAT fractal::AutoOptimizer::minLearningRate [protected]

Definition at line 74 of file AutoOptimizer.h.

7.1.4.8 FLOAT fractal::AutoOptimizer::momentum [protected]

Definition at line 75 of file AutoOptimizer.h.

7.1.4.9 FLOAT fractal::AutoOptimizer::rmsDecayRate [protected]

Definition at line 79 of file AutoOptimizer.h.

7.1.4.10 bool fractal::AutoOptimizer::rmsprop [protected]

Definition at line 77 of file AutoOptimizer.h.

7.1.4.11 std::string fractal::AutoOptimizer::workspacePath [protected]

Definition at line 71 of file AutoOptimizer.h.

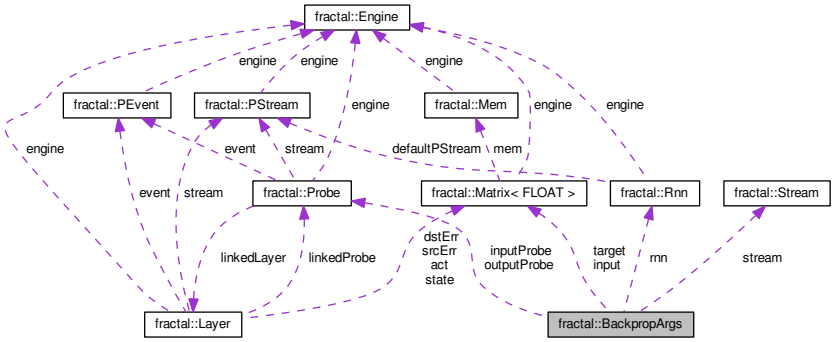
The documentation for this class was generated from the following files:

- src/util/AutoOptimizer.h
- src/util/AutoOptimizer.cc

7.2 fractal::BackpropArgs Class Reference

`#include <Optimizer.h>`

Collaboration diagram for fractal::BackpropArgs:



Public Attributes

- Rnn * rnn
- Stream * stream
- unsigned long numFrame
- unsigned long nStream
- unsigned long batchSize
- unsigned long frameStep
- unsigned long nInput
- unsigned long nOutput
- Probe * inputProbe
- Probe * outputProbe
- unsigned long * inputChannel
- unsigned long * outputChannel
- Matrix< FLOAT > * input
- Matrix< FLOAT > * target

7.2.1 Detailed Description

Definition at line 31 of file Optimizer.h.

7.2.2 Member Data Documentation

7.2.2.1 unsigned long fractal::BackpropArgs::batchSize

Definition at line 39 of file Optimizer.h.

7.2.2.2 unsigned long fractal::BackpropArgs::frameStep

Definition at line 40 of file Optimizer.h.

7.2.2.3 Matrix<FLOAT>* fractal::BackpropArgs::input

Definition at line 50 of file Optimizer.h.

7.2.2.4 unsigned long* fractal::BackpropArgs::inputChannel

Definition at line 47 of file Optimizer.h.

7.2.2.5 Probe* fractal::BackpropArgs::inputProbe

Definition at line 44 of file Optimizer.h.

7.2.2.6 unsigned long fractal::BackpropArgs::nInput

Definition at line 41 of file Optimizer.h.

7.2.2.7 unsigned long fractal::BackpropArgs::nOutput

Definition at line 42 of file Optimizer.h.

7.2.2.8 unsigned long fractal::BackpropArgs::nStream

Definition at line 38 of file Optimizer.h.

7.2.2.9 unsigned long fractal::BackpropArgs::numFrame

Definition at line 37 of file Optimizer.h.

7.2.2.10 unsigned long* fractal::BackpropArgs::outputChannel

Definition at line 48 of file Optimizer.h.

7.2.2.11 Probe* fractal::BackpropArgs::outputProbe

Definition at line 45 of file Optimizer.h.

7.2.2.12 Rnn* fractal::BackpropArgs::rnn

Definition at line 34 of file Optimizer.h.

7.2.2.13 Stream* fractal::BackpropArgs::stream

Definition at line 35 of file Optimizer.h.

7.2.2.14 Matrix<FLOAT>* fractal::BackpropArgs::target

Definition at line 51 of file Optimizer.h.

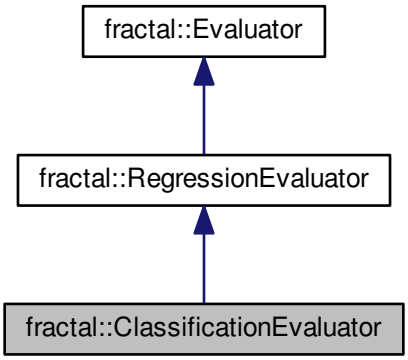
The documentation for this class was generated from the following file:

- src/util/Optimizer.h

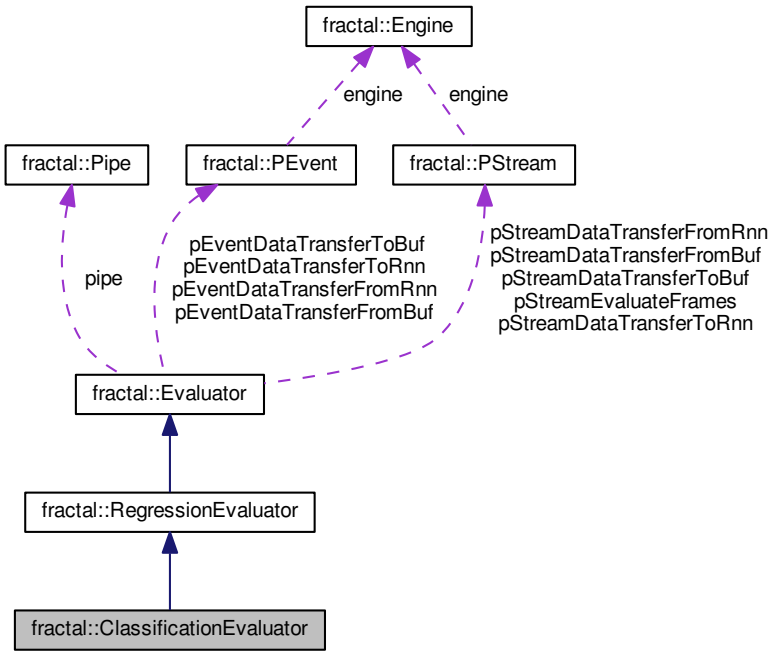
7.3 fractal::ClassificationEvaluator Class Reference

`#include <ClassificationEvaluator.h>`

Inheritance diagram for fractal::ClassificationEvaluator:



Collaboration diagram for fractal::ClassificationEvaluator:



Public Member Functions

- ClassificationEvaluator ()
- virtual const double GetLoss (const unsigned long outputIdx) const
- const double GetAverageCrossEntropy (const unsigned long outputIdx) const
- const double GetFrameErrorRate (const unsigned long outputIdx) const
- const unsigned long GetFrameErrorCount (const unsigned long outputIdx) const

Protected Member Functions

- virtual void Reset ()
- virtual void EvaluateFrames (const unsigned long outputIdx, Matrix< FLOAT > &target, Matrix< FLOAT > &output, const unsigned long nStream, PStream &stream)
- virtual void MemAlloc ()

Protected Attributes

- std::vector< unsigned long > nError
- std::vector< double > ceSum

Additional Inherited Members

7.3.1 Detailed Description

Definition at line 28 of file ClassificationEvaluator.h.

7.3.2 Constructor & Destructor Documentation

7.3.2.1 fractal::ClassificationEvaluator::ClassificationEvaluator ( ) [inline]

Definition at line 31 of file ClassificationEvaluator.h.

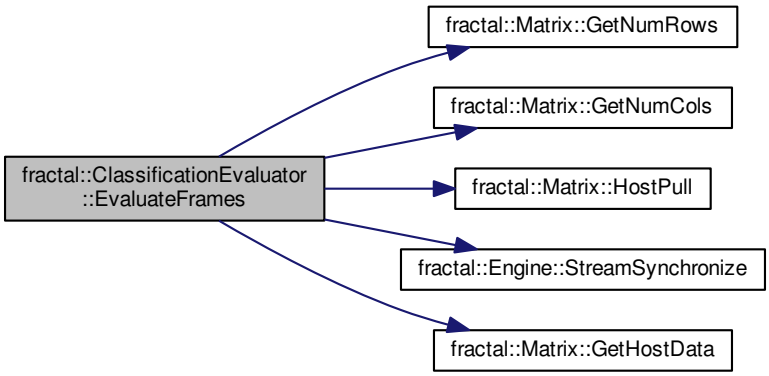
7.3.3 Member Function Documentation

7.3.3.1 void fractal::ClassificationEvaluator::EvaluateFrames (const unsigned long *outputIdx*, Matrix< FLOAT > & *target*, Matrix< FLOAT > & *output*, const unsigned long *nStream*, PStream & *stream*) [protected], [virtual]

Reimplemented from fractal::RegressionEvaluator.

Definition at line 70 of file ClassificationEvaluator.cc.

Here is the call graph for this function:



7.3.3.2 const double fractal::ClassificationEvaluator::GetAverageCrossEntropy (const unsigned long *outputIdx*) const

Definition at line 32 of file ClassificationEvaluator.cc.

Here is the caller graph for this function:

7.3.3.3 const unsigned long fractal::ClassificationEvaluator::GetFrameErrorCount (const unsigned long *outputIdx*) const

Definition at line 48 of file ClassificationEvaluator.cc.

7.3.3.4 const double fractal::ClassificationEvaluator::GetFrameErrorRate (const unsigned long *outputIdx*) const

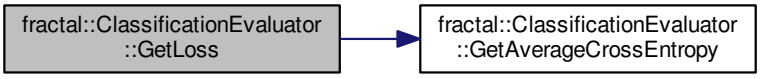
Definition at line 40 of file ClassificationEvaluator.cc.

7.3.3.5 const double fractal::ClassificationEvaluator::GetLoss (const unsigned long *outputIdx*) const [virtual]

Reimplemented from fractal::RegressionEvaluator.

Definition at line 26 of file ClassificationEvaluator.cc.

Here is the call graph for this function:

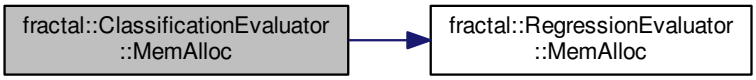


7.3.3.6 void fractal::ClassificationEvaluator::MemAlloc ( ) [protected], [virtual]

Reimplemented from fractal::RegressionEvaluator.

Definition at line 148 of file ClassificationEvaluator.cc.

Here is the call graph for this function:

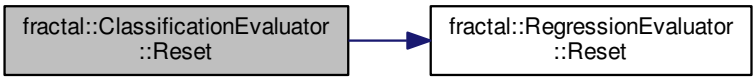


7.3.3.7 void fractal::ClassificationEvaluator::Reset ( ) [protected], [virtual]

Reimplemented from fractal::RegressionEvaluator.

Definition at line 56 of file ClassificationEvaluator.cc.

Here is the call graph for this function:



7.3.4 Member Data Documentation

7.3.4.1 std::vector<double> fractal::ClassificationEvaluator::ceSum [protected]

Definition at line 46 of file ClassificationEvaluator.h.

7.3.4.2 std::vector<unsigned long> fractal::ClassificationEvaluator::nError [protected]

Definition at line 45 of file ClassificationEvaluator.h.

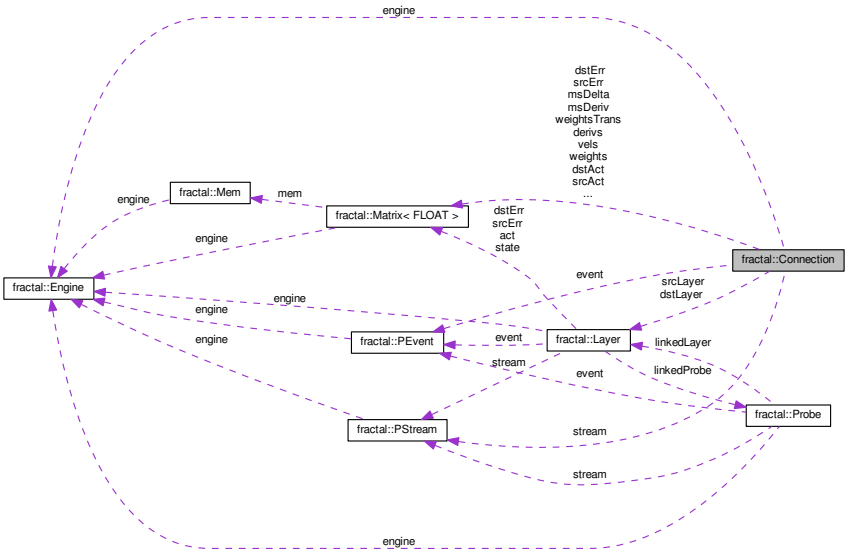
The documentation for this class was generated from the following files:

- src/util/ClassificationEvaluator.h
- src/util/ClassificationEvaluator.cc

7.4 fractal::Connection Class Reference

`#include <Connection.h>`

Collaboration diagram for fractal::Connection:



Public Member Functions

- Connection (Layer *const from, Layer *const to, const unsigned long delayAmount, const bool isIdentity)
- virtual ~Connection ()
- void SetEngine (Engine *const engine, PStream *const stream)
- void SetBatchSize (const unsigned long batchSize)
- void UnlinkMatrices ()
- void InitWeights (const InitWeightParam &param)
- void InitAdadelta (const FLOAT decayRate)
- void InitNesterov ()
- void InitRmsprop (const FLOAT decayRate)
- void InitErr (const unsigned long batchFrom, const unsigned long batchTo)
- void Forward (const unsigned long batchFrom, const unsigned long batchTo, const unsigned long nStream)
- void UpdateDstErr (const unsigned long batchFrom, const unsigned long batchTo)
- void Backward (const unsigned long batchFrom, const unsigned long batchTo, const unsigned long nStream)
- void UpdateWeights (const unsigned long batchFrom, const unsigned long batchTo, const unsigned long nFrame, const FLOAT rate, const FLOAT momentum, const bool adaptiveRates, const bool rmsprop)
- const bool IsDelayed () const
- const bool IsIdentity () const

- Layer ∗const GetSrcLayer () const
- Layer ∗const GetDstLayer () const
- void SetPStream (PStream ∗const stream)
- PStream & GetPStream ()
- void EventRecord ()
- void StreamWaitEvent (PStream &stream)
- void ForwardWait ()
- void BackwardWait ()
- void SaveState (const std::string &filename)
- void LoadState (const std::string &filename)
- const unsigned long GetNumWeights ()

Protected Member Functions

- void TransposeWeightMatrix ()

Protected Attributes

- Engine ∗ engine
- bool _identity
- unsigned long delayAmount
- Layer ∗ srcLayer
- Layer ∗ dstLayer
- unsigned long batchSize
- FLOAT rmsDecayRate
- Matrix< FLOAT > weights
- Matrix< FLOAT > weightsTrans
- bool weightsTransValid
- Matrix< FLOAT > vels
- Matrix< FLOAT > derivs
- Matrix< FLOAT > msDeriv
- Matrix< FLOAT > msDelta
- Matrix< FLOAT > dstAct
- Matrix< FLOAT > srcAct
- Matrix< FLOAT > dstErr
- Matrix< FLOAT > srcErr
- PStream ∗ stream
- PEvent event
- friend Layer

7.4.1 Detailed Description

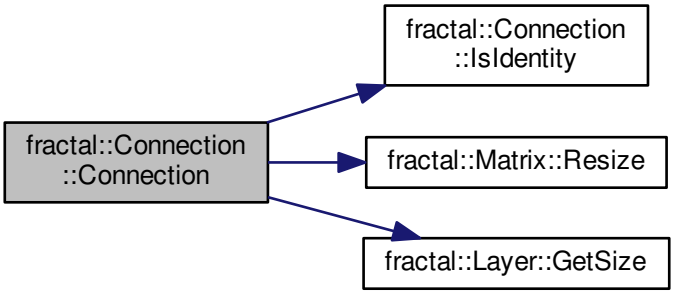
Definition at line 36 of file Connection.h.

7.4.2 Constructor & Destructor Documentation

7.4.2.1 fractal::Connection::Connection (Layer ∗const *from*, Layer ∗const *to*, const unsigned long *delayAmount*, const bool *isIdentity*)

Definition at line 29 of file Connection.cc.

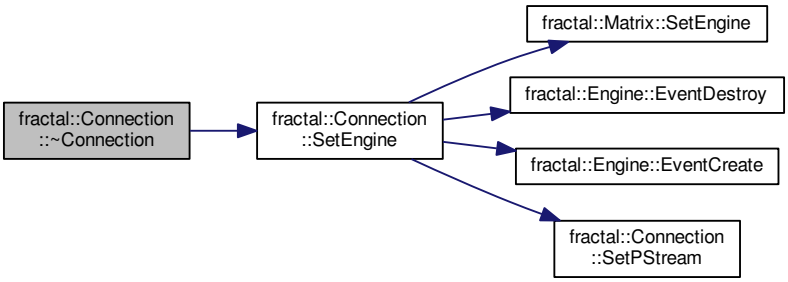
Here is the call graph for this function:



7.4.2.2 fractal::Connection::~Connection ( ) [virtual]

Definition at line 54 of file Connection.cc.

Here is the call graph for this function:

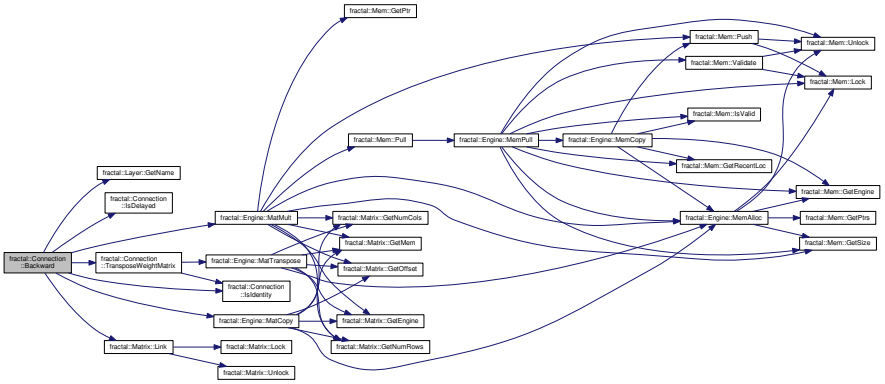


7.4.3 Member Function Documentation

7.4.3.1 void fractal::Connection::Backward (const unsigned long *batchFrom,* const unsigned long *batchTo,* const unsigned long *nStream*)

Definition at line 264 of file Connection.cc.

Here is the call graph for this function:

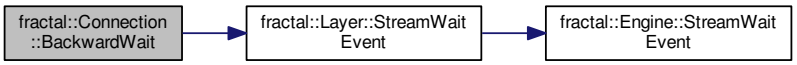


Here is the caller graph for this function:

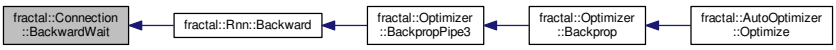
7.4.3.2 void fractal::Connection::BackwardWait ( )

Definition at line 400 of file Connection.cc.

Here is the call graph for this function:



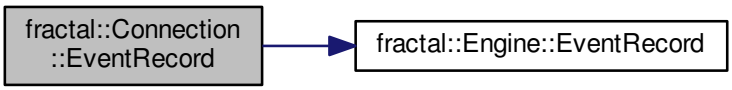
Here is the caller graph for this function:



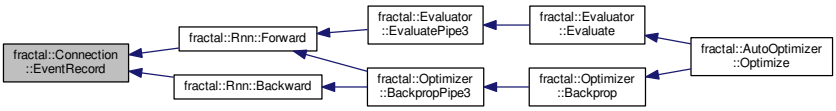
7.4.3.3 void fractal::Connection::EventRecord ( )

Definition at line 380 of file Connection.cc.

Here is the call graph for this function:



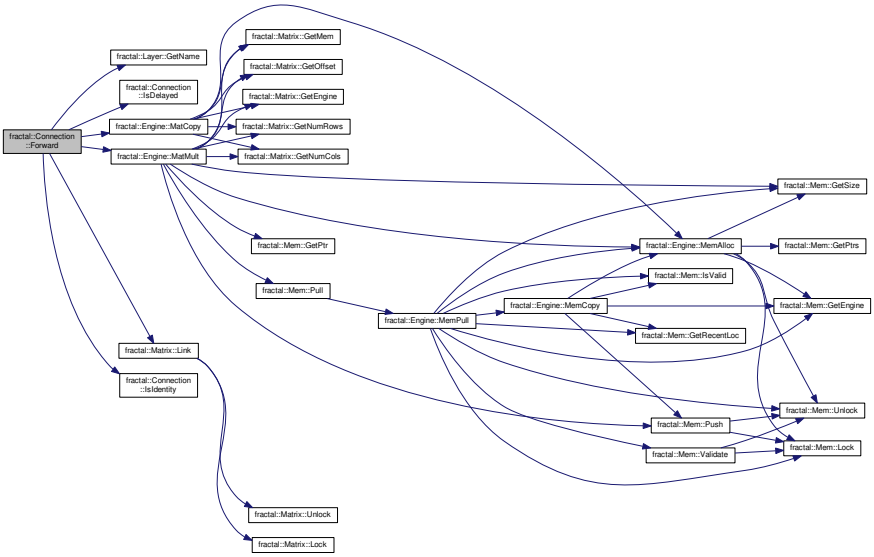
Here is the caller graph for this function:



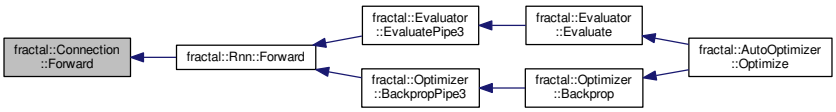
7.4.3.4 void fractal::Connection::Forward (const unsigned long *batchFrom,* const unsigned long *batchTo,* const unsigned long *nStream*)

Definition at line 195 of file Connection.cc.

Here is the call graph for this function:



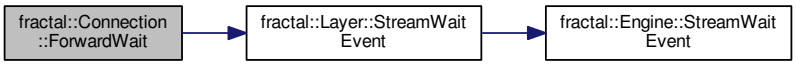
Here is the caller graph for this function:



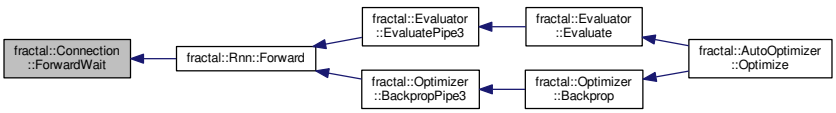
7.4.3.5 void fractal::Connection::ForwardWait ( )

Definition at line 394 of file Connection.cc.

Here is the call graph for this function:



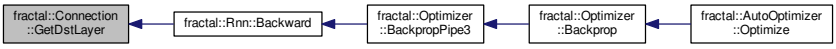
Here is the caller graph for this function:



7.4.3.6 Layer* const fractal::Connection::GetDstLayer ( ) const [inline]

Definition at line 65 of file Connection.h.

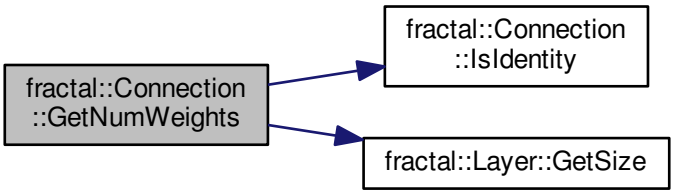
Here is the caller graph for this function:



7.4.3.7 const unsigned long fractal::Connection::GetNumWeights ( )

Definition at line 500 of file Connection.cc.

Here is the call graph for this function:



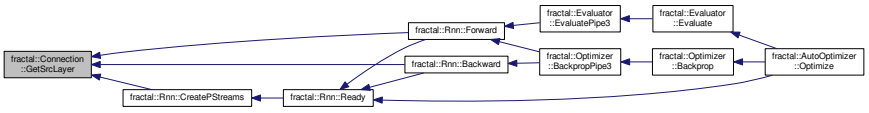
7.4.3.8 PStream & fractal::Connection::GetPStream ( )

Definition at line 373 of file Connection.cc.

7.4.3.9 Layer* const fractal::Connection::GetSrcLayer ( ) const [inline]

Definition at line 64 of file Connection.h.

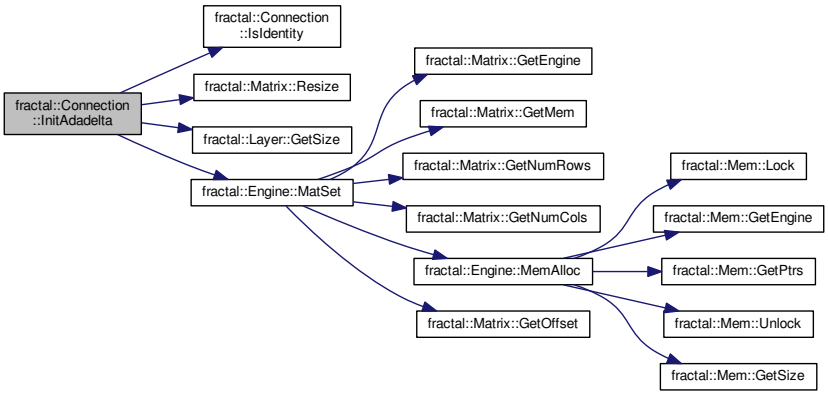
Here is the caller graph for this function:



7.4.3.10 void fractal::Connection::InitAdadelta (const FLOAT *decayRate*)

Definition at line 138 of file Connection.cc.

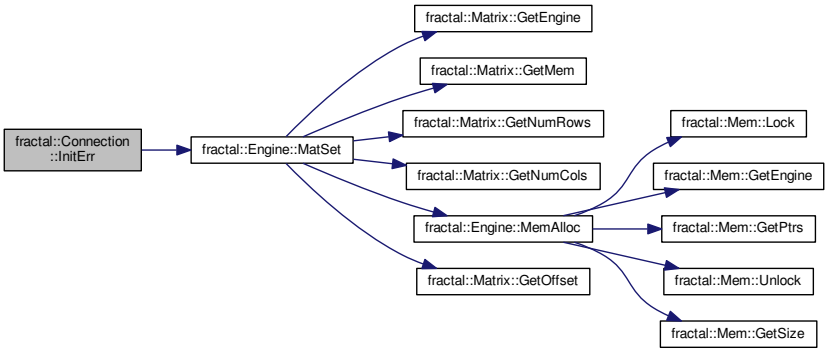
Here is the call graph for this function:



7.4.3.11 void fractal::Connection::InitErr (const unsigned long *batchFrom,* const unsigned long *batchTo*)

Definition at line 183 of file Connection.cc.

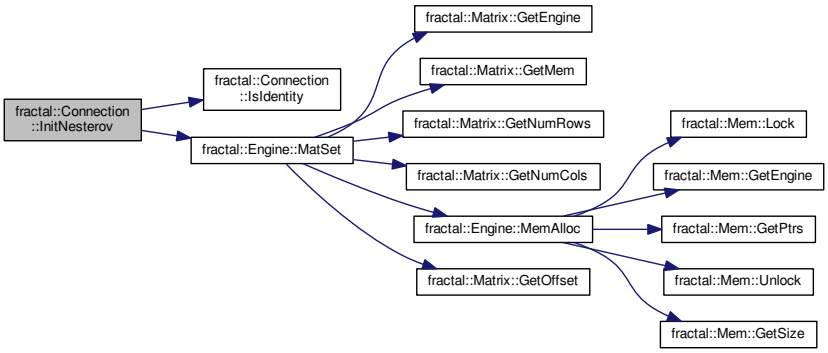
Here is the call graph for this function:



7.4.3.12 void fractal::Connection::InitNesterov ( )

Definition at line 156 of file Connection.cc.

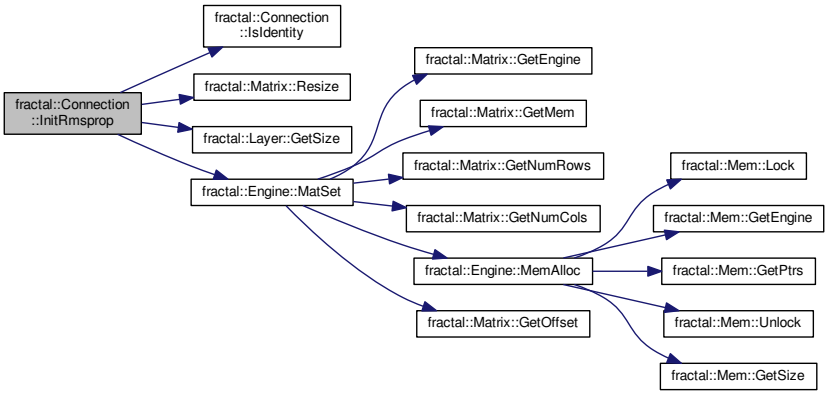
Here is the call graph for this function:



7.4.3.13 void fractal::Connection::InitRmsprop (const FLOAT *decayRate*)

Definition at line 167 of file Connection.cc.

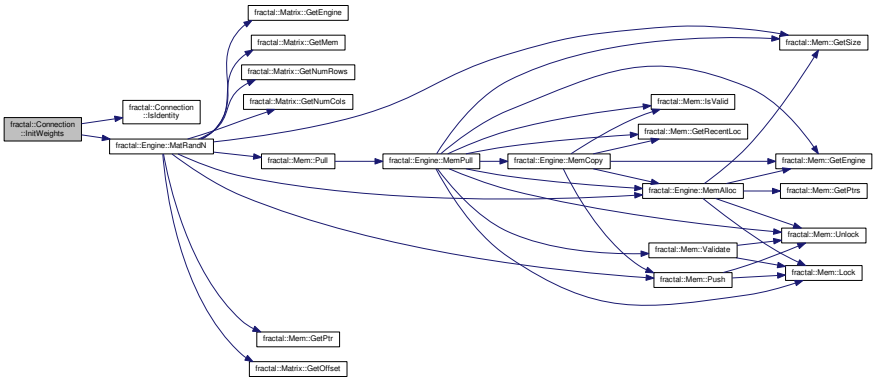
Here is the call graph for this function:



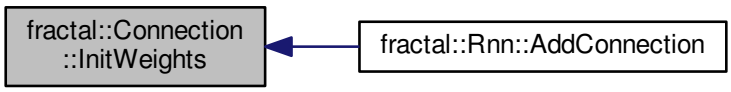
7.4.3.14 void fractal::Connection::InitWeights (const InitWeightParam & *param*)

Definition at line 125 of file Connection.cc.

Here is the call graph for this function:



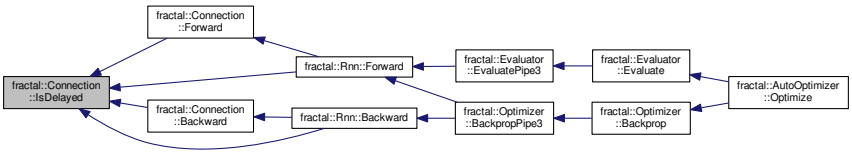
Here is the caller graph for this function:



7.4.3.15 const bool fractal::Connection::IsDelayed ( ) const [inline]

Definition at line 62 of file Connection.h.

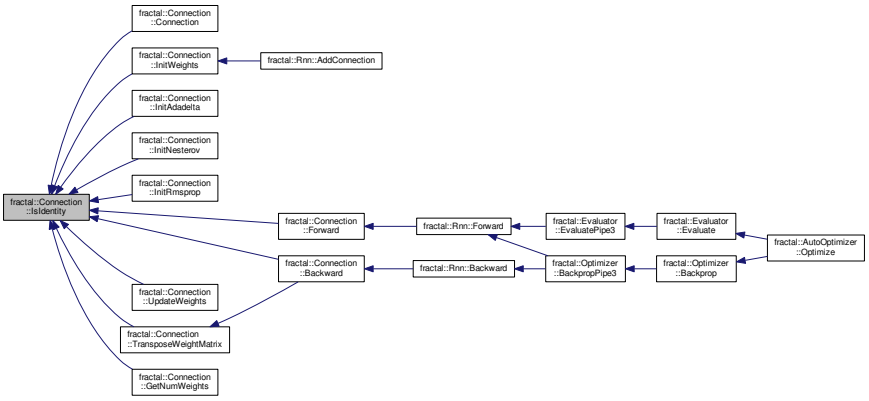
Here is the caller graph for this function:



7.4.3.16 const bool fractal::Connection::IsIdentity ( ) const [inline]

Definition at line 63 of file Connection.h.

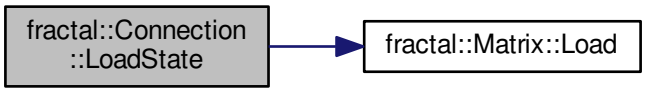
Here is the caller graph for this function:



7.4.3.17 void fractal::Connection::LoadState (const std::string & *filename*)

Definition at line 446 of file Connection.cc.

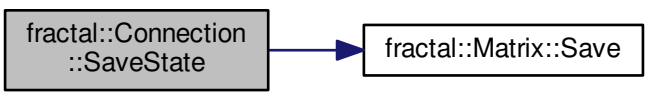
Here is the call graph for this function:



7.4.3.18 void fractal::Connection::SaveState (const std::string & *filename*)

Definition at line 419 of file Connection.cc.

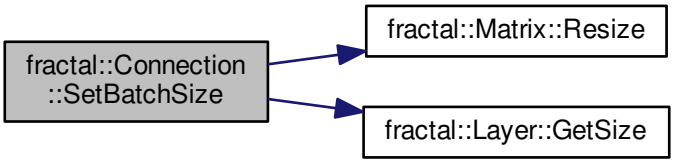
Here is the call graph for this function:



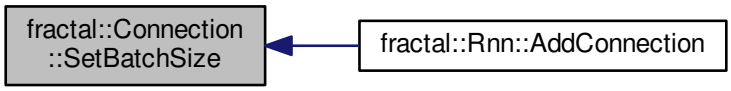
7.4.3.19 void fractal::Connection::SetBatchSize (const unsigned long *batchSize*)

Definition at line 93 of file Connection.cc.

Here is the call graph for this function:



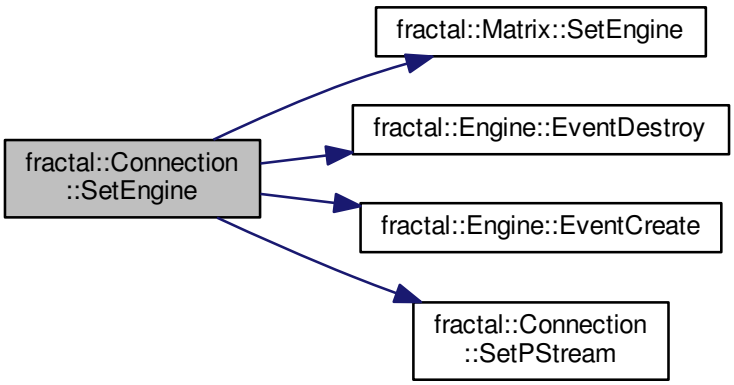
Here is the caller graph for this function:



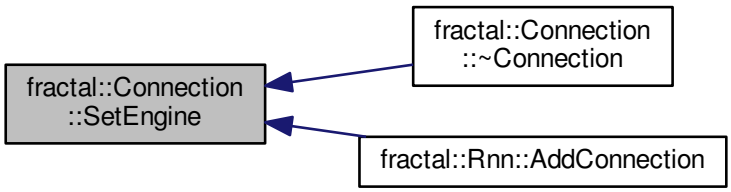
7.4.3.20 void fractal::Connection::SetEngine (Engine ∗const *engine,* PStream ∗const *stream*)

Definition at line 60 of file Connection.cc.

Here is the call graph for this function:



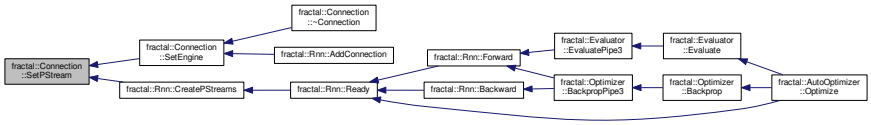
Here is the caller graph for this function:



7.4.3.21 void fractal::Connection::SetPStream (PStream ∗const *stream*)

Definition at line 366 of file Connection.cc.

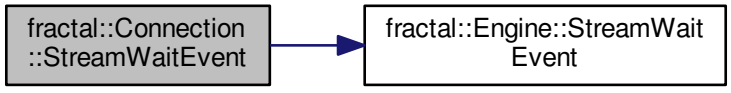
Here is the caller graph for this function:



7.4.3.22 void fractal::Connection::StreamWaitEvent (PStream & *stream*)

Definition at line 387 of file Connection.cc.

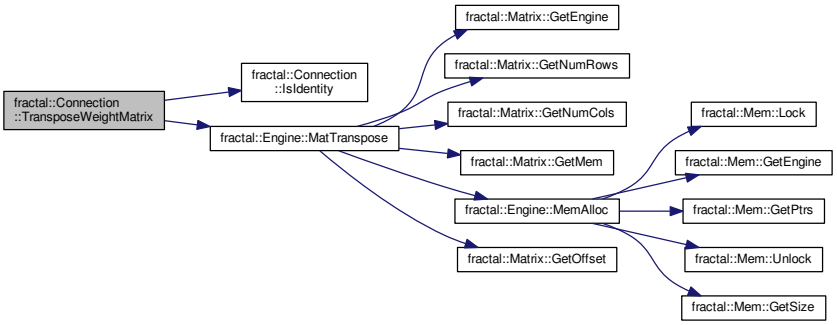
Here is the call graph for this function:



7.4.3.23 void fractal::Connection::TransposeWeightMatrix ( ) [protected]

Definition at line 406 of file Connection.cc.

Here is the call graph for this function:

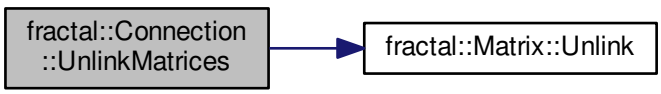


Here is the caller graph for this function:

7.4.3.24 void fractal::Connection::UnlinkMatrices ( )

Definition at line 108 of file Connection.cc.

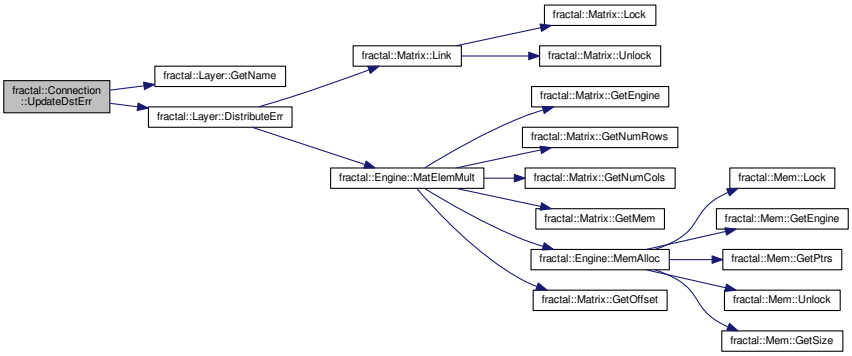
Here is the call graph for this function:



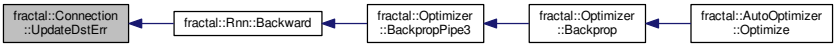
7.4.3.25 void fractal::Connection::UpdateDstErr (const unsigned long *batchFrom,* const unsigned long *batchTo*)

Definition at line 248 of file Connection.cc.

Here is the call graph for this function:



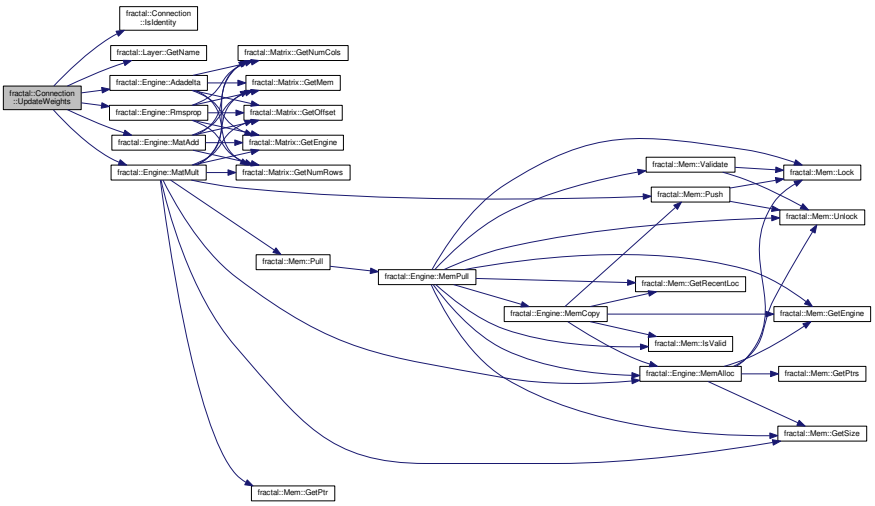
Here is the caller graph for this function:



7.4.3.26 void fractal::Connection::UpdateWeights (const unsigned long *batchFrom,* const unsigned long *batchTo,* const unsigned long *nFrame,* const FLOAT *rate,* const FLOAT *momentum,* const bool *adaptiveRates,* const bool *rmsprop*)

Definition at line 316 of file Connection.cc.

Here is the call graph for this function:



7.4.4 Member Data Documentation

7.4.4.1 bool fractal::Connection::_identity [protected]

Definition at line 86 of file Connection.h.

7.4.4.2 unsigned long fractal::Connection::batchSize [protected]

Definition at line 91 of file Connection.h.

7.4.4.3 unsigned long fractal::Connection::delayAmount [protected]

Definition at line 87 of file Connection.h.

7.4.4.4 Matrix<FLOAT> fractal::Connection::derivs [protected]

Definition at line 99 of file Connection.h.

7.4.4.5 Matrix<FLOAT> fractal::Connection::dstAct [protected]

Definition at line 101 of file Connection.h.

7.4.4.6 Matrix<FLOAT> fractal::Connection::dstErr [protected]

Definition at line 102 of file Connection.h.

7.4.4.7 Layer * fractal::Connection::dstLayer [protected]

Definition at line 89 of file Connection.h.

7.4.4.8 Engine* fractal::Connection::engine [protected]

Definition at line 84 of file Connection.h.

7.4.4.9 PEvent fractal::Connection::event [protected]

Definition at line 105 of file Connection.h.

7.4.4.10 friend fractal::Connection::Layer [protected]

Definition at line 107 of file Connection.h.

7.4.4.11 Matrix<FLOAT> fractal::Connection::msDelta [protected]

Definition at line 100 of file Connection.h.

7.4.4.12 Matrix<FLOAT> fractal::Connection::msDeriv [protected]

Definition at line 99 of file Connection.h.

7.4.4.13 FLOAT fractal::Connection::rmsDecayRate [protected]

Definition at line 93 of file Connection.h.

7.4.4.14 Matrix<FLOAT> fractal::Connection::srcAct [protected]

Definition at line 101 of file Connection.h.

7.4.4.15 Matrix<FLOAT> fractal::Connection::srcErr [protected]

Definition at line 102 of file Connection.h.

7.4.4.16 Layer* fractal::Connection::srcLayer [protected]

Definition at line 89 of file Connection.h.

7.4.4.17 PStream* fractal::Connection::stream [protected]

Definition at line 104 of file Connection.h.

7.4.4.18 Matrix<FLOAT> fractal::Connection::vels [protected]

Definition at line 98 of file Connection.h.

7.4.4.19 Matrix<FLOAT> fractal::Connection::weights [protected]

Definition at line 95 of file Connection.h.

7.4.4.20 Matrix<FLOAT> fractal::Connection::weightsTrans [protected]

Definition at line 95 of file Connection.h.

7.4.4.21 bool fractal::Connection::weightsTransValid [protected]

Definition at line 96 of file Connection.h.

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

- src/core/Connection.h
- src/core/Connection.cc

7.5 fractal::DataSet Class Reference

`#include <DataSet.h>`

Public Member Functions

- virtual const unsigned long GetNumChannel () const =0
- virtual const unsigned long GetDimension (const unsigned long channelIdx) const =0
- virtual const unsigned long GetNumSeq () const =0

- virtual const unsigned long GetNumFrame (const unsigned long seqIdx) const =0
- virtual void GetFrameData (const unsigned long seqIdx, const unsigned long channelIdx, const unsigned long frameIdx, FLOAT *const frame)=0

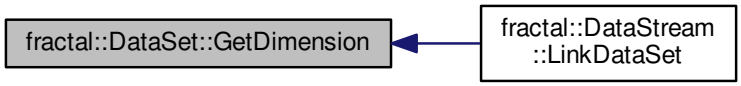
7.5.1 Detailed Description

Definition at line 26 of file DataSet.h.

7.5.2 Member Function Documentation

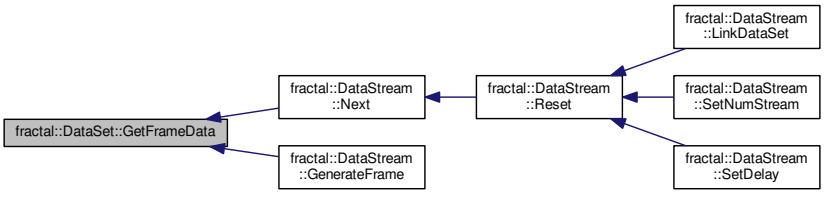
7.5.2.1 virtual const unsigned long fractal::DataSet::GetDimension (const unsigned long *channelIdx*) const [pure virtual]

Here is the caller graph for this function:



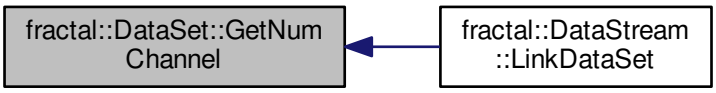
7.5.2.2 virtual void fractal::DataSet::GetFrameData (const unsigned long *seqIdx,* const unsigned long *channelIdx,* const unsigned long *frameIdx,* FLOAT *const *frame*) [pure virtual]

Here is the caller graph for this function:



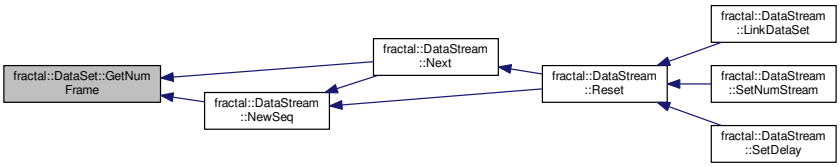
7.5.2.3 virtual const unsigned long fractal::DataSet::GetNumChannel ( ) const [pure virtual]

Here is the caller graph for this function:



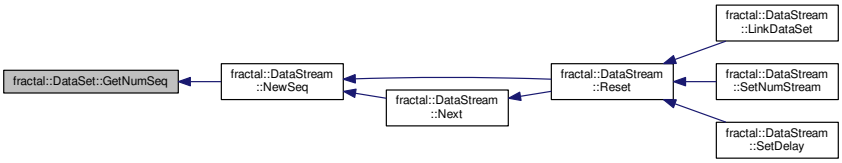
7.5.2.4 virtual const unsigned long fractal::DataSet::GetNumFrame (const unsigned long *seqIdx*) const [pure virtual]

Here is the caller graph for this function:



7.5.2.5 virtual const unsigned long fractal::DataSet::GetNumSeq ( ) const [pure virtual]

Here is the caller graph for this function:



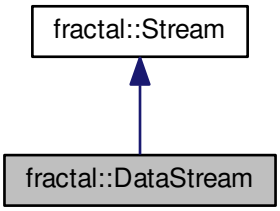
The documentation for this class was generated from the following file:

- src/util/DataSet.h

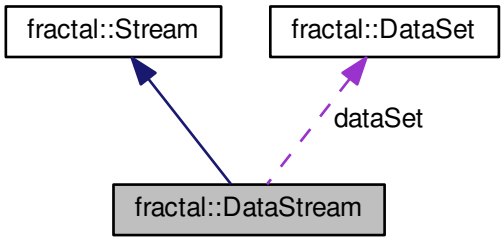
7.6 fractal::DataStream Class Reference

`#include <DataStream.h>`

Inheritance diagram for fractal::DataStream:



Collaboration diagram for fractal::DataStream:



Public Member Functions

- DataStream ()
- void SetNumStream (const unsigned long nStream)
- const unsigned long GetNumStream () const
- const unsigned long GetNumChannel () const
- const unsigned long GetDimension (const unsigned long channelIdx) const
- void Reset ()
- void Next (const unsigned long streamIdx)
- void GenerateFrame (const unsigned long streamIdx, const unsigned long channelIdx, FLOAT ∗const frame)
- void SetDelay (const unsigned long channelIdx, const unsigned long delay)
- void LinkDataSet (DataSet ∗dataSet)
- void UnlinkDataSet ()
- void SetRandomSeed (unsigned long long seed)

Protected Member Functions

- void Alloc ()
- void NewSeq (const unsigned long streamIdx)

Protected Attributes

- unsigned long nStream
- unsigned long nChannel
- std::vector< unsigned long > dim
- std::vector< unsigned long > delay
- std::vector< unsigned long > seqIdx
- std::vector< unsigned long > frameIdx
- std::vector< std::vector < unsigned long > > bufIdx
- std::vector< std::vector < std::vector< FLOAT > > > buf
- DataSet * dataSet
- std::mt19937_64 randGen

7.6.1 Detailed Description

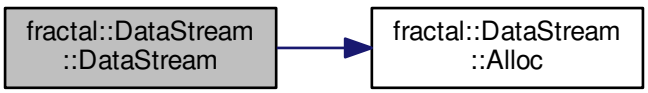
Definition at line 35 of file DataStream.h.

7.6.2 Constructor & Destructor Documentation

7.6.2.1 fractal::DataStream::DataStream ( )

Definition at line 28 of file DataStream.cc.

Here is the call graph for this function:

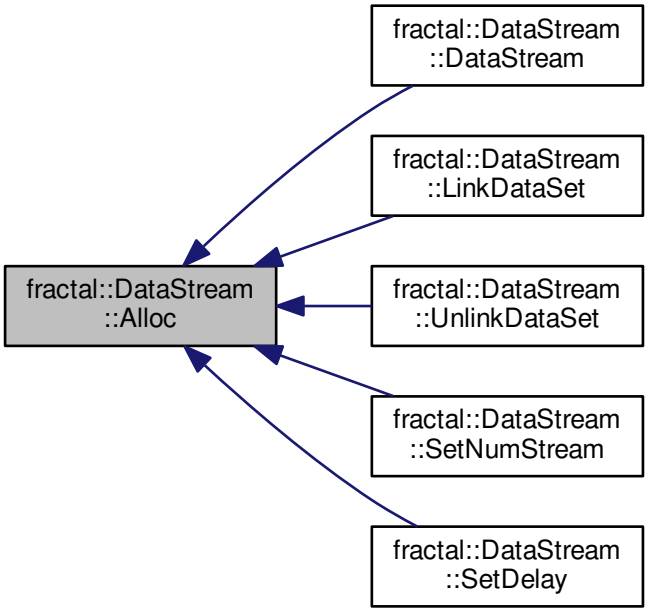


7.6.3 Member Function Documentation

7.6.3.1 void fractal::DataStream::Alloc ( ) [protected]

Definition at line 75 of file DataStream.cc.

Here is the caller graph for this function:

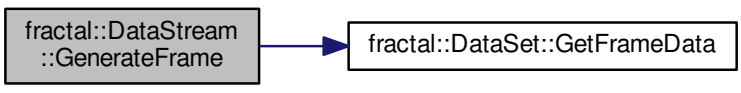


7.6.3.2 void fractal::DataStream::GenerateFrame (const unsigned long *streamIdx,* const unsigned long *channelIdx,* FLOAT ∗const *frame*) [virtual]

Implements fractal::Stream.

Definition at line 204 of file DataStream.cc.

Here is the call graph for this function:



7.6.3.3 const unsigned long fractal::DataStream::GetDimension (const unsigned long *channelIdx*) const [virtual]

Implements fractal::Stream.

Definition at line 130 of file DataStream.cc.

7.6.3.4 const unsigned long fractal::DataStream::GetNumChannel ( ) const [virtual]

Implements fractal::Stream.

Definition at line 124 of file DataStream.cc.

7.6.3.5 const unsigned long fractal::DataStream::GetNumStream ( ) const [virtual]

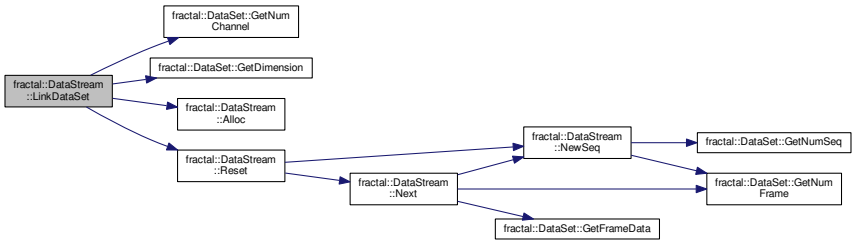
Implements fractal::Stream.

Definition at line 118 of file DataStream.cc.

7.6.3.6 void fractal::DataStream::LinkDataSet (DataSet * *dataSet*)

Definition at line 38 of file DataStream.cc.

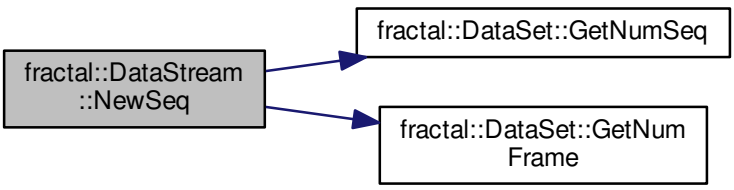
Here is the call graph for this function:



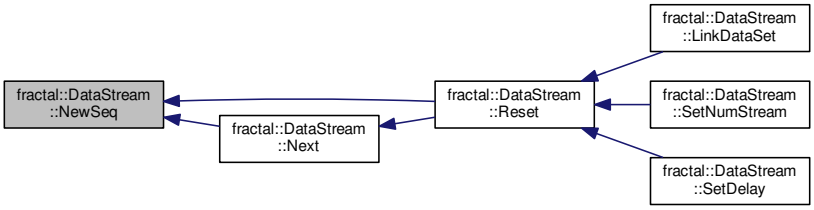
7.6.3.7 void fractal::DataStream::NewSeq (const unsigned long *streamIdx*) [protected]

Definition at line 241 of file DataStream.cc.

Here is the call graph for this function:



Here is the caller graph for this function:

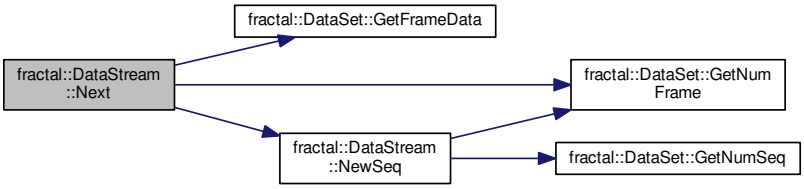


7.6.3.8 void fractal::DataStream::Next (const unsigned long *streamIdx*) [virtual]

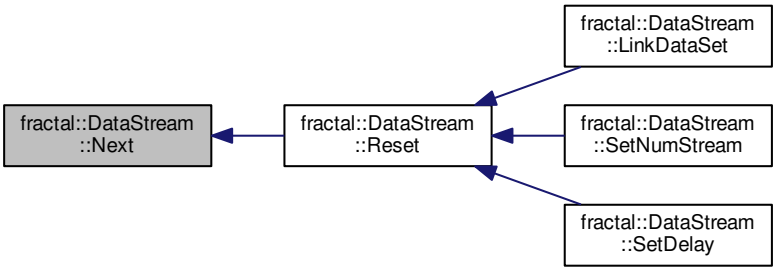
Implements fractal::Stream.

Definition at line 169 of file DataStream.cc.

Here is the call graph for this function:



Here is the caller graph for this function:

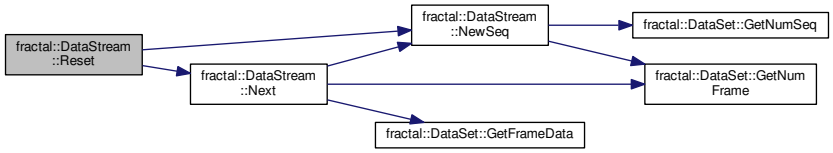


7.6.3.9 void fractal::DataStream::Reset ( ) [virtual]

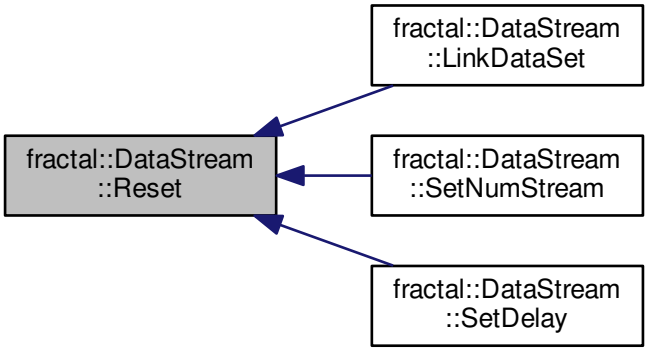
Implements fractal::Stream.

Definition at line 138 of file DataStream.cc.

Here is the call graph for this function:



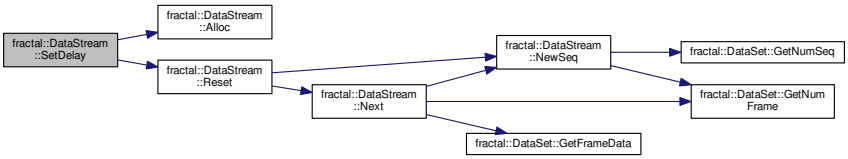
Here is the caller graph for this function:



7.6.3.10 void fractal::DataStream::SetDelay (const unsigned long *channelIdx*, const unsigned long *delay*)

Definition at line 230 of file DataStream.cc.

Here is the call graph for this function:

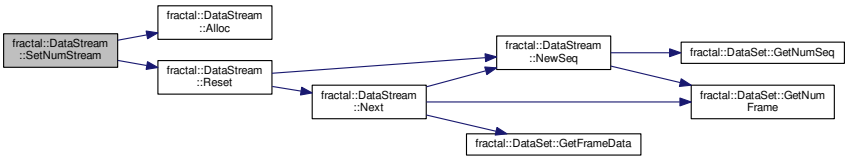


7.6.3.11 void fractal::DataStream::SetNumStream (const unsigned long *nStream*) [virtual]

Implements fractal::Stream.

Definition at line 107 of file DataStream.cc.

Here is the call graph for this function:



7.6.3.12 void fractal::DataStream::SetRandomSeed (unsigned long long *seed*)

Definition at line 258 of file DataStream.cc.

7.6.3.13 void fractal::DataStream::UnlinkDataSet ( )

Definition at line 66 of file DataStream.cc.

Here is the call graph for this function:

7.6.4 Member Data Documentation

7.6.4.1 std::vector<std::vector<std::vector<FLOAT> > > fractal::DataStream::buf [protected]

Definition at line 67 of file DataStream.h.

7.6.4.2 std::vector<std::vector<unsigned long> > fractal::DataStream::bufIdx [protected]

Definition at line 66 of file DataStream.h.

7.6.4.3 DataSet* fractal::DataStream::dataSet [protected]

Definition at line 69 of file DataStream.h.

7.6.4.4 std::vector<unsigned long> fractal::DataStream::delay [protected]

Definition at line 64 of file DataStream.h.

7.6.4.5 std::vector<unsigned long> fractal::DataStream::dim [protected]

Definition at line 63 of file DataStream.h.

7.6.4.6 std::vector<unsigned long> fractal::DataStream::frameIdx [protected]

Definition at line 65 of file DataStream.h.

7.6.4.7 unsigned long fractal::DataStream::nChannel [protected]

Definition at line 61 of file DataStream.h.

7.6.4.8 unsigned long fractal::DataStream::nStream [protected]

Definition at line 60 of file DataStream.h.

7.6.4.9 std::mt19937_64 fractal::DataStream::randGen [protected]

Definition at line 71 of file DataStream.h.

7.6.4.10 std::vector<unsigned long> fractal::DataStream::seqIdx [protected]

Definition at line 65 of file DataStream.h.

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

- src/util/DataStream.h
- src/util/DataStream.cc

7.7 fractal::Engine Class Reference

`#include <Engine.h>`

Public Member Functions

- Engine ()
- virtual ~Engine ()
- const unsigned long GetNumLoc ()
- const unsigned long GetHostLoc ()
- void MemAdd (Mem *mem)
- void MemDel (Mem *mem)
- void MemAlloc (Mem *mem, unsigned long loc)
- void MemDealloc (Mem *mem)
- void MemPull (Mem *mem, const unsigned long loc, PStream &stream)
- void MemCopy (const Mem *memSrc, const size_t offsetSrc, Mem *memDst, const size_t offsetDst, const size_t size, PStream &stream)
- void MemCopyFromHost (Mem *memDst, const size_t offsetDst, const void *ptrSrc, const size_t size, PStream &stream)
- void MemCopyToHost (const Mem *memSrc, const size_t offsetSrc, void *ptrDst, const size_t size, PStream &stream)

- void MatMult (Matrix< FLOAT > &A, const bool transA, Matrix< FLOAT > &B, const bool transB, Matrix< FLOAT > &C, const FLOAT alpha, const FLOAT beta, PStream &stream)
- void MatElemMult (Matrix< FLOAT > &A, Matrix< FLOAT > &B, Matrix< FLOAT > &C, PStream &stream)
- void MatAdd (Matrix< FLOAT > &A, Matrix< FLOAT > &B, const FLOAT alpha, PStream &stream)
- void MatAdd (Matrix< FLOAT > &A, Matrix< FLOAT > &B, Matrix< FLOAT > &C, PStream &stream)
- void MatSet (Matrix< FLOAT > &mat, const FLOAT val, PStream &stream)
- void MatRandN (Matrix< FLOAT > &mat, const FLOAT mean, const FLOAT stdev, PStream &stream)
- void MatCopy (Matrix< FLOAT > &A, Matrix< FLOAT > &B, PStream &stream)
- void MatTranspose (Matrix< FLOAT > &A, Matrix< FLOAT > &B, PStream &stream)
- void FuncSigmoid (Matrix< FLOAT > &X, Matrix< FLOAT > &Y, PStream &stream)
- void FuncTanh (Matrix< FLOAT > &X, Matrix< FLOAT > &Y, PStream &stream)
- void FuncSoftplus (Matrix< FLOAT > &X, Matrix< FLOAT > &Y, PStream &stream)
- void FuncRectLinear (Matrix< FLOAT > &X, Matrix< FLOAT > &Y, PStream &stream)
- void FuncSoftmax (Matrix< FLOAT > &X, Matrix< FLOAT > &Y, PStream &stream)
- void FuncBoundRange (Matrix< FLOAT > &X, Matrix< FLOAT > &Y, const FLOAT min, const FLOAT max, PStream &stream)
- void FuncSigmoidDeriv (Matrix< FLOAT > &X, Matrix< FLOAT > &Y, PStream &stream)
- void FuncTanhDeriv (Matrix< FLOAT > &X, Matrix< FLOAT > &Y, PStream &stream)
- void FuncSoftplusDeriv (Matrix< FLOAT > &X, Matrix< FLOAT > &Y, PStream &stream)
- void FuncRectLinearDeriv (Matrix< FLOAT > &X, Matrix< FLOAT > &Y, PStream &stream)
- void Rmsprop (Matrix< FLOAT > &newDerivs, Matrix< FLOAT > &derivs, Matrix< FLOAT > &meanSquares, const FLOAT decayRate, PStream &stream)
- void Adadelta (Matrix< FLOAT > &deltas, Matrix< FLOAT > &derivs, Matrix< FLOAT > &msDeriv, Matrix< FLOAT > &msDelta, const FLOAT learningRate, const FLOAT decayRate, PStream &stream)
- void EventCreate (PEvent &event, const unsigned long loc)
- void EventDestroy (PEvent &event)
- void EventRecord (PEvent &event, PStream &stream)
- void EventSynchronize (PEvent &event)
- void StreamCreate (PStream &stream, const unsigned long loc)
- void StreamDestroy (PStream &stream)
- void StreamWaitEvent (PStream &stream, PEvent &event)
- void StreamSynchronize (PStream &stream)
- void SetRandomSeed (unsigned long long seed)

Protected Member Functions

- void MemCopy (const Mem *memSrc, const size_t offsetSrc, const unsigned long locSrc, Mem *memDst, const size_t offsetDst, const unsigned long locDst, const size_t size, PStream &stream)
- void MemCopy (const void *ptrSrc, const unsigned long locSrc, void *ptrDst, const unsigned long locDst, const size_t size, PStream &stream)

Protected Attributes

- unsigned long numLoc
- unsigned long hostLoc
- unsigned long memCount
- unsigned long memAllocCount
- unsigned long eventCount
- unsigned long streamCount
- std::recursive_mutex mtxMem
- std::mutex mtxStream
- std::mutex mtxEvent
- cublasHandle_t cublasHandle
- curandGenerator_t curandGen

7.7.1 Detailed Description

Definition at line 80 of file Engine.h.

7.7.2 Constructor & Destructor Documentation

7.7.2.1 fractal::Engine::Engine ( )

Definition at line 61 of file Engine.cc.

7.7.2.2 fractal::Engine::~Engine ( ) [virtual]

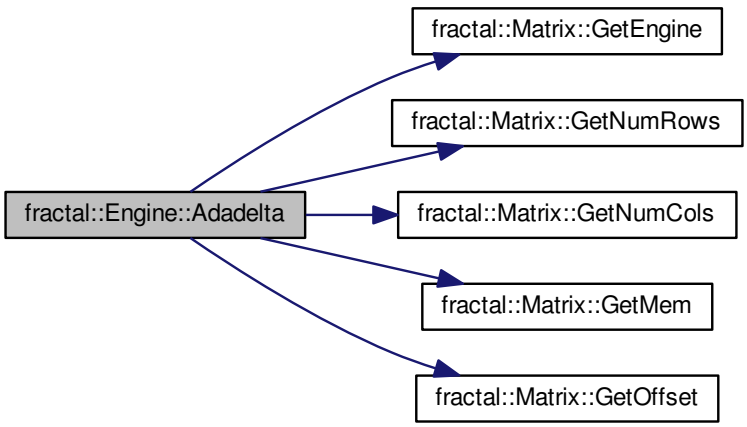
Definition at line 83 of file Engine.cc.

7.7.3 Member Function Documentation

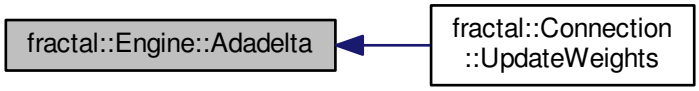
7.7.3.1 void fractal::Engine::Adadelta (Matrix< FLOAT > & *deltas*, Matrix< FLOAT > & *derivs*, Matrix< FLOAT > & *msDeriv*, Matrix< FLOAT > & *msDelta*, const FLOAT *learningRate*, const FLOAT *decayRate*, PStream & *stream*)

Definition at line 1063 of file Engine.cc.

Here is the call graph for this function:



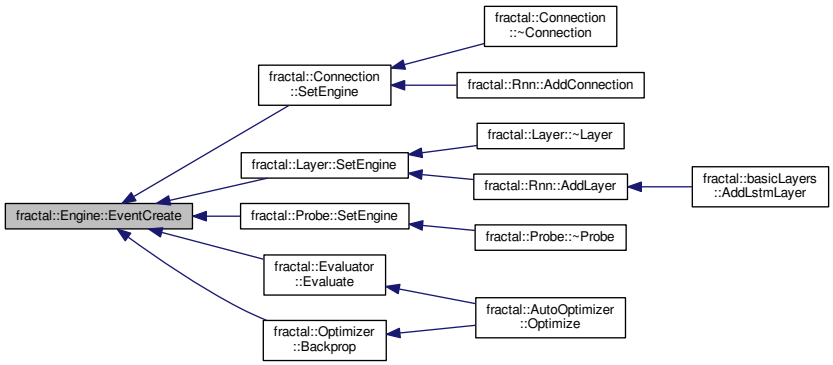
Here is the caller graph for this function:



7.7.3.2 void fractal::Engine::EventCreate (PEvent & *event,* const unsigned long *loc*)

Definition at line 1152 of file Engine.cc.

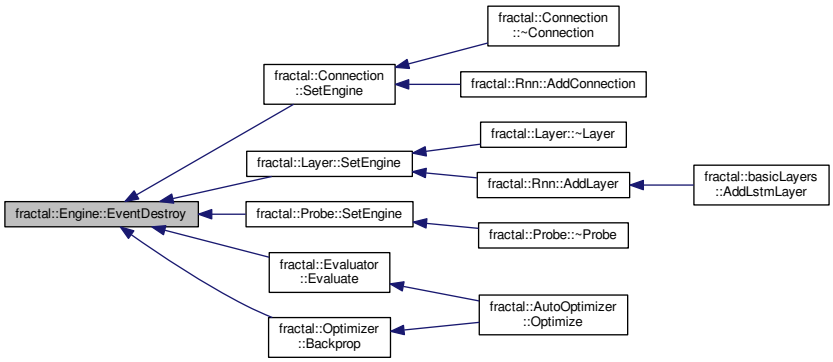
Here is the caller graph for this function:



7.7.3.3 void fractal::Engine::EventDestroy (PEvent & *event*)

Definition at line 1172 of file Engine.cc.

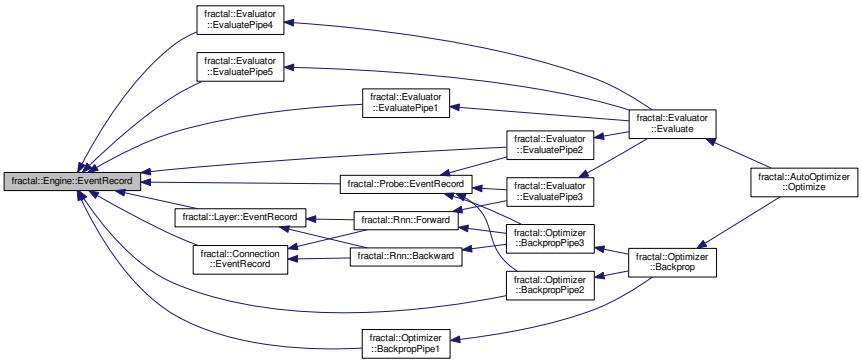
Here is the caller graph for this function:



7.7.3.4 void fractal::Engine::EventRecord (PEvent & *event*, PStream & *stream*)

Definition at line 1189 of file Engine.cc.

Here is the caller graph for this function:



7.7.3.5 void fractal::Engine::EventSynchronize (PEvent & *event*)

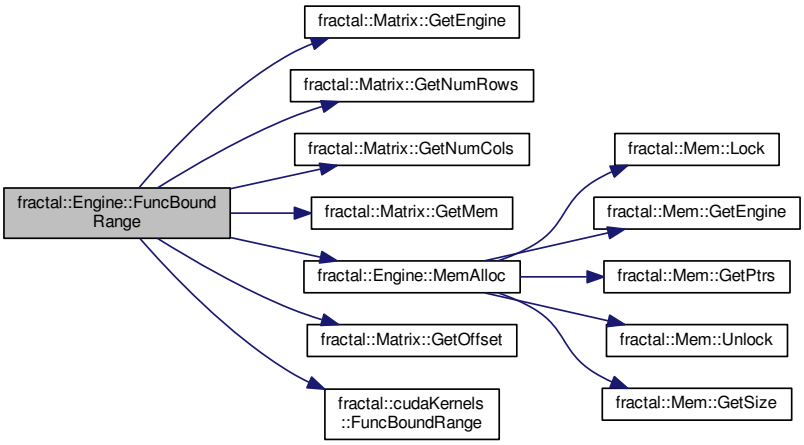
Definition at line 1210 of file Engine.cc.

Here is the caller graph for this function:

7.7.3.6 void fractal::Engine::FuncBoundRange (Matrix< FLOAT > & *X*, Matrix< FLOAT > & *Y*, const FLOAT *min*, const FLOAT *max*, PStream & *stream*)

Definition at line 882 of file Engine.cc.

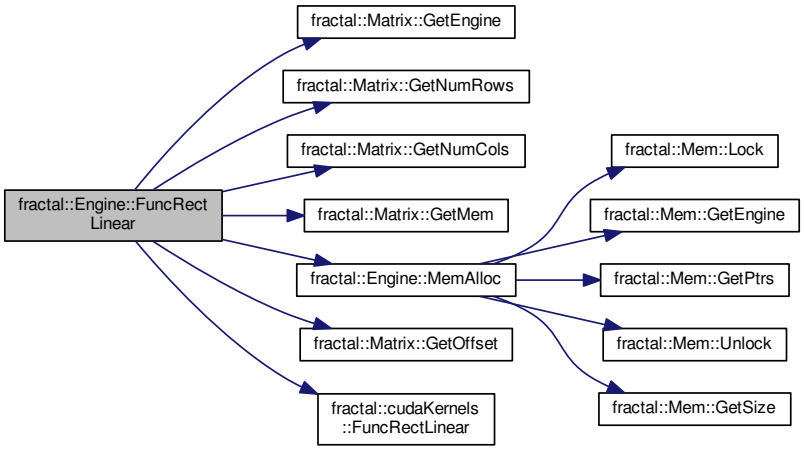
Here is the call graph for this function:



7.7.3.7 void fractal::Engine::FuncRectLinear (Matrix< FLOAT > & *X*, Matrix< FLOAT > & *Y*, PStream & *stream*)

Definition at line 810 of file Engine.cc.

Here is the call graph for this function:

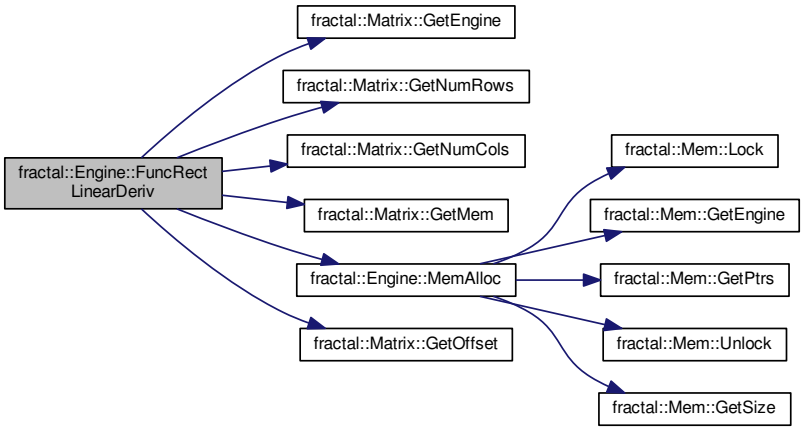


Here is the caller graph for this function:

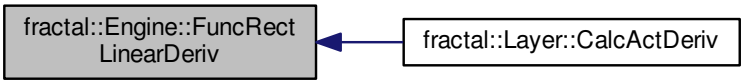
7.7.3.8 void fractal::Engine::FuncRectLinearDeriv (Matrix< FLOAT > & *X*, Matrix< FLOAT > & *Y*, PStream & *stream*)

Definition at line 1027 of file Engine.cc.

Here is the call graph for this function:



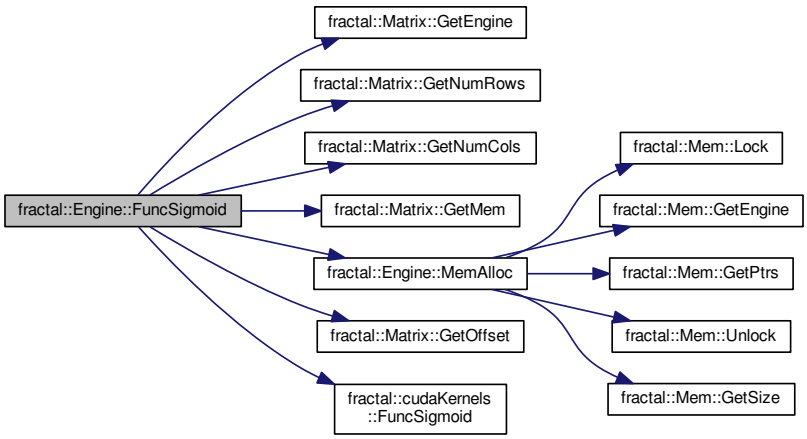
Here is the caller graph for this function:



7.7.3.9 void fractal::Engine::FuncSigmoid (Matrix< FLOAT > & *X*, Matrix< FLOAT > & *Y*, PStream & *stream*)

Definition at line 702 of file Engine.cc.

Here is the call graph for this function:

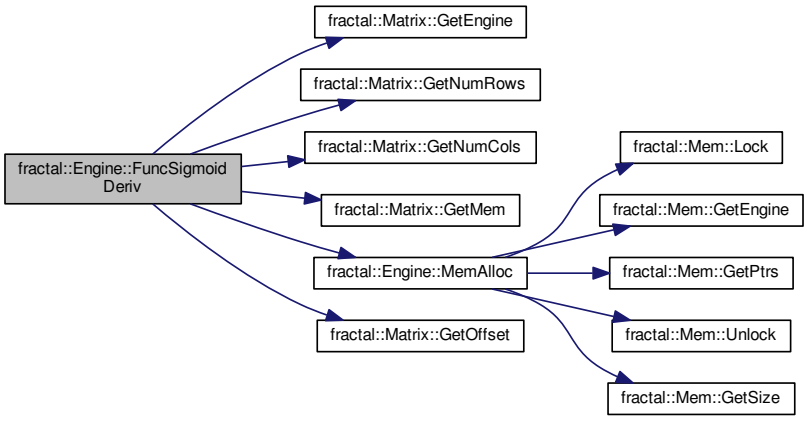


Here is the caller graph for this function:

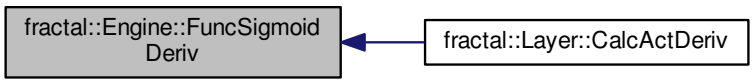
7.7.3.10 void fractal::Engine::FuncSigmoidDeriv (Matrix< FLOAT > & *X*, Matrix< FLOAT > & *Y*, PStream & *stream*)

Definition at line 919 of file Engine.cc.

Here is the call graph for this function:



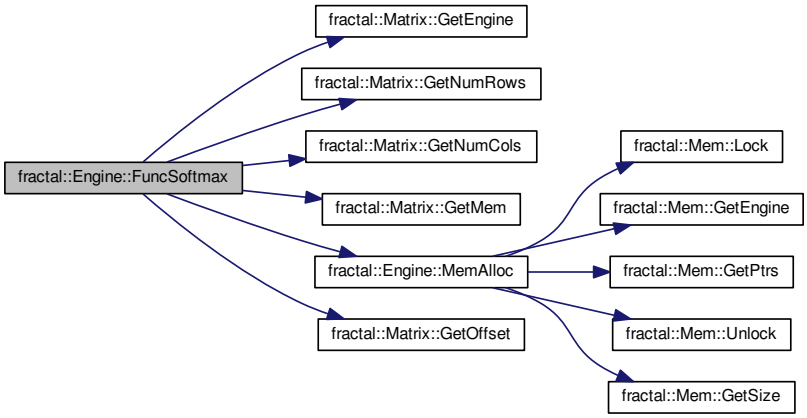
Here is the caller graph for this function:



7.7.3.11 void fractal::Engine::FuncSoftmax (Matrix< FLOAT > & *X*, Matrix< FLOAT > & *Y*, PStream & *stream*)

Definition at line 846 of file Engine.cc.

Here is the call graph for this function:

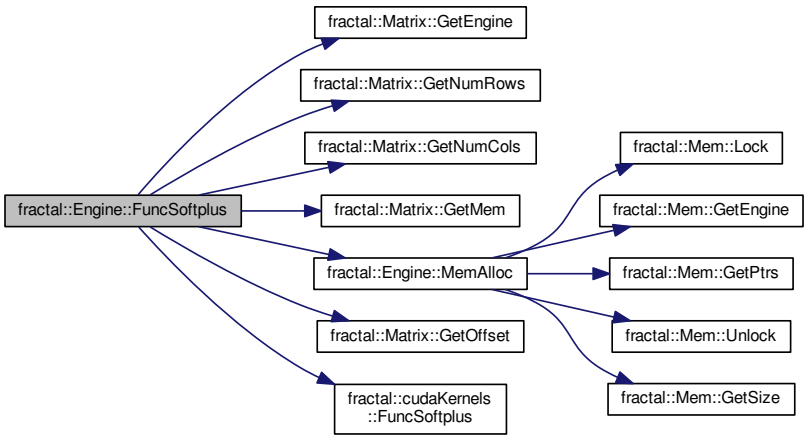


Here is the caller graph for this function:

7.7.3.12 void fractal::Engine::FuncSoftplus (Matrix< FLOAT > & *X*, Matrix< FLOAT > & *Y*, PStream & *stream*)

Definition at line 774 of file Engine.cc.

Here is the call graph for this function:

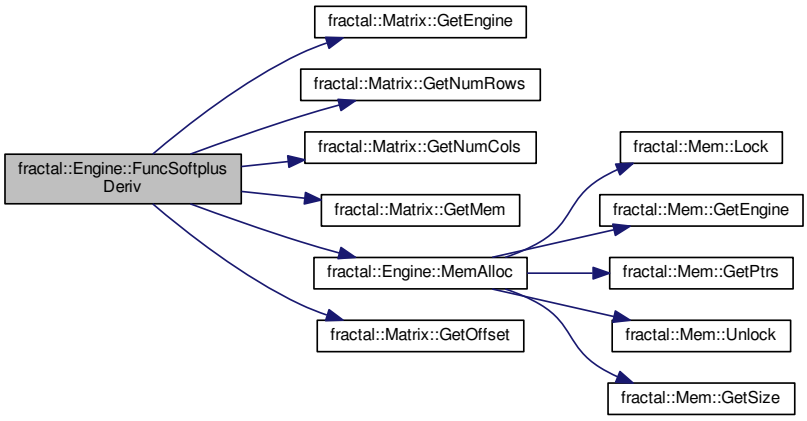


Here is the caller graph for this function:

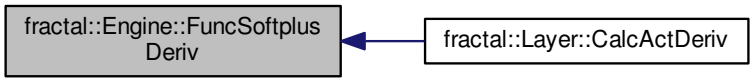
7.7.3.13 void fractal::Engine::FuncSoftplusDeriv (Matrix< FLOAT > & *X*, Matrix< FLOAT > & *Y*, PStream & *stream*)

Definition at line 991 of file Engine.cc.

Here is the call graph for this function:



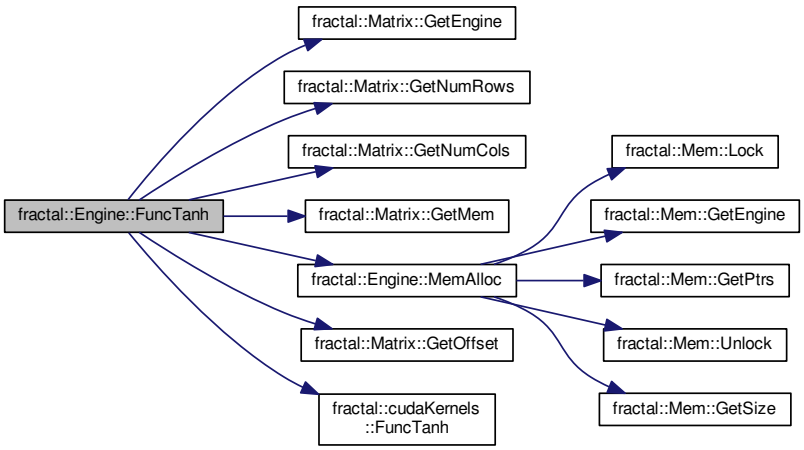
Here is the caller graph for this function:



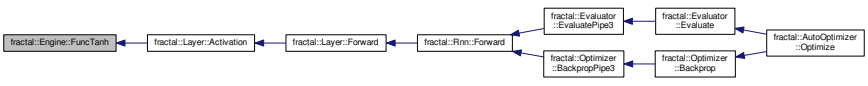
7.7.3.14 void fractal::Engine::FuncTanh (Matrix< FLOAT > & *X*, Matrix< FLOAT > & *Y*, PStream & *stream*)

Definition at line 738 of file Engine.cc.

Here is the call graph for this function:



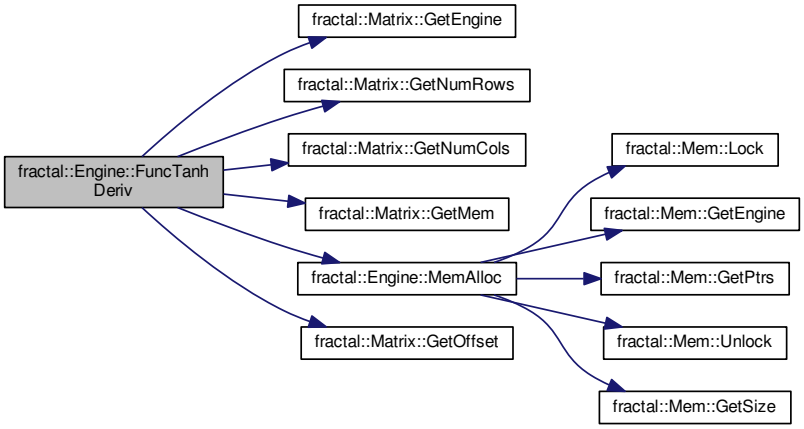
Here is the caller graph for this function:



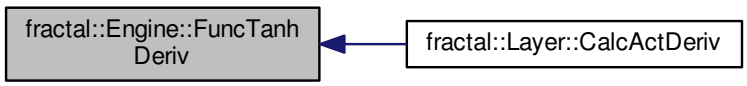
7.7.3.15 void fractal::Engine::FuncTanhDeriv (Matrix< FLOAT > & *X*, Matrix< FLOAT > & *Y*, PStream & *stream*)

Definition at line 955 of file Engine.cc.

Here is the call graph for this function:



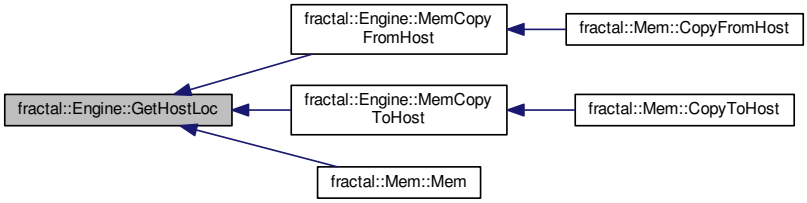
Here is the caller graph for this function:



7.7.3.16 const unsigned long fractal::Engine::GetHostLoc ( ) [inline]

Definition at line 87 of file Engine.h.

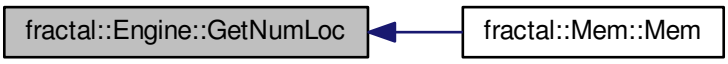
Here is the caller graph for this function:



7.7.3.17 const unsigned long fractal::Engine::GetNumLoc ( ) [inline]

Definition at line 86 of file Engine.h.

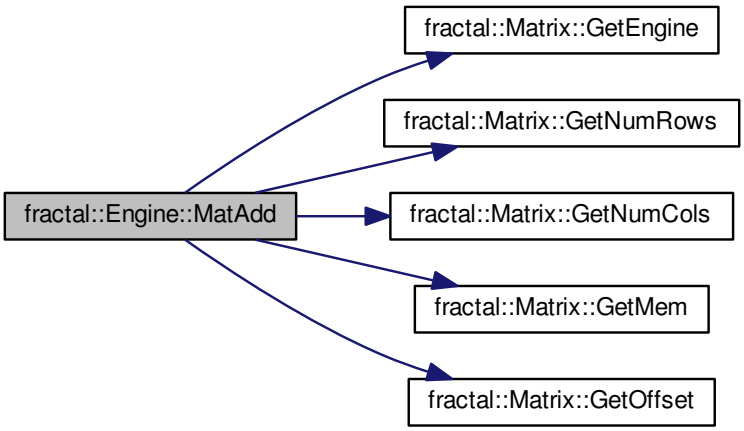
Here is the caller graph for this function:



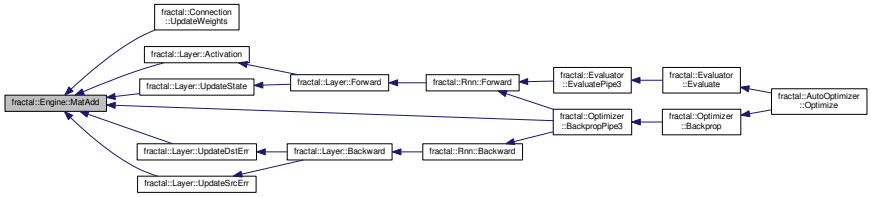
7.7.3.18 void fractal::Engine::MatAdd (Matrix< FLOAT > & *A*, Matrix< FLOAT > & *B*, const FLOAT *alpha*, PStream & *stream*)

Definition at line 455 of file Engine.cc.

Here is the call graph for this function:



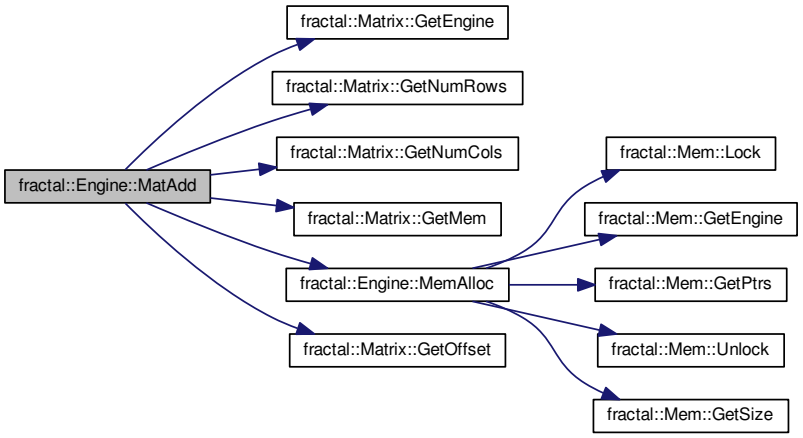
Here is the caller graph for this function:



7.7.3.19 void fractal::Engine::MatAdd (Matrix< FLOAT > & *A*, Matrix< FLOAT > & *B*, Matrix< FLOAT > & *C*, PStream & *stream*)

Definition at line 496 of file Engine.cc.

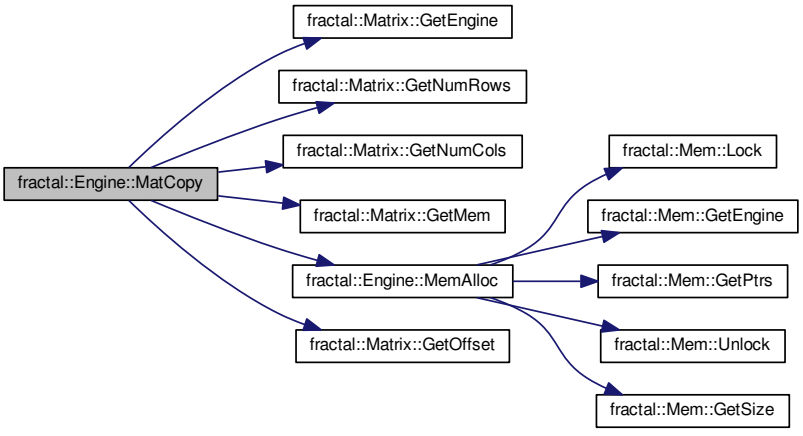
Here is the call graph for this function:



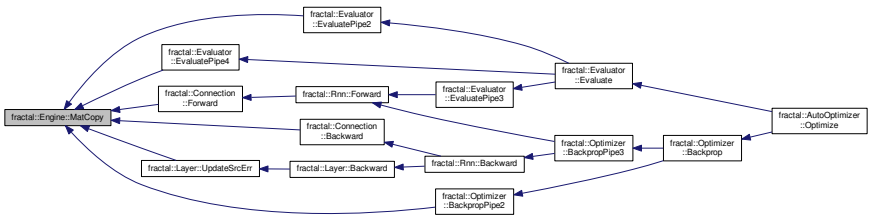
7.7.3.20 void fractal::Engine::MatCopy (Matrix< FLOAT > & *A*, Matrix< FLOAT > & *B*, PStream & *stream*)

Definition at line 604 of file Engine.cc.

Here is the call graph for this function:



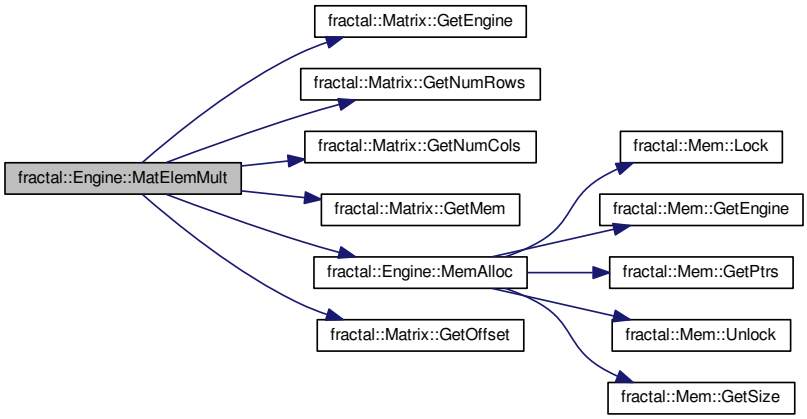
Here is the caller graph for this function:



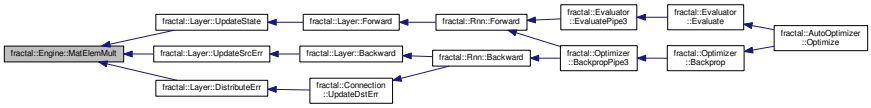
7.7.3.21 void fractal::Engine::MatElemMult (Matrix< FLOAT > & *A,* Matrix< FLOAT > & *B,* Matrix< FLOAT > & *C,* PStream & *stream*)

Definition at line 413 of file Engine.cc.

Here is the call graph for this function:



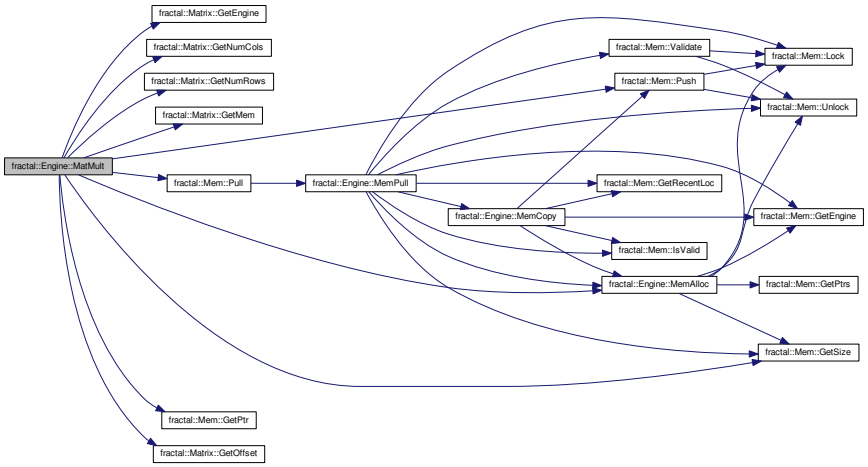
Here is the caller graph for this function:



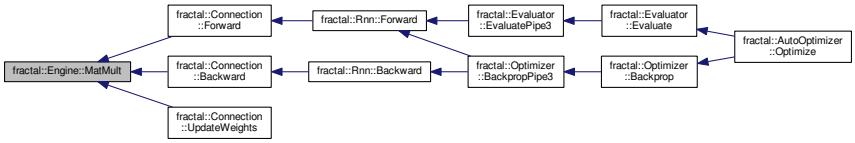
7.7.3.22 void fractal::Engine::MatMult (Matrix< FLOAT > & *A,* const bool *transA,* Matrix< FLOAT > & *B,* const bool *transB,* Matrix< FLOAT > & *C,* const FLOAT *alpha,* const FLOAT *beta,* PStream & *stream*)

Definition at line 336 of file Engine.cc.

Here is the call graph for this function:



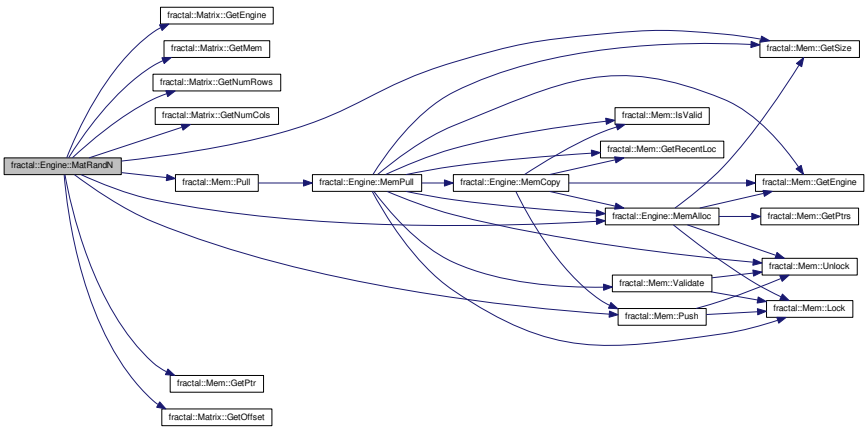
Here is the caller graph for this function:



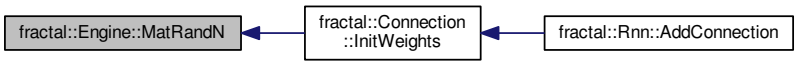
7.7.3.23 void fractal::Engine::MatRandN (Matrix< FLOAT > & *mat,* const FLOAT *mean,* const FLOAT *stdev,* PStream & *stream*)

Definition at line 567 of file Engine.cc.

Here is the call graph for this function:



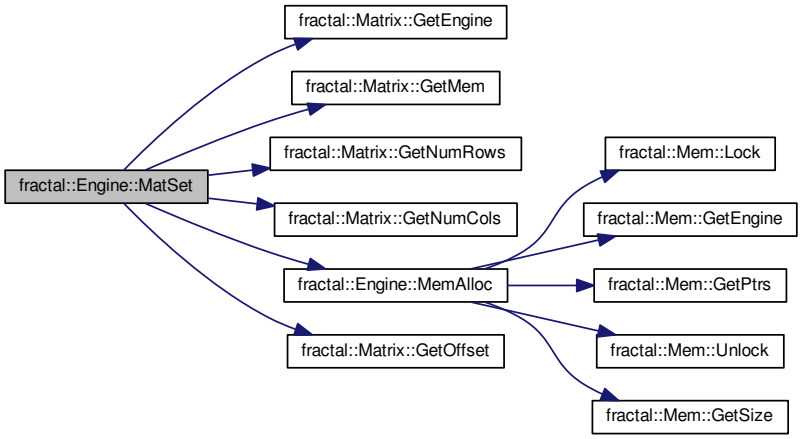
Here is the caller graph for this function:



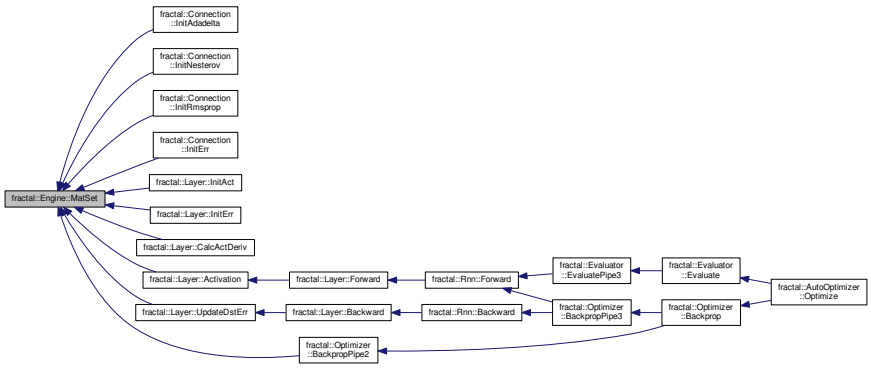
7.7.3.24 void fractal::Engine::MatSet (Matrix< FLOAT > & *mat,* const FLOAT *val,* PStream & *stream*)

Definition at line 538 of file Engine.cc.

Here is the call graph for this function:



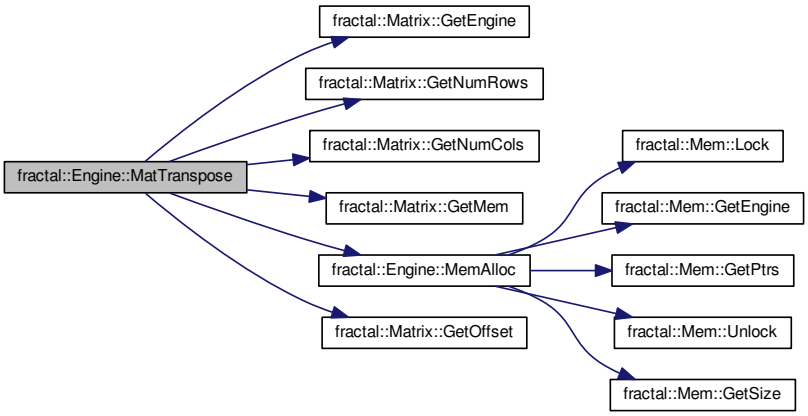
Here is the caller graph for this function:



7.7.3.25 void fractal::Engine::MatTranspose (Matrix< FLOAT > & *A*, Matrix< FLOAT > & *B*, PStream & *stream*)

Definition at line 648 of file Engine.cc.

Here is the call graph for this function:



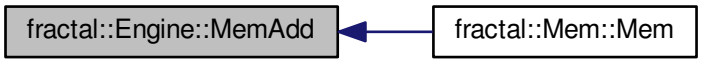
Here is the caller graph for this function:

7.7.3.26 void fractal::Engine::MemAdd (Mem * *mem*)

Definition at line 100 of file Engine.cc.

Here is the call graph for this function:

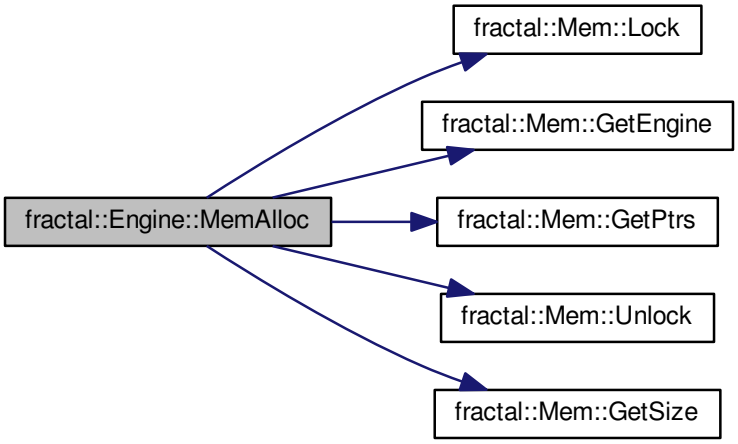
Here is the caller graph for this function:



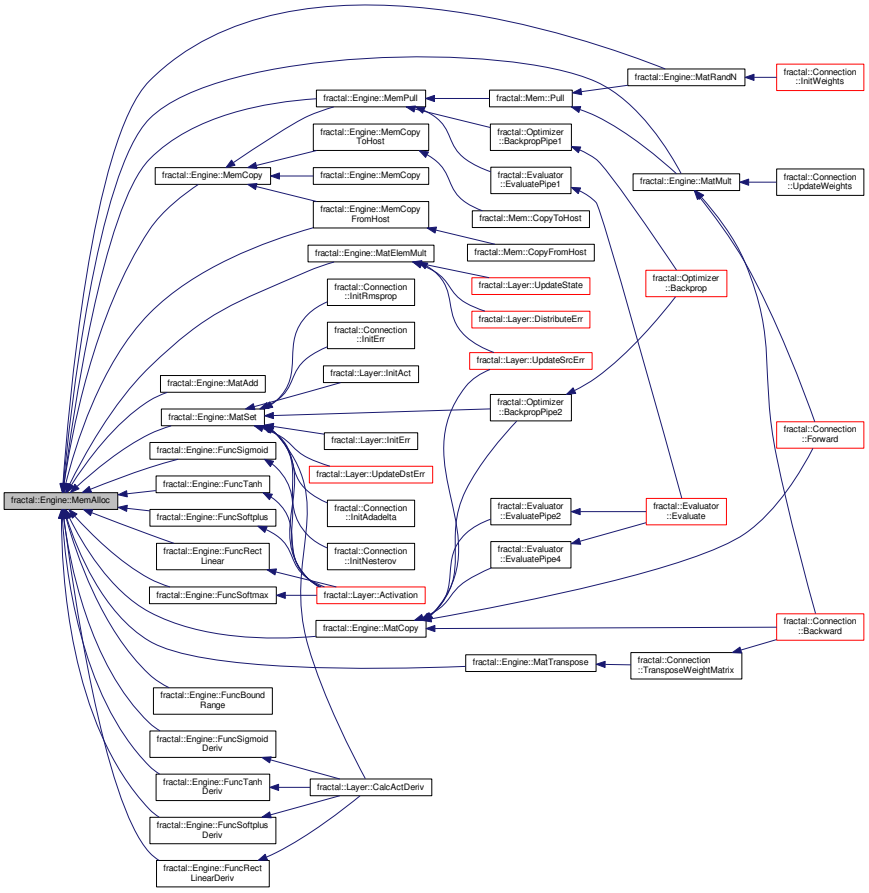
7.7.3.27 void fractal::Engine::MemAlloc (Mem * *mem,* unsigned long *loc*)

Definition at line 124 of file Engine.cc.

Here is the call graph for this function:



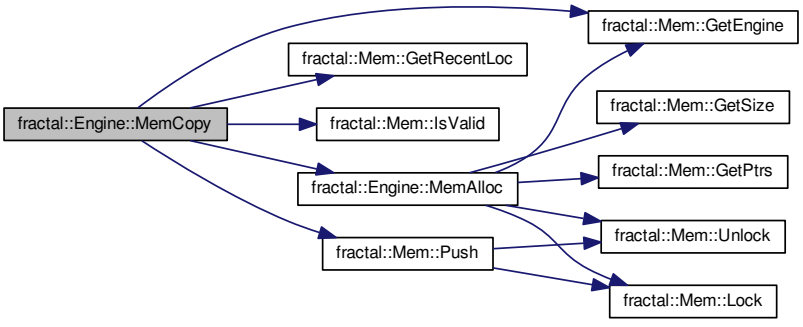
Here is the caller graph for this function:



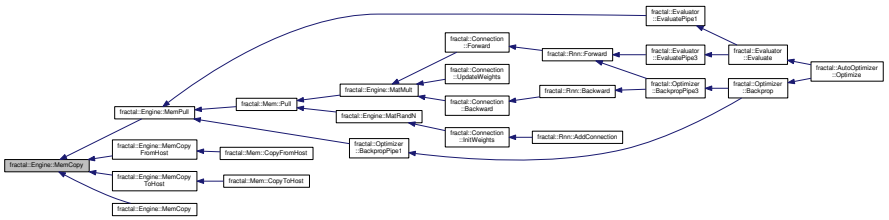
7.7.3.28 void fractal::Engine::MemCopy (const Mem * *memSrc*, const size_t *offsetSrc*, Mem * *memDst*, const size_t *offsetDst*, const size_t *size*, PStream & *stream*)

Definition at line 240 of file Engine.cc.

Here is the call graph for this function:



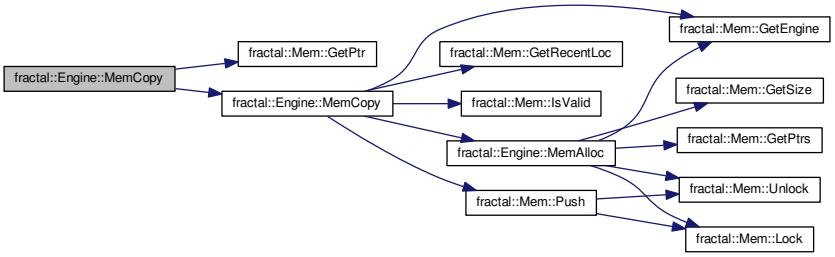
Here is the caller graph for this function:



7.7.3.29 void fractal::Engine::MemCopy (const Mem * *memSrc,* const size_t *offsetSrc,* const unsigned long *locSrc,* Mem * *memDst,* const size_t *offsetDst,* const unsigned long *locDst,* const size_t *size,* PStream & *stream*) [protected]

Definition at line 299 of file Engine.cc.

Here is the call graph for this function:



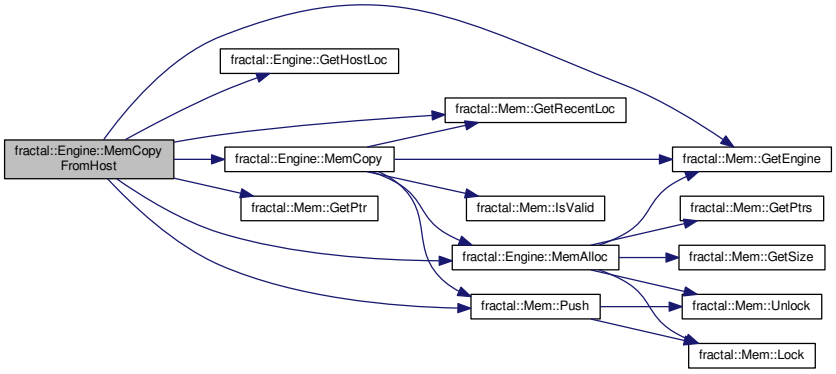
7.7.3.30 void fractal::Engine::MemCopy (const void * *ptrSrc,* const unsigned long *locSrc,* void * *ptrDst,* const unsigned long *locDst,* const size_t *size,* PStream & *stream*) [protected]

Definition at line 317 of file Engine.cc.

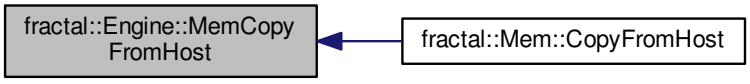
7.7.3.31 void fractal::Engine::MemCopyFromHost (Mem * *memDst,* const size_t *offsetDst,* const void * *ptrSrc,* const size_t *size,* PStream & *stream*)

Definition at line 263 of file Engine.cc.

Here is the call graph for this function:



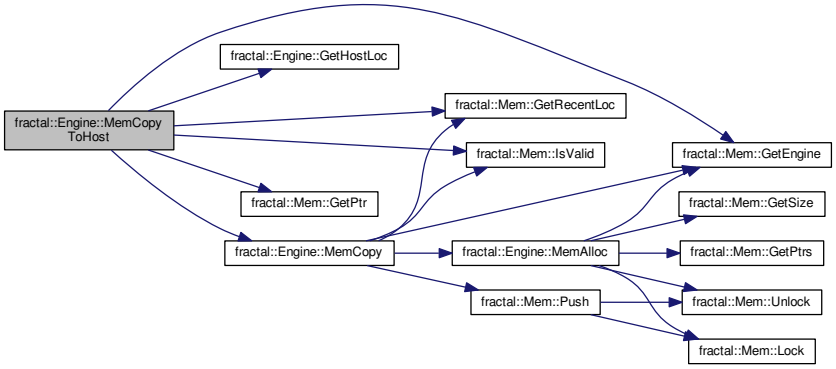
Here is the caller graph for this function:



7.7.3.32 void fractal::Engine::MemCopyToHost (const Mem * *memSrc,* const size_t *offsetSrc,* void * *ptrDst,* const size_t *size,* PStream & *stream*)

Definition at line 282 of file Engine.cc.

Here is the call graph for this function:

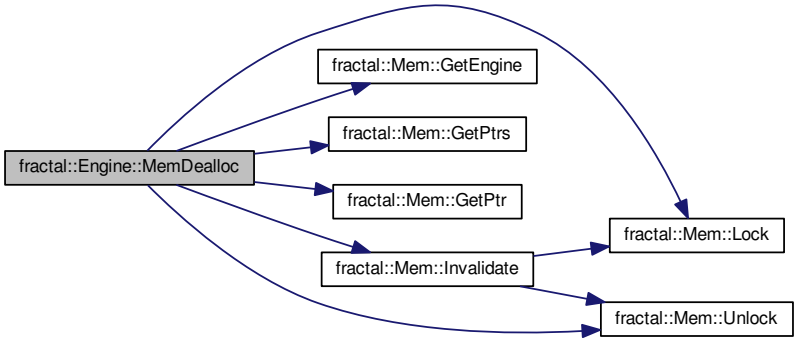


Here is the caller graph for this function:

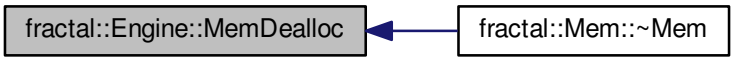
7.7.3.33 void fractal::Engine::MemDealloc (Mem * *mem*)

Definition at line 167 of file Engine.cc.

Here is the call graph for this function:



Here is the caller graph for this function:

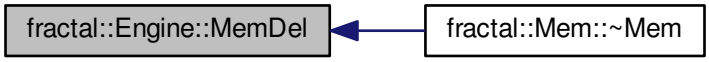


7.7.3.34 void fractal::Engine::MemDel (Mem * *mem*)

Definition at line 112 of file Engine.cc.

Here is the call graph for this function:

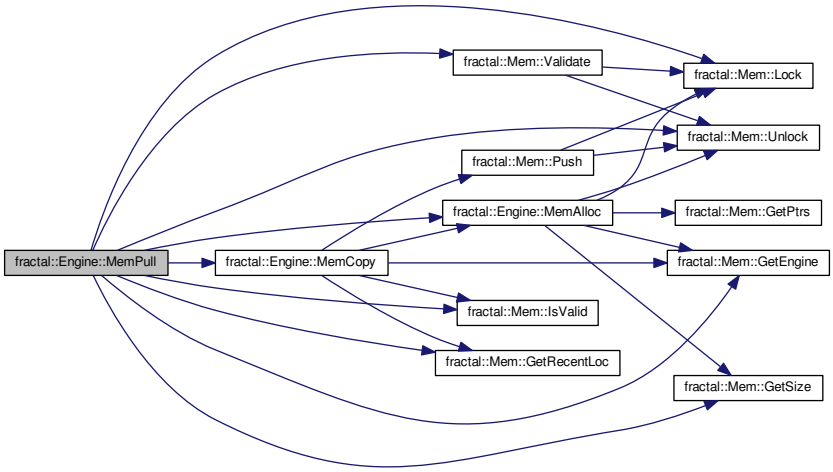
Here is the caller graph for this function:



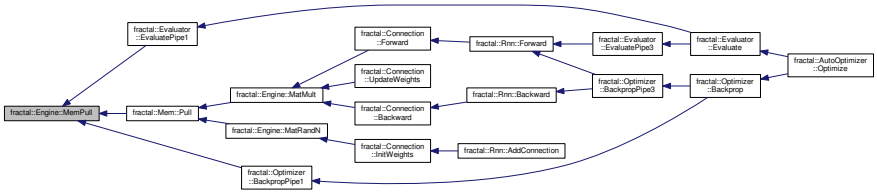
7.7.3.35 void fractal::Engine::MemPull (Mem * *mem*, const unsigned long *loc*, PStream & *stream*)

Definition at line 209 of file Engine.cc.

Here is the call graph for this function:



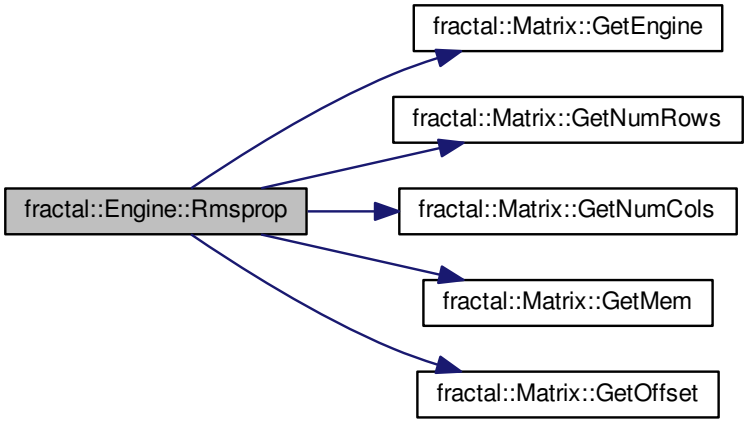
Here is the caller graph for this function:



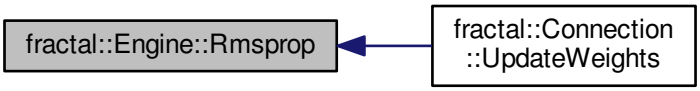
7.7.3.36 void fractal::Engine::Rmsprop (Matrix< FLOAT > & *newDerivs*, Matrix< FLOAT > & *derivs*, Matrix< FLOAT > & *meanSquares*, const FLOAT *decayRate*, PStream & *stream*)

Definition at line 1110 of file Engine.cc.

Here is the call graph for this function:



Here is the caller graph for this function:



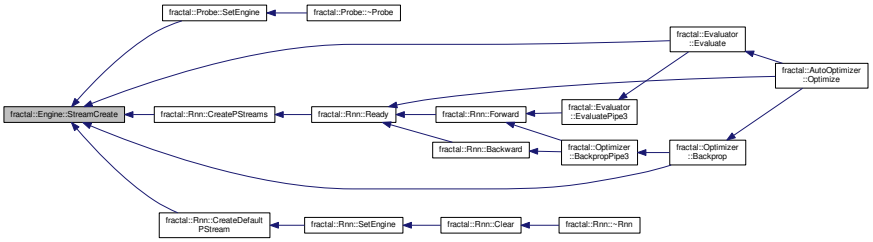
7.7.3.37 void fractal::Engine::SetRandomSeed (unsigned long long *seed*)

Definition at line 1316 of file Engine.cc.

7.7.3.38 void fractal::Engine::StreamCreate (PStream & *stream,* const unsigned long *loc*)

Definition at line 1224 of file Engine.cc.

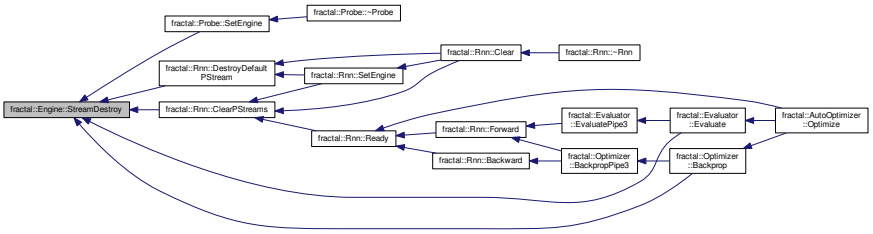
Here is the caller graph for this function:



7.7.3.39 void fractal::Engine::StreamDestroy (PStream & *stream*)

Definition at line 1246 of file Engine.cc.

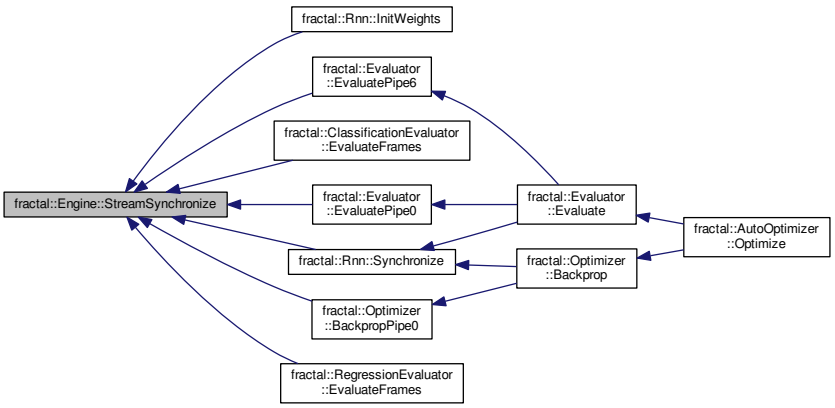
Here is the caller graph for this function:



7.7.3.40 void fractal::Engine::StreamSynchronize (PStream & *stream*)

Definition at line 1300 of file Engine.cc.

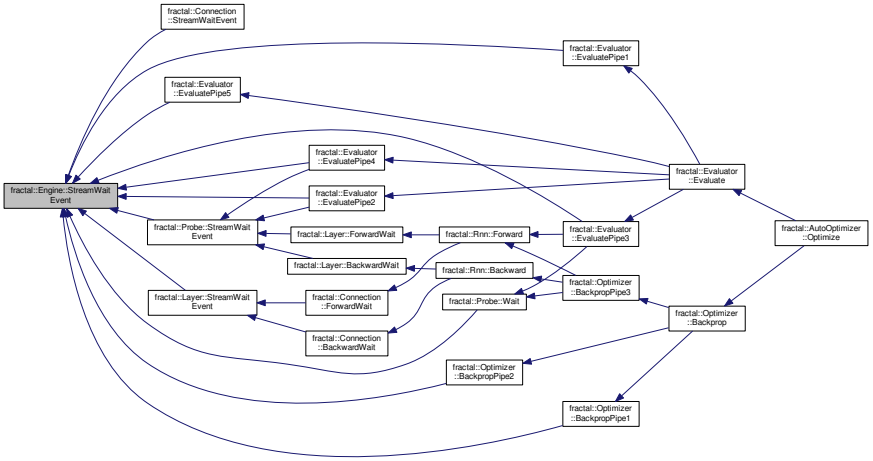
Here is the caller graph for this function:



7.7.3.41 void fractal::Engine::StreamWaitEvent (PStream & *stream,* PEvent & *event*)

Definition at line 1266 of file Engine.cc.

Here is the caller graph for this function:



7.7.4 Member Data Documentation

7.7.4.1 cublasHandle_t fractal::Engine::cublasHandle [protected]

Definition at line 175 of file Engine.h.

7.7.4.2 curandGenerator_t fractal::Engine::curandGen [protected]

Definition at line 176 of file Engine.h.

7.7.4.3 unsigned long fractal::Engine::eventCount [protected]

Definition at line 167 of file Engine.h.

7.7.4.4 unsigned long fractal::Engine::hostLoc [protected]

Definition at line 162 of file Engine.h.

7.7.4.5 unsigned long fractal::Engine::memAllocCount [protected]

Definition at line 165 of file Engine.h.

7.7.4.6 unsigned long fractal::Engine::memCount [protected]

Definition at line 164 of file Engine.h.

7.7.4.7 std::mutex fractal::Engine::mtxEvent [protected]

Definition at line 172 of file Engine.h.

7.7.4.8 std::recursive_mutex fractal::Engine::mtxMem [protected]

Definition at line 170 of file Engine.h.

7.7.4.9 std::mutex fractal::Engine::mtxStream [protected]

Definition at line 171 of file Engine.h.

7.7.4.10 unsigned long fractal::Engine::numLoc [protected]

Definition at line 161 of file Engine.h.

7.7.4.11 unsigned long fractal::Engine::streamCount [protected]

Definition at line 168 of file Engine.h.

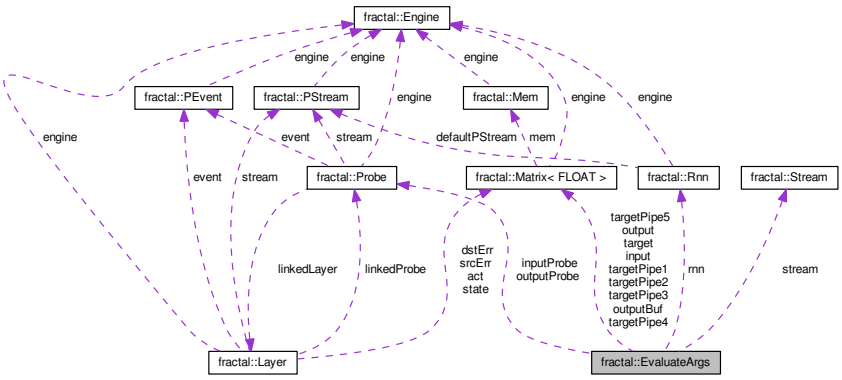
The documentation for this class was generated from the following files:

- src/core/Engine.h
- src/core/Engine.cc

7.8 fractal::EvaluateArgs Class Reference

`#include <Evaluator.h>`

Collaboration diagram for fractal::EvaluateArgs:



Public Attributes

- Rnn * rnn
- Stream * stream
- unsigned long numFrame
- unsigned long nStream
- unsigned long frameStep
- unsigned long nInput
- unsigned long nOutput
- Probe * inputProbe

- Probe * outputProbe
- unsigned long * inputChannel
- unsigned long * outputChannel
- Matrix< FLOAT > * input
- Matrix< FLOAT > * output
- Matrix< FLOAT > * outputBuf
- Matrix< FLOAT > * target
- Matrix< FLOAT > * targetPipe1
- Matrix< FLOAT > * targetPipe2
- Matrix< FLOAT > * targetPipe3
- Matrix< FLOAT > * targetPipe4
- Matrix< FLOAT > * targetPipe5

7.8.1 Detailed Description

Definition at line 31 of file Evaluator.h.

7.8.2 Member Data Documentation

7.8.2.1 unsigned long fractal::EvaluateArgs::frameStep

Definition at line 39 of file Evaluator.h.

7.8.2.2 Matrix<FLOAT>* fractal::EvaluateArgs::input

Definition at line 49 of file Evaluator.h.

7.8.2.3 unsigned long* fractal::EvaluateArgs::inputChannel

Definition at line 46 of file Evaluator.h.

7.8.2.4 Probe* fractal::EvaluateArgs::inputProbe

Definition at line 43 of file Evaluator.h.

7.8.2.5 unsigned long fractal::EvaluateArgs::nInput

Definition at line 40 of file Evaluator.h.

7.8.2.6 unsigned long fractal::EvaluateArgs::nOutput

Definition at line 41 of file Evaluator.h.

7.8.2.7 unsigned long fractal::EvaluateArgs::nStream

Definition at line 38 of file Evaluator.h.

7.8.2.8 unsigned long fractal::EvaluateArgs::numFrame

Definition at line 37 of file Evaluator.h.

7.8.2.9 Matrix<FLOAT>* fractal::EvaluateArgs::output

Definition at line 50 of file Evaluator.h.

7.8.2.10 Matrix<FLOAT> * fractal::EvaluateArgs::outputBuf

Definition at line 50 of file Evaluator.h.

7.8.2.11 unsigned long* fractal::EvaluateArgs::outputChannel

Definition at line 47 of file Evaluator.h.

7.8.2.12 Probe* fractal::EvaluateArgs::outputProbe

Definition at line 44 of file Evaluator.h.

7.8.2.13 Rnn* fractal::EvaluateArgs::rnn

Definition at line 34 of file Evaluator.h.

7.8.2.14 Stream* fractal::EvaluateArgs::stream

Definition at line 35 of file Evaluator.h.

7.8.2.15 Matrix<FLOAT>* fractal::EvaluateArgs::target

Definition at line 51 of file Evaluator.h.

7.8.2.16 Matrix<FLOAT>* fractal::EvaluateArgs::targetPipe1

Definition at line 52 of file Evaluator.h.

7.8.2.17 Matrix<FLOAT> * fractal::EvaluateArgs::targetPipe2

Definition at line 52 of file Evaluator.h.

7.8.2.18 Matrix<FLOAT> * fractal::EvaluateArgs::targetPipe3

Definition at line 52 of file Evaluator.h.

7.8.2.19 Matrix<FLOAT> * fractal::EvaluateArgs::targetPipe4

Definition at line 52 of file Evaluator.h.

7.8.2.20 Matrix<FLOAT> * fractal::EvaluateArgs::targetPipe5

Definition at line 52 of file Evaluator.h.

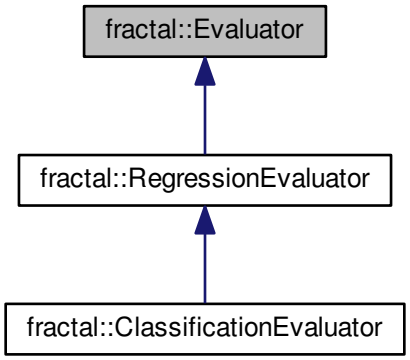
The documentation for this class was generated from the following file:

- src/util/Evaluator.h

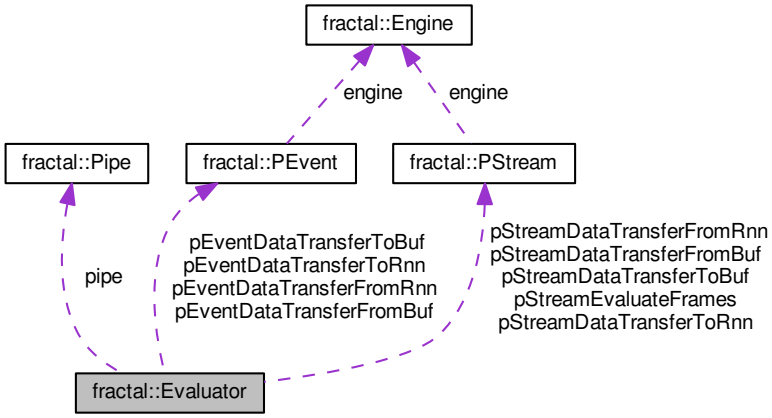
7.9 fractal::Evaluator Class Reference

`#include <Evaluator.h>`

Inheritance diagram for fractal::Evaluator:



Collaboration diagram for fractal::Evaluator:



Public Member Functions

- Evaluator ()
- void Evaluate (Rnn &rnn, Stream &stream, const PortMapList &inputPorts, const PortMapList &outputPorts, const unsigned long numFrame, const unsigned long stepSize)
- virtual const double GetLoss (const unsigned long outputIdx) const =0
- const unsigned long GetNumOutput ()

Protected Member Functions

- virtual void Reset ()=0
- virtual void EvaluateFrames (const unsigned long outputIdx, Matrix< FLOAT > &target, Matrix< FLOAT > &output, const unsigned long nStream, PStream &stream)=0
- virtual void MemAlloc ()=0
- void SetNumOutput (const unsigned long nOutput)

Static Protected Member Functions

- static void EvaluatePipe0 (Evaluator *evaluator, EvaluateArgs &args)
- static void EvaluatePipe1 (Evaluator *evaluator, EvaluateArgs &args)
- static void EvaluatePipe2 (Evaluator *evaluator, EvaluateArgs &args)
- static void EvaluatePipe3 (Evaluator *evaluator, EvaluateArgs &args)
- static void EvaluatePipe4 (Evaluator *evaluator, EvaluateArgs &args)
- static void EvaluatePipe5 (Evaluator *evaluator, EvaluateArgs &args)
- static void EvaluatePipe6 (Evaluator *evaluator, EvaluateArgs &args)

Protected Attributes

- Pipe pipe [7]
- PStream pStreamDataTransferToBuf
- PStream pStreamDataTransferToRnn
- PStream pStreamDataTransferFromRnn
- PStream pStreamDataTransferFromBuf
- PStream pStreamEvaluateFrames
- PEvent pEventDataTransferToBuf
- PEvent pEventDataTransferToRnn
- PEvent pEventDataTransferFromRnn
- PEvent pEventDataTransferFromBuf
- unsigned long nOutput

7.9.1 Detailed Description

Definition at line 56 of file Evaluator.h.

7.9.2 Constructor & Destructor Documentation

7.9.2.1 fractal::Evaluator::Evaluator ( ) [inline]

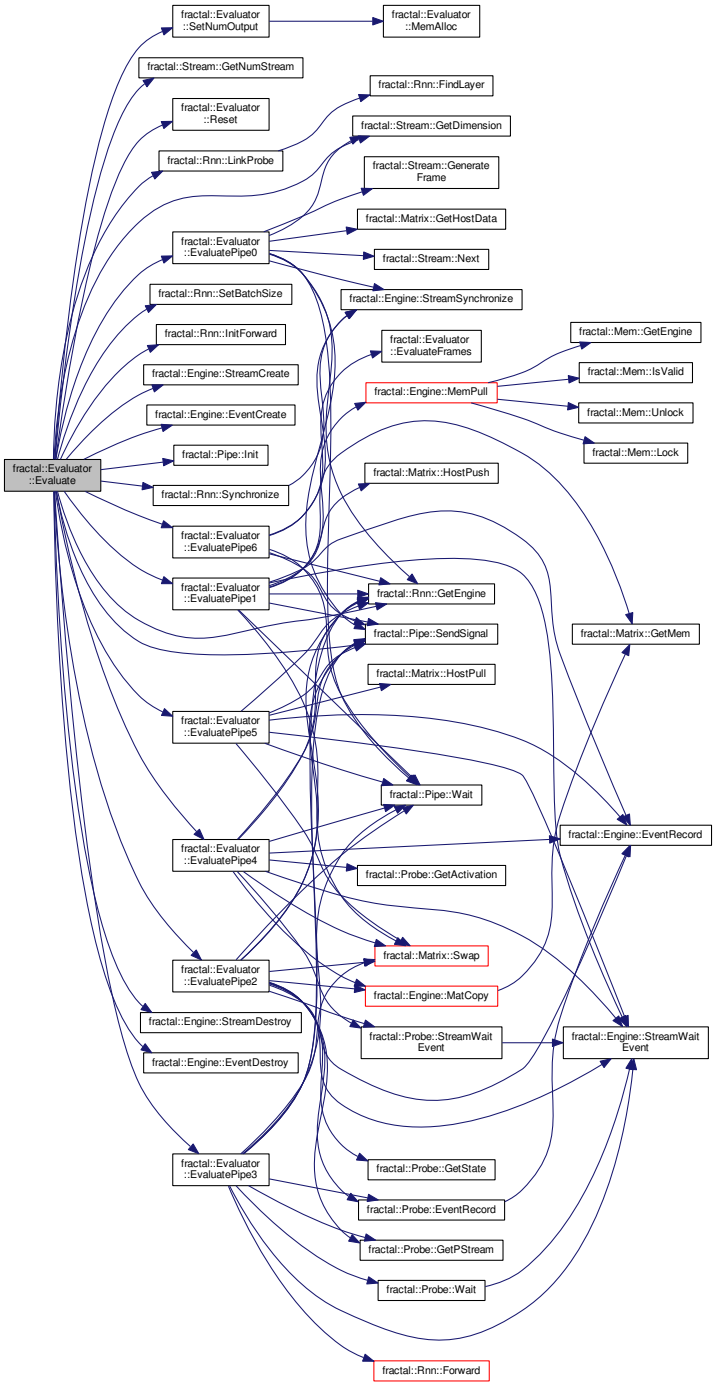
Definition at line 59 of file Evaluator.h.

7.9.3 Member Function Documentation

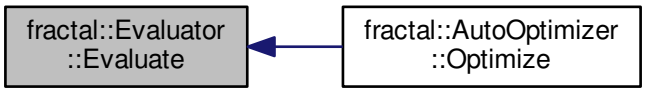
7.9.3.1 void fractal::Evaluator::Evaluate (Rnn & *rnn*, Stream & *stream*, const PortMapList & *inputPorts*, const PortMapList & *outputPorts*, const unsigned long *numFrame*, const unsigned long *stepSize*)

Definition at line 42 of file Evaluator.cc.

Here is the call graph for this function:



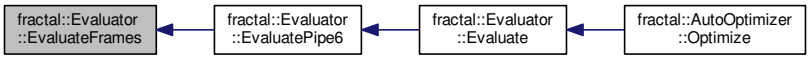
Here is the caller graph for this function:



7.9.3.2 virtual void fractal::Evaluator::EvaluateFrames (const unsigned long *outputIdx,* Matrix< FLOAT > & *target,* Matrix< FLOAT > & *output,* const unsigned long *nStream,* PStream & *stream*) [protected], [pure virtual]

Implemented in fractal::ClassificationEvaluator, and fractal::RegressionEvaluator.

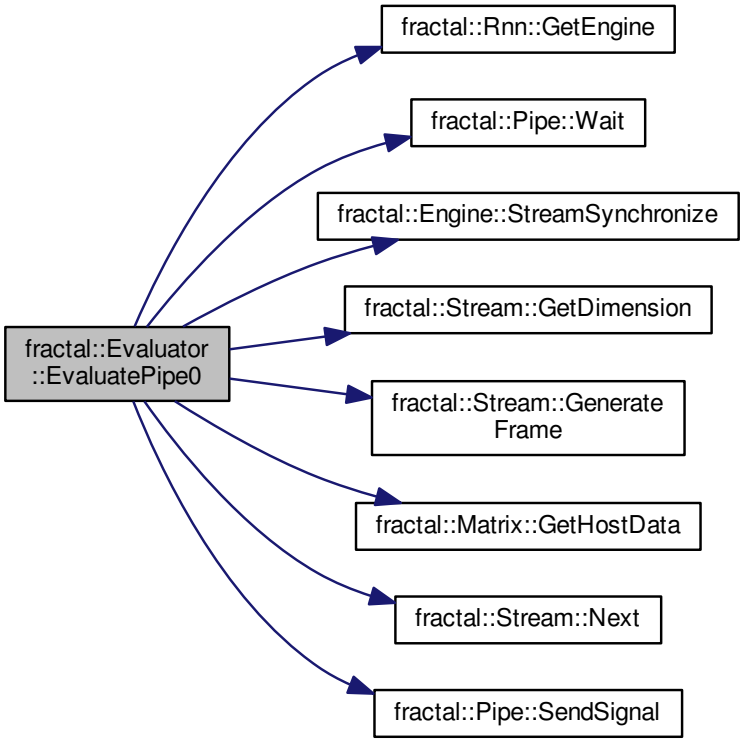
Here is the caller graph for this function:



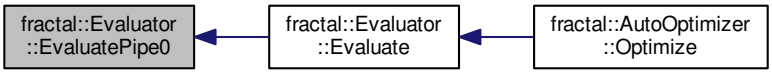
7.9.3.3 void fractal::Evaluator::EvaluatePipe0 (Evaluator * *evaluator,* EvaluateArgs & *args*) [static], [protected]

Definition at line 217 of file Evaluator.cc.

Here is the call graph for this function:



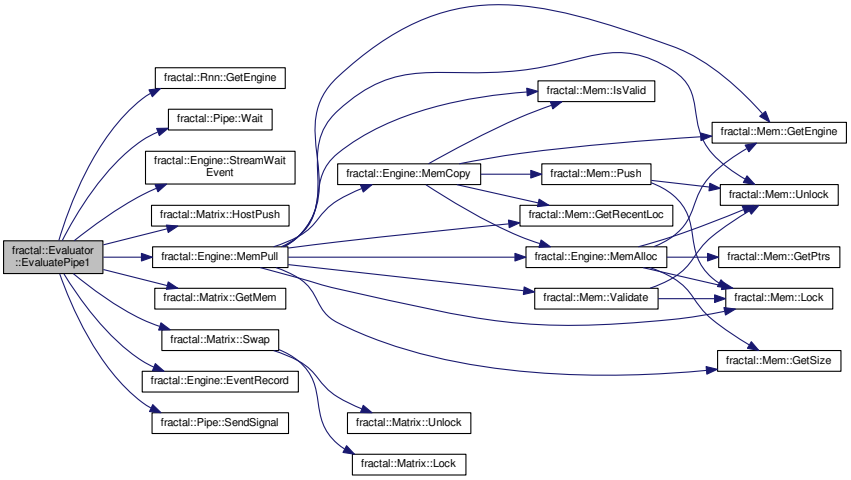
Here is the caller graph for this function:



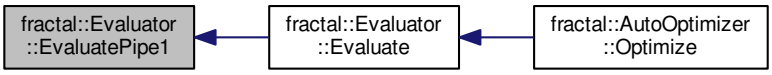
7.9.3.4 void fractal::Evaluator::EvaluatePipe1 (Evaluator * *evaluator,* EvaluateArgs & *args*) [static], [protected]

Definition at line 264 of file Evaluator.cc.

Here is the call graph for this function:



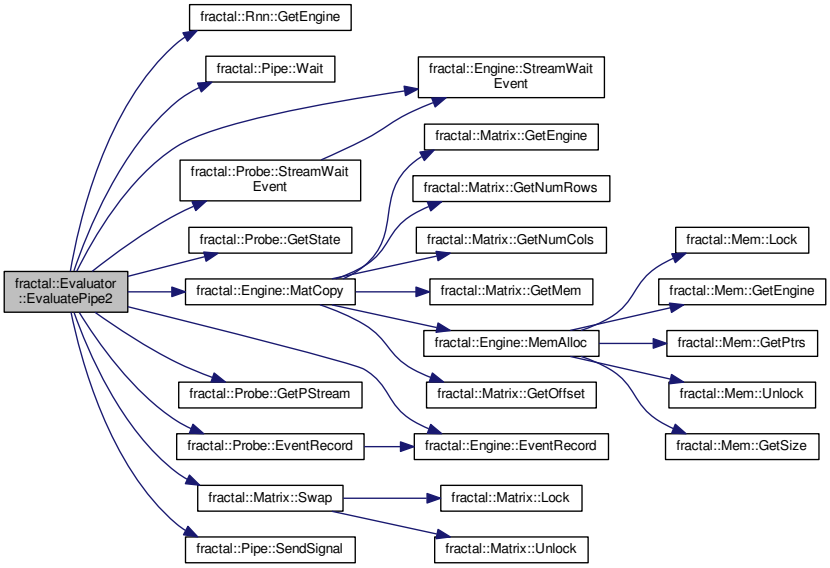
Here is the caller graph for this function:



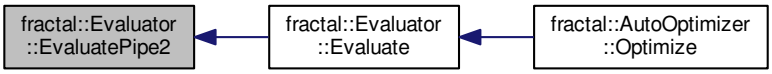
7.9.3.5 void fractal::Evaluator::EvaluatePipe2 (Evaluator * *evaluator*, EvaluateArgs & *args*) [static], [protected]

Definition at line 296 of file Evaluator.cc.

Here is the call graph for this function:



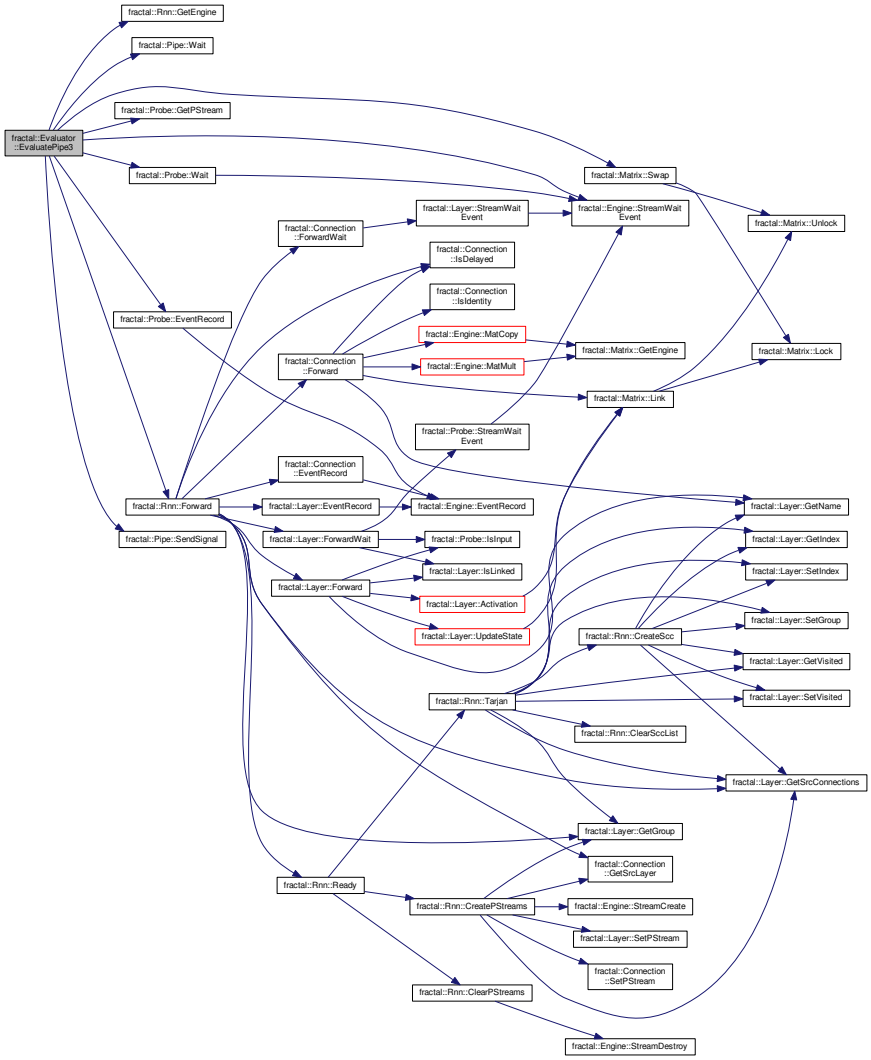
Here is the caller graph for this function:



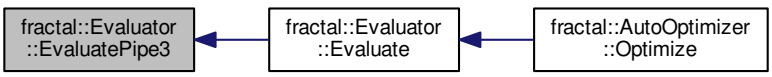
7.9.3.6 void fractal::Evaluator::EvaluatePipe3 (Evaluator * *evaluator,* EvaluateArgs & *args*) [static], [protected]

Definition at line 342 of file Evaluator.cc.

Here is the call graph for this function:



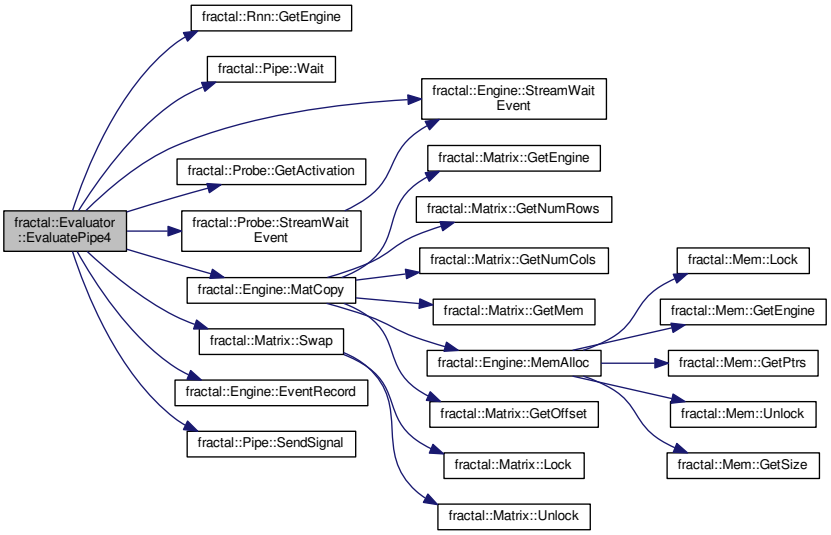
Here is the caller graph for this function:



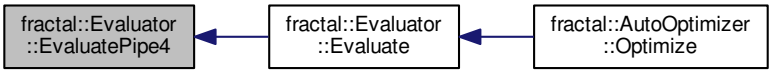
7.9.3.7 void fractal::Evaluator::EvaluatePipe4 (Evaluator * *evaluator,* EvaluateArgs & *args*) [static], [protected]

Definition at line 389 of file Evaluator.cc.

Here is the call graph for this function:



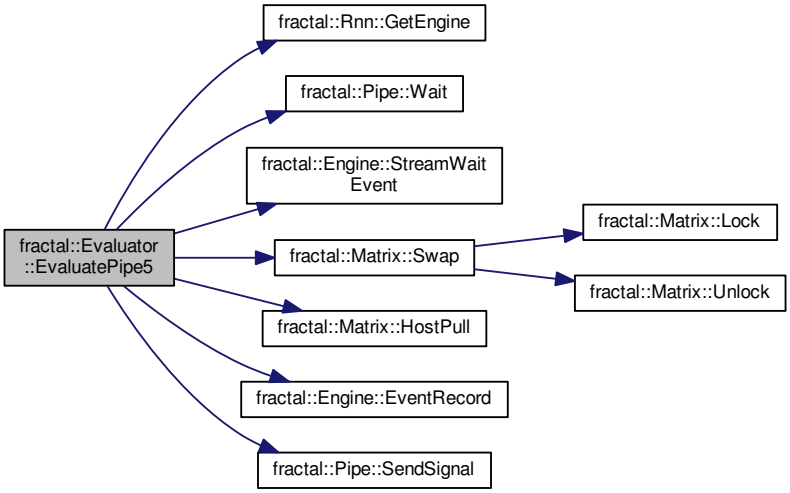
Here is the caller graph for this function:



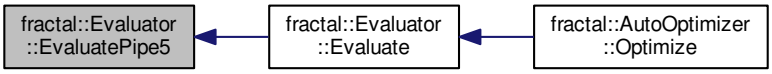
7.9.3.8 void fractal::Evaluator::EvaluatePipe5 (Evaluator * *evaluator,* EvaluateArgs & *args*) [static], [protected]

Definition at line 426 of file Evaluator.cc.

Here is the call graph for this function:



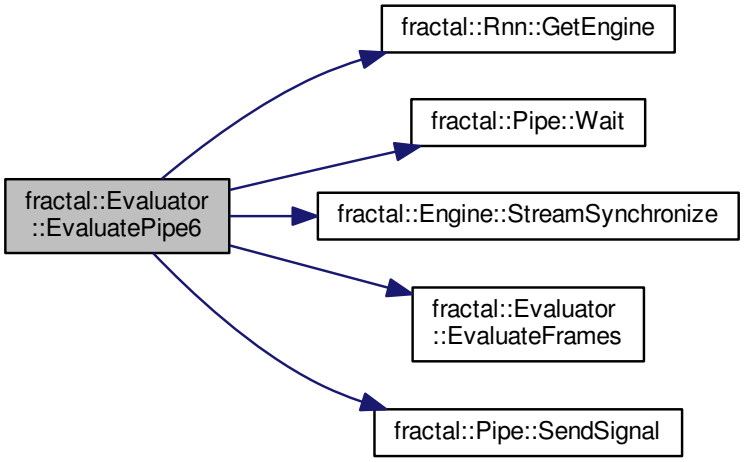
Here is the caller graph for this function:



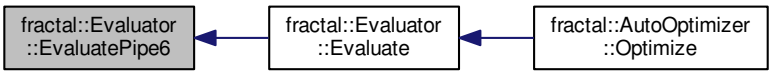
7.9.3.9 void fractal::Evaluator::EvaluatePipe6 (Evaluator * *evaluator,* EvaluateArgs & *args*) [static], [protected]

Definition at line 458 of file Evaluator.cc.

Here is the call graph for this function:



Here is the caller graph for this function:



7.9.3.10 virtual const double fractal::Evaluator::GetLoss (const unsigned long *outputIdx*) const [pure virtual]

Implemented in fractal::ClassificationEvaluator, and fractal::RegressionEvaluator.

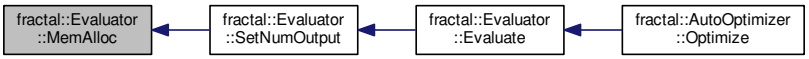
7.9.3.11 const unsigned long fractal::Evaluator::GetNumOutput ( ) [inline]

Definition at line 66 of file Evaluator.h.

7.9.3.12 virtual void fractal::Evaluator::MemAlloc ( ) [protected], [pure virtual]

Implemented in fractal::ClassificationEvaluator, and fractal::RegressionEvaluator.

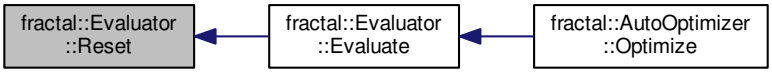
Here is the caller graph for this function:



7.9.3.13 virtual void fractal::Evaluator::Reset ( ) [protected], [pure virtual]

Implemented in fractal::ClassificationEvaluator, and fractal::RegressionEvaluator.

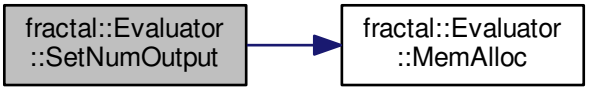
Here is the caller graph for this function:



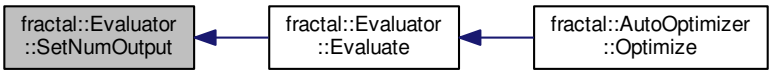
7.9.3.14 void fractal::Evaluator::SetNumOutput (const unsigned long *nOutput*) [protected]

Definition at line 33 of file Evaluator.cc.

Here is the call graph for this function:



Here is the caller graph for this function:



7.9.4 Member Data Documentation

7.9.4.1 unsigned long fractal::Evaluator::nOutput [protected]

Definition at line 98 of file Evaluator.h.

7.9.4.2 PEvent fractal::Evaluator::pEventDataTransferFromBuf [protected]

Definition at line 96 of file Evaluator.h.

7.9.4.3 PEvent fractal::Evaluator::pEventDataTransferFromRnn [protected]

Definition at line 95 of file Evaluator.h.

7.9.4.4 PEvent fractal::Evaluator::pEventDataTransferToBuf [protected]

Definition at line 93 of file Evaluator.h.

7.9.4.5 PEvent fractal::Evaluator::pEventDataTransferToRnn [protected]

Definition at line 94 of file Evaluator.h.

7.9.4.6 Pipe fractal::Evaluator::pipe[7] [protected]

Definition at line 87 of file Evaluator.h.

7.9.4.7 PStream fractal::Evaluator::pStreamDataTransferFromBuf [protected]

Definition at line 91 of file Evaluator.h.

7.9.4.8 PStream fractal::Evaluator::pStreamDataTransferFromRnn [protected]

Definition at line 90 of file Evaluator.h.

7.9.4.9 PStream fractal::Evaluator::pStreamDataTransferToBuf [protected]

Definition at line 88 of file Evaluator.h.

7.9.4.10 PStream fractal::Evaluator::pStreamDataTransferToRnn [protected]

Definition at line 89 of file Evaluator.h.

7.9.4.11 PStream fractal::Evaluator::pStreamEvaluateFrames [protected]

Definition at line 92 of file Evaluator.h.

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

- src/util/Evaluator.h
- src/util/Evaluator.cc

7.10 fractal::InitWeightParam Class Reference

`#include <InitWeightParam.h>`

Public Member Functions

- InitWeightParam ()
- InitWeightParam (FLOAT stdev)
- InitWeightParam (FLOAT mean, FLOAT stdev)
- const bool IsValid () const

Public Attributes

- FLOAT mean
- FLOAT stdev

7.10.1 Detailed Description

Definition at line 26 of file InitWeightParam.h.

7.10.2 Constructor & Destructor Documentation

7.10.2.1 fractal::InitWeightParam::InitWeightParam ( ) [inline]

Definition at line 29 of file InitWeightParam.h.

7.10.2.2 fractal::InitWeightParam::InitWeightParam (FLOAT *stdev*) [inline]

Definition at line 30 of file InitWeightParam.h.

7.10.2.3 fractal::InitWeightParam::InitWeightParam (FLOAT *mean,* FLOAT *stdev*) [inline]

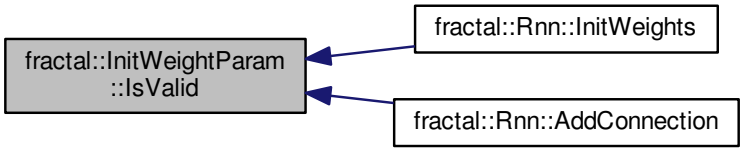
Definition at line 31 of file InitWeightParam.h.

7.10.3 Member Function Documentation

7.10.3.1 const bool fractal::InitWeightParam::IsValid ( ) const [inline]

Definition at line 33 of file InitWeightParam.h.

Here is the caller graph for this function:



7.10.4 Member Data Documentation

7.10.4.1 FLOAT fractal::InitWeightParam::mean

Definition at line 35 of file InitWeightParam.h.

7.10.4.2 FLOAT fractal::InitWeightParam::stdev

Definition at line 36 of file InitWeightParam.h.

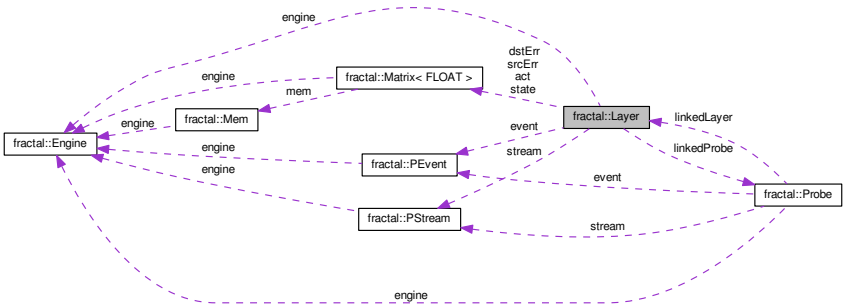
The documentation for this class was generated from the following file:

- src/core/InitWeightParam.h

7.11 fractal::Layer Class Reference

`#include <Layer.h>`

Collaboration diagram for fractal::Layer:



Public Types

- typedef std::list< Connection * > ConnList

Public Member Functions

- Layer (const std::string &name, ActType actType, StateType stateType, const unsigned long size, const LayerParam &param)
- virtual ~Layer ()
- void SetEngine (Engine *const engine, PStream *const stream)
- void AddSrcConnection (Connection *const conn)
- void AddDstConnection (Connection *const conn)
- void RemoveSrcConnection (Connection *const conn)
- void RemoveDstConnection (Connection *const conn)
- const std::string & GetName () const
- const unsigned long GetSize () const
- const unsigned long GetBatchSize () const
- void SetBatchSize (const unsigned long batchSize)
- void SetInitVal (const FLOAT val)
- void SetStatePenalty (const FLOAT val)
- void UnlinkMatrices ()
- void InitAct (const unsigned long batchFrom, const unsigned long batchTo)
- void InitErr (const unsigned long batchFrom, const unsigned long batchTo)
- void Forward (const unsigned long batchFrom, const unsigned long batchTo)
- void Backward (const unsigned long batchFrom, const unsigned long batchTo)
- void CalcActDeriv (const unsigned long batchFrom, const unsigned long batchTo)
- void LinkProbe (Probe *const probe)
- void UnlinkProbe ()
- const bool IsLinked () const
- const ConnList & GetSrcConnections () const
- const ConnList & GetDstConnections () const
- void SetVisited (const bool isVisited)
- const bool GetVisited () const
- void SetIndex (const long index)
- const long GetIndex () const
- void SetGroup (const long group)
- const long GetGroup () const
- void SetPStream (PStream *const stream)
- PStream & GetPStream ()
- void EventRecord ()
- void StreamWaitEvent (PStream &stream)
- void ForwardWait ()
- void BackwardWait ()

Protected Member Functions

- Layer (const Layer &obj)
- void Activation (const unsigned long batchFrom, const unsigned long batchTo)
- void UpdateState (const unsigned long batchFrom, const unsigned long batchTo)
- void UpdateDstErr (const unsigned long batchFrom, const unsigned long batchTo)
- void UpdateSrcErr (const unsigned long batchFrom, const unsigned long batchTo)
- void DistributeErr (Connection *conn, const unsigned long batchFrom, const unsigned long batchTo)

Protected Attributes

- Engine * engine
- ActType actType
- StateType stateType
- std::string name
- unsigned long size
- unsigned long batchSize
- ConnList srcList
- ConnList dstList
- Matrix< FLOAT > act
- Matrix< FLOAT > state
- Matrix< FLOAT > srcErr
- Matrix< FLOAT > dstErr
- Probe * linkedProbe
- FLOAT initVal
- FLOAT statePenalty
- bool isVisited
- long index
- long group
- PStream * stream
- PEvent event
- friend Probe
- friend Connection

7.11.1 Detailed Description

Definition at line 53 of file Layer.h.

7.11.2 Member Typedef Documentation

7.11.2.1 typedef std::list<Connection *> fractal::Layer::ConnList

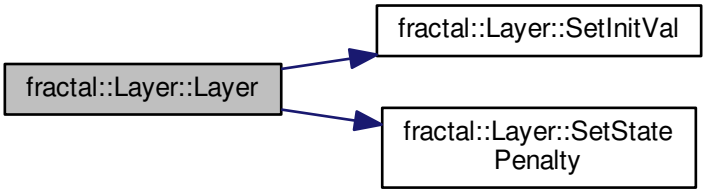
Definition at line 56 of file Layer.h.

7.11.3 Constructor & Destructor Documentation

7.11.3.1 fractal::Layer::Layer (const std::string & *name*, ActType *actType*, StateType *stateType*, const unsigned long *size*, const LayerParam & *param*)

Definition at line 29 of file Layer.cc.

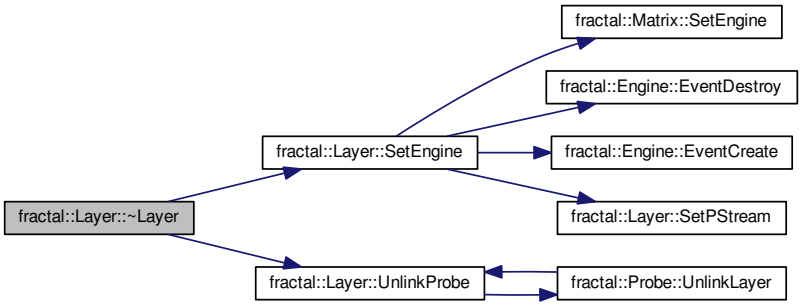
Here is the call graph for this function:



7.11.3.2 fractal::Layer::~Layer ( ) [virtual]

Definition at line 53 of file Layer.cc.

Here is the call graph for this function:



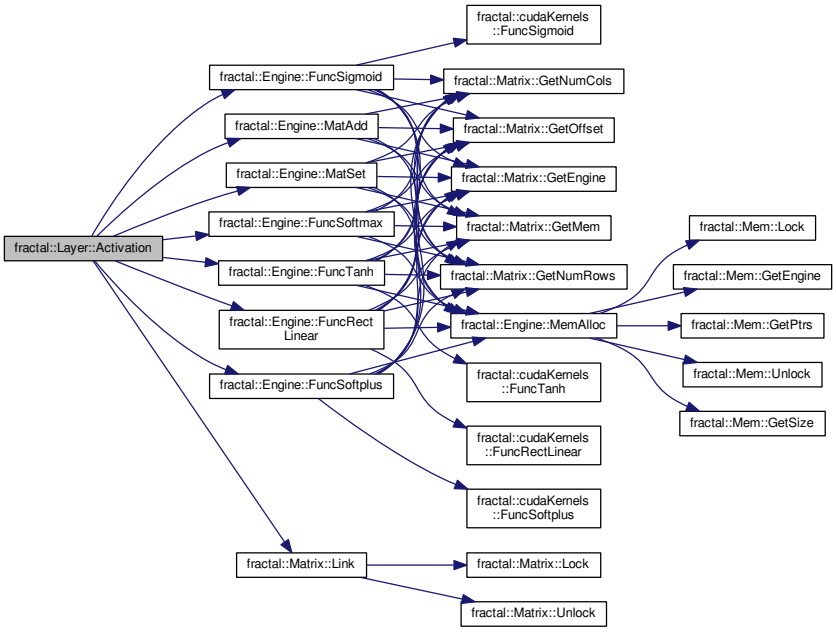
7.11.3.3 fractal::Layer::Layer (const Layer & *obj*) [protected]

7.11.4 Member Function Documentation

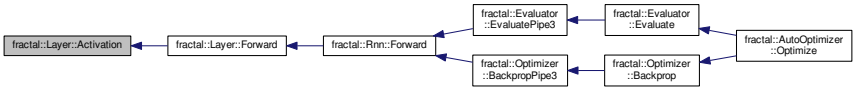
7.11.4.1 void fractal::Layer::Activation (const unsigned long *batchFrom*, const unsigned long *batchTo*) [protected]

Definition at line 302 of file Layer.cc.

Here is the call graph for this function:



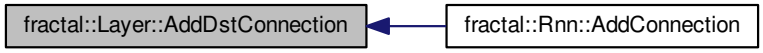
Here is the caller graph for this function:



7.11.4.2 void fractal::Layer::AddDstConnection (Connection ∗const *conn*)

Definition at line 254 of file Layer.cc.

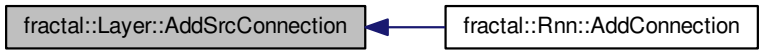
Here is the caller graph for this function:



7.11.4.3 void fractal::Layer::AddSrcConnection (Connection ∗const *conn*)

Definition at line 248 of file Layer.cc.

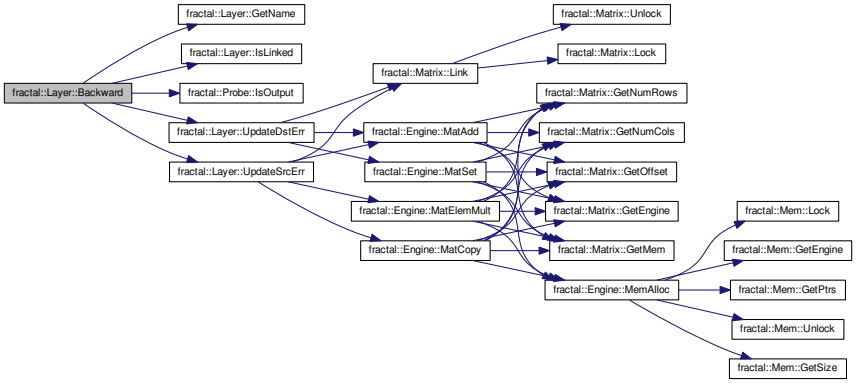
Here is the caller graph for this function:



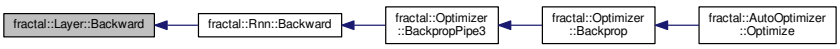
7.11.4.4 void fractal::Layer::Backward (const unsigned long *batchFrom,* const unsigned long *batchTo*)

Definition at line 172 of file Layer.cc.

Here is the call graph for this function:



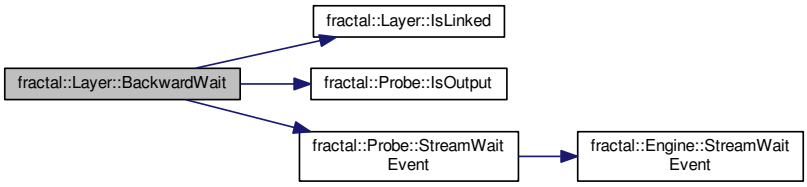
Here is the caller graph for this function:



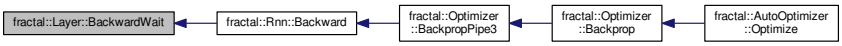
7.11.4.5 void fractal::Layer::BackwardWait ( )

Definition at line 614 of file Layer.cc.

Here is the call graph for this function:



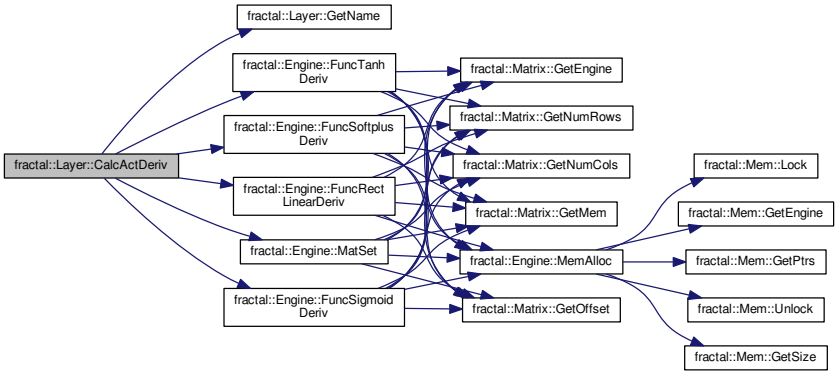
Here is the caller graph for this function:



7.11.4.6 void fractal::Layer::CalcActDeriv (const unsigned long *batchFrom,* const unsigned long *batchTo*)

Definition at line 195 of file Layer.cc.

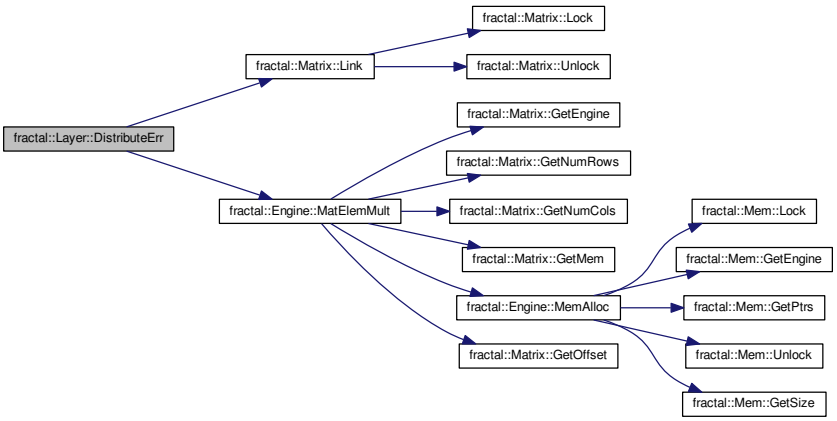
Here is the call graph for this function:



7.11.4.7 void fractal::Layer::DistributeErr (Connection * *conn,* const unsigned long *batchFrom,* const unsigned long *batchTo*) [protected]

Definition at line 518 of file Layer.cc.

Here is the call graph for this function:



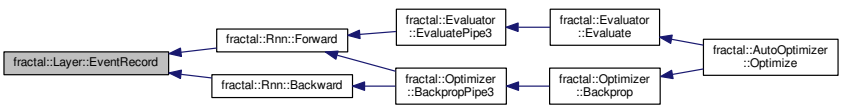
Here is the caller graph for this function:

7.11.4.8 void fractal::Layer::EventRecord ( )

Definition at line 583 of file Layer.cc.

Here is the call graph for this function:

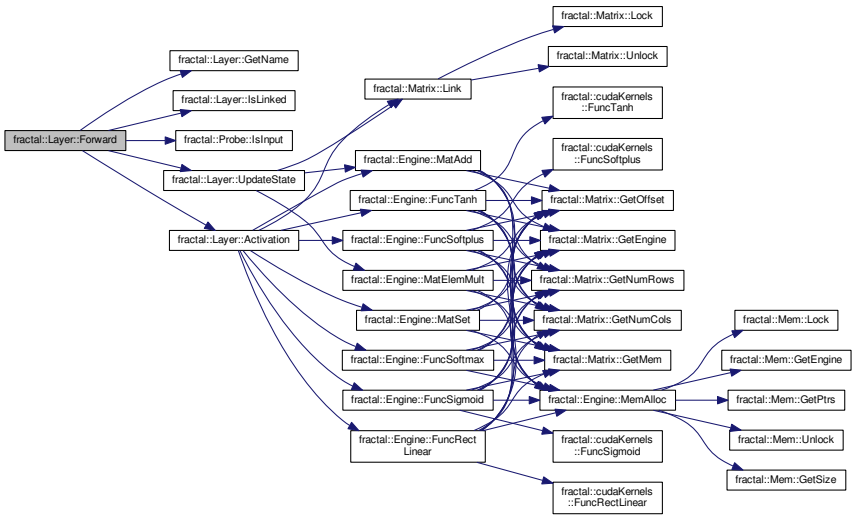
Here is the caller graph for this function:



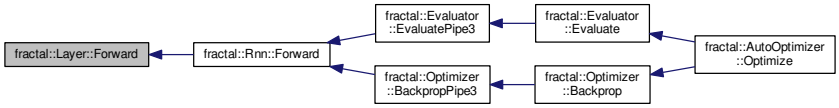
7.11.4.9 void fractal::Layer::Forward (const unsigned long *batchFrom,* const unsigned long *batchTo*)

Definition at line 148 of file Layer.cc.

Here is the call graph for this function:



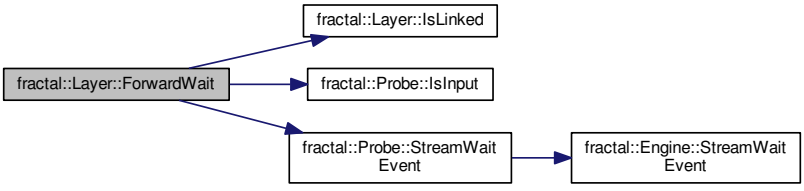
Here is the caller graph for this function:



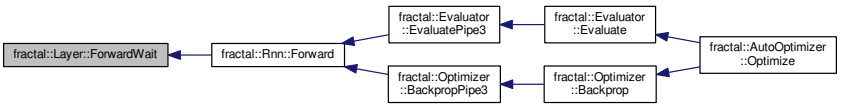
7.11.4.10 void fractal::Layer::ForwardWait ( )

Definition at line 597 of file Layer.cc.

Here is the call graph for this function:



Here is the caller graph for this function:



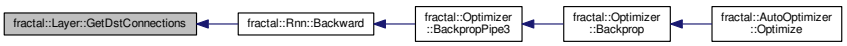
7.11.4.11 const unsigned long fractal::Layer::GetBatchSize ( ) const [inline]

Definition at line 70 of file Layer.h.

7.11.4.12 const ConnList& fractal::Layer::GetDstConnections ( ) const [inline]

Definition at line 91 of file Layer.h.

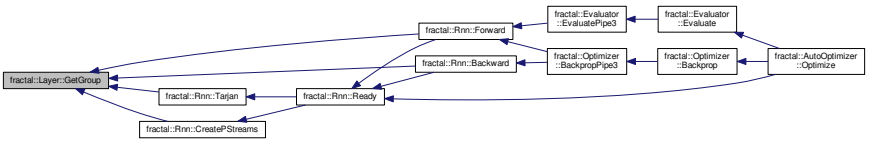
Here is the caller graph for this function:



7.11.4.13 const long fractal::Layer::GetGroup ( ) const [inline]

Definition at line 99 of file Layer.h.

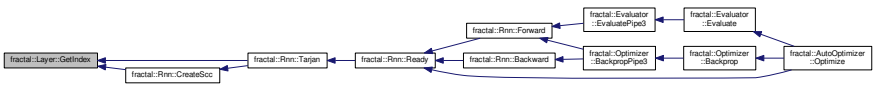
Here is the caller graph for this function:



7.11.4.14 const long fractal::Layer::GetIndex ( ) const [inline]

Definition at line 97 of file Layer.h.

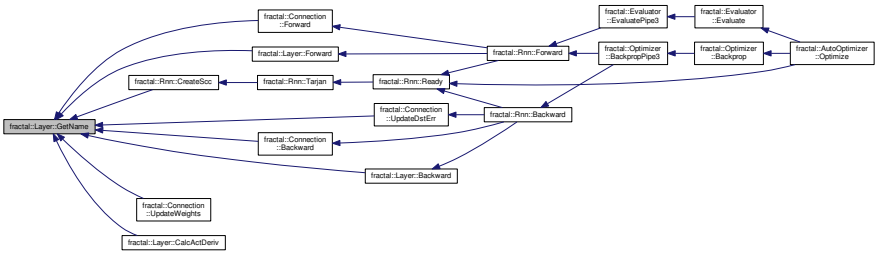
Here is the caller graph for this function:



7.11.4.15 const std::string& fractal::Layer::GetName ( ) const [inline]

Definition at line 68 of file Layer.h.

Here is the caller graph for this function:



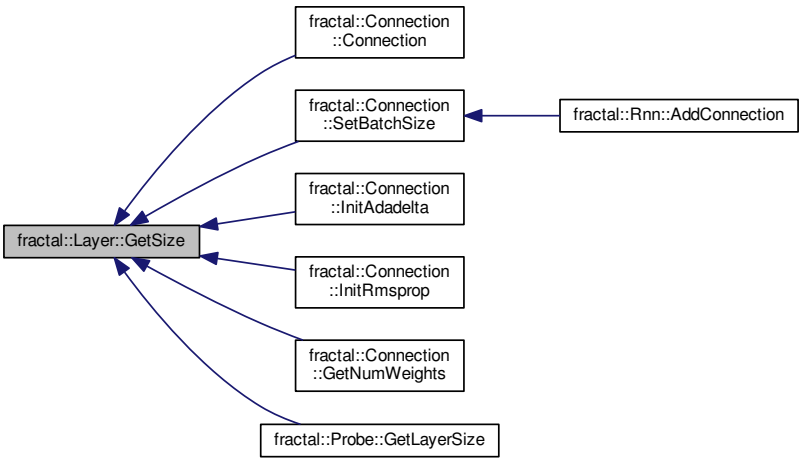
7.11.4.16 PStream & fractal::Layer::GetPStream ( )

Definition at line 576 of file Layer.cc.

7.11.4.17 const unsigned long fractal::Layer::GetSize ( ) const [inline]

Definition at line 69 of file Layer.h.

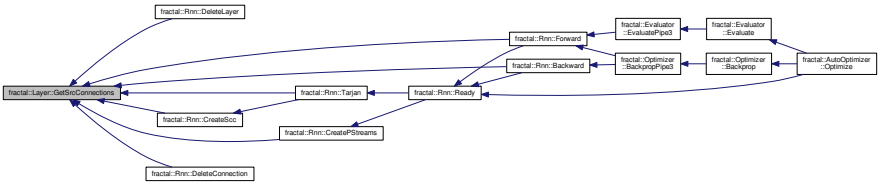
Here is the caller graph for this function:



7.11.4.18 const ConnList& fractal::Layer::GetSrcConnections ( ) const [inline]

Definition at line 90 of file Layer.h.

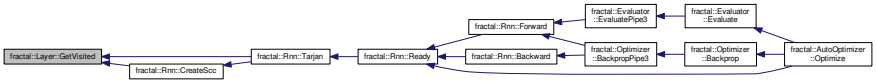
Here is the caller graph for this function:



7.11.4.19 const bool fractal::Layer::GetVisited ( ) const [inline]

Definition at line 95 of file Layer.h.

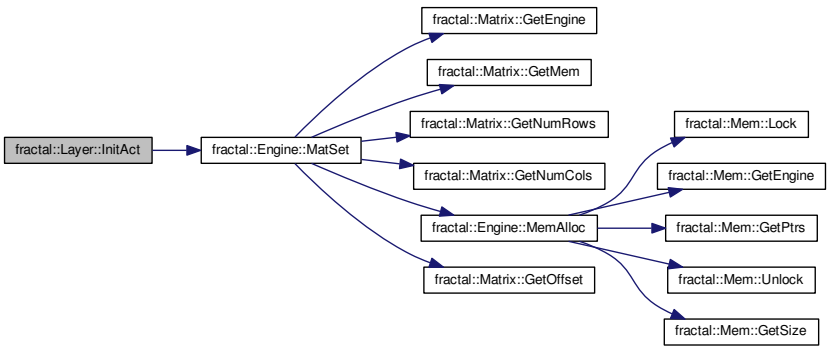
Here is the caller graph for this function:



7.11.4.20 void fractal::Layer::InitAct (const unsigned long *batchFrom,* const unsigned long *batchTo*)

Definition at line 122 of file Layer.cc.

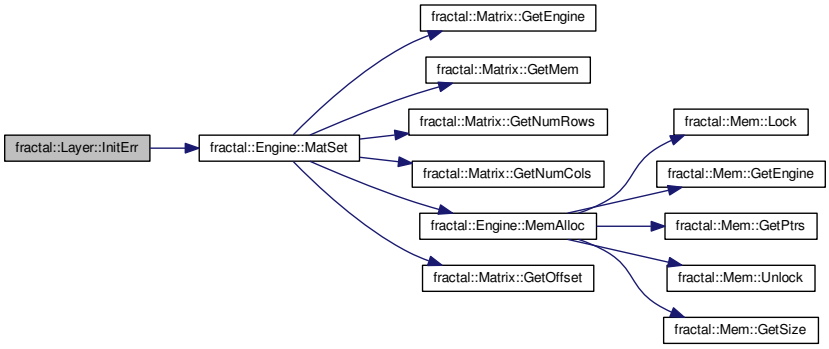
Here is the call graph for this function:



7.11.4.21 void fractal::Layer::InitErr (const unsigned long *batchFrom,* const unsigned long *batchTo*)

Definition at line 135 of file Layer.cc.

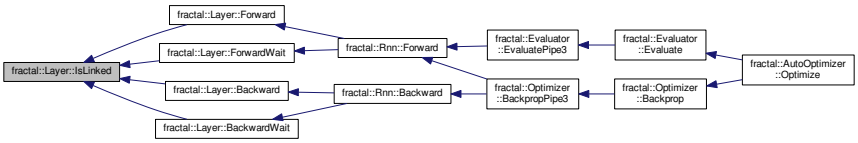
Here is the call graph for this function:



7.11.4.22 const bool fractal::Layer::IsLinked ( ) const

Definition at line 296 of file Layer.cc.

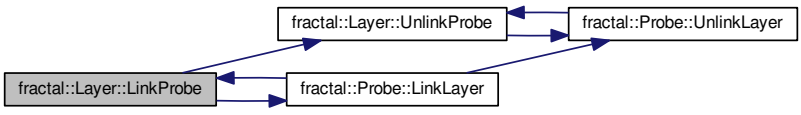
Here is the caller graph for this function:



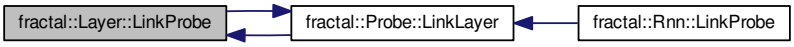
7.11.4.23 void fractal::Layer::LinkProbe (Probe ∗const *probe*)

Definition at line 272 of file Layer.cc.

Here is the call graph for this function:



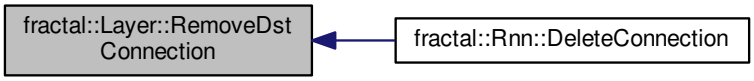
Here is the caller graph for this function:



7.11.4.24 void fractal::Layer::RemoveDstConnection (Connection ∗const *conn*)

Definition at line 266 of file Layer.cc.

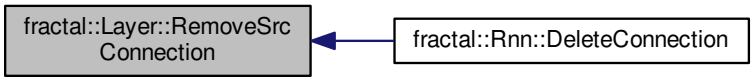
Here is the caller graph for this function:



7.11.4.25 void fractal::Layer::RemoveSrcConnection (Connection ∗const *conn*)

Definition at line 260 of file Layer.cc.

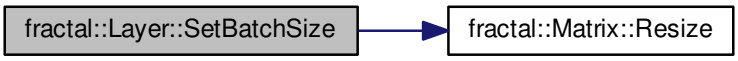
Here is the caller graph for this function:



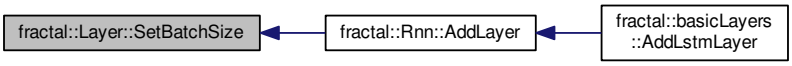
7.11.4.26 void fractal::Layer::SetBatchSize (const unsigned long *batchSize*)

Definition at line 86 of file Layer.cc.

Here is the call graph for this function:



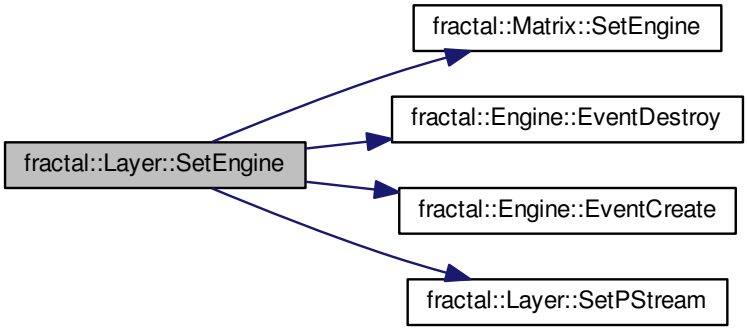
Here is the caller graph for this function:



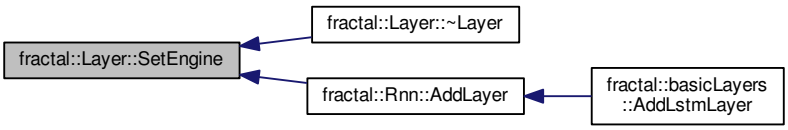
7.11.4.27 void fractal::Layer::SetEngine (Engine *const *engine*, PStream *const *stream*)

Definition at line 61 of file Layer.cc.

Here is the call graph for this function:



Here is the caller graph for this function:



7.11.4.28 void fractal::Layer::SetGroup (const long *group*) [inline]

Definition at line 98 of file Layer.h.

Here is the caller graph for this function:

7.11.4.29 void fractal::Layer::SetIndex (const long *index*) [inline]

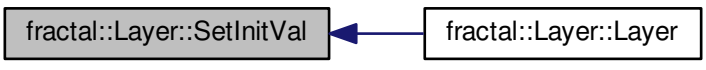
Definition at line 96 of file Layer.h.

Here is the caller graph for this function:

7.11.4.30 void fractal::Layer::SetInitVal (const FLOAT *val*)

Definition at line 116 of file Layer.cc.

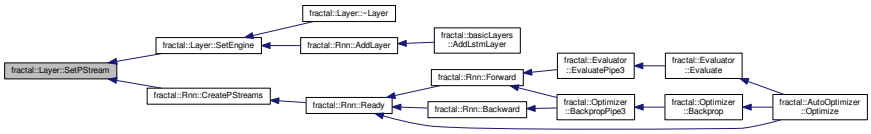
Here is the caller graph for this function:



7.11.4.31 void fractal::Layer::SetPStream (PStream ∗const *stream*)

Definition at line 569 of file Layer.cc.

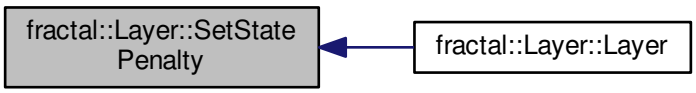
Here is the caller graph for this function:



7.11.4.32 void fractal::Layer::SetStatePenalty (const FLOAT *val*)

Definition at line 101 of file Layer.cc.

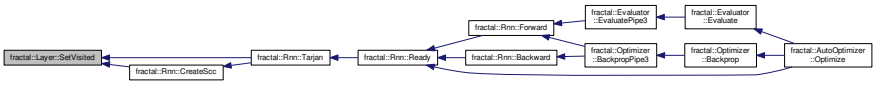
Here is the caller graph for this function:



7.11.4.33 void fractal::Layer::SetVisited (const bool *isVisited*) [inline]

Definition at line 94 of file Layer.h.

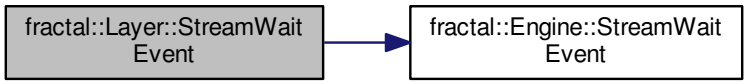
Here is the caller graph for this function:



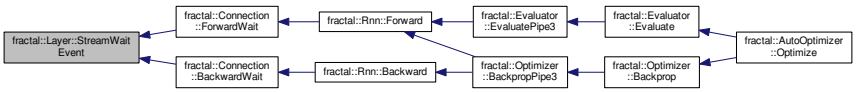
7.11.4.34 void fractal::Layer::StreamWaitEvent (PStream & *stream*)

Definition at line 590 of file Layer.cc.

Here is the call graph for this function:



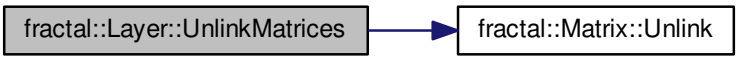
Here is the caller graph for this function:



7.11.4.35 void fractal::Layer::UnlinkMatrices ( )

Definition at line 107 of file Layer.cc.

Here is the call graph for this function:

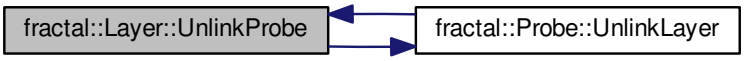


Here is the caller graph for this function:

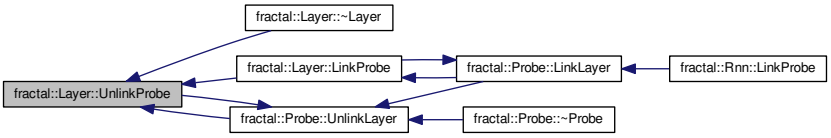
7.11.4.36 void fractal::Layer::UnlinkProbe ( )

Definition at line 283 of file Layer.cc.

Here is the call graph for this function:



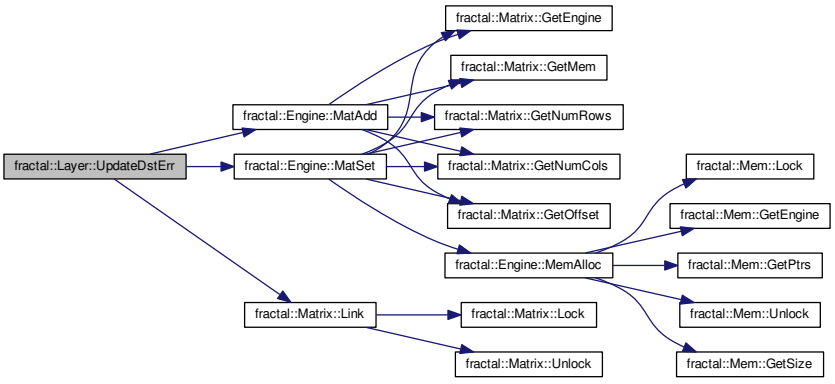
Here is the caller graph for this function:



7.11.4.37 void fractal::Layer::UpdateDstErr (const unsigned long *batchFrom*, const unsigned long *batchTo*) [protected]

Definition at line 423 of file Layer.cc.

Here is the call graph for this function:

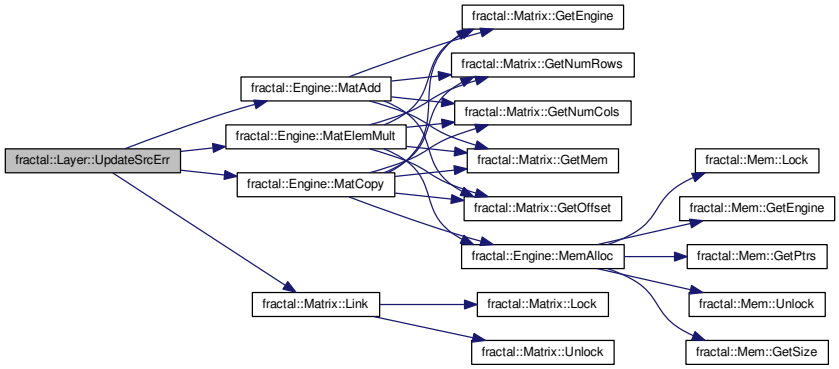


Here is the caller graph for this function:

7.11.4.38 void fractal::Layer::UpdateSrcErr (const unsigned long *batchFrom*, const unsigned long *batchTo*) [protected]

Definition at line 474 of file Layer.cc.

Here is the call graph for this function:

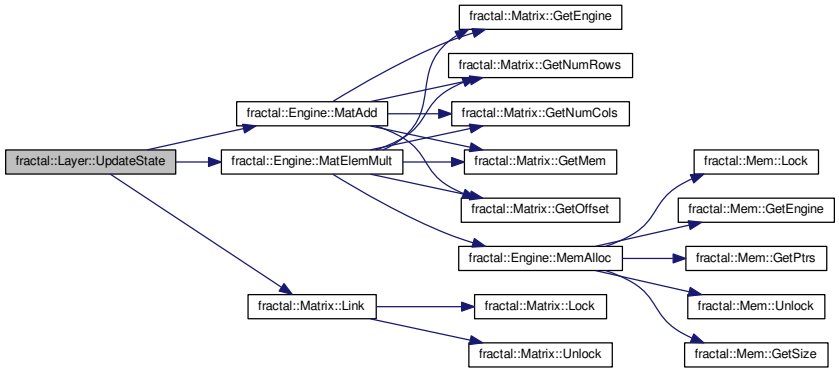


Here is the caller graph for this function:

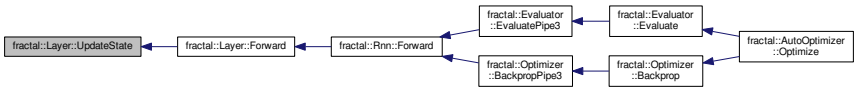
7.11.4.39 void fractal::Layer::UpdateState (const unsigned long *batchFrom*, const unsigned long *batchTo*) [protected]

Definition at line 356 of file Layer.cc.

Here is the call graph for this function:



Here is the caller graph for this function:



7.11.5 Member Data Documentation

7.11.5.1 Matrix<FLOAT> fractal::Layer::act [protected]

Definition at line 131 of file Layer.h.

7.11.5.2 ActType fractal::Layer::actType [protected]

Definition at line 122 of file Layer.h.

7.11.5.3 unsigned long fractal::Layer::batchSize [protected]

Definition at line 126 of file Layer.h.

7.11.5.4 friend fractal::Layer::Connection [protected]

Definition at line 145 of file Layer.h.

7.11.5.5 Matrix<FLOAT> fractal::Layer::dstErr [protected]

Definition at line 131 of file Layer.h.

7.11.5.6 ConnList fractal::Layer::dstList [protected]

Definition at line 129 of file Layer.h.

7.11.5.7 Engine* fractal::Layer::engine [protected]

Definition at line 120 of file Layer.h.

7.11.5.8 PEvent fractal::Layer::event [protected]

Definition at line 142 of file Layer.h.

7.11.5.9 long fractal::Layer::group [protected]

Definition at line 139 of file Layer.h.

7.11.5.10 long fractal::Layer::index [protected]

Definition at line 139 of file Layer.h.

7.11.5.11 FLOAT fractal::Layer::initVal [protected]

Definition at line 135 of file Layer.h.

7.11.5.12 bool fractal::Layer::isVisited [protected]

Definition at line 138 of file Layer.h.

7.11.5.13 Probe* fractal::Layer::linkedProbe [protected]

Definition at line 133 of file Layer.h.

7.11.5.14 std::string fractal::Layer::name [protected]

Definition at line 125 of file Layer.h.

7.11.5.15 friend fractal::Layer::Probe [protected]

Definition at line 144 of file Layer.h.

7.11.5.16 unsigned long fractal::Layer::size [protected]

Definition at line 126 of file Layer.h.

7.11.5.17 Matrix<FLOAT> fractal::Layer::srcErr [protected]

Definition at line 131 of file Layer.h.

7.11.5.18 ConnList fractal::Layer::srcList [protected]

Definition at line 128 of file Layer.h.

7.11.5.19 Matrix<FLOAT> fractal::Layer::state [protected]

Definition at line 131 of file Layer.h.

7.11.5.20 FLOAT fractal::Layer::statePenalty [protected]

Definition at line 135 of file Layer.h.

7.11.5.21 StateType fractal::Layer::stateType [protected]

Definition at line 123 of file Layer.h.

7.11.5.22 PStream* fractal::Layer::stream [protected]

Definition at line 141 of file Layer.h.

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

- src/core/Layer.h
- src/core/Layer.cc

7.12 fractal::LayerParam Class Reference

`#include <Layer.h>`

Public Member Functions

- LayerParam ()

Public Attributes

- FLOAT initVal
- FLOAT statePenalty

7.12.1 Detailed Description

Definition at line 43 of file Layer.h.

7.12.2 Constructor & Destructor Documentation

7.12.2.1 fractal::LayerParam::LayerParam ( ) [inline]

Definition at line 46 of file Layer.h.

7.12.3 Member Data Documentation

7.12.3.1 FLOAT fractal::LayerParam::initVal

Definition at line 48 of file Layer.h.

7.12.3.2 FLOAT fractal::LayerParam::statePenalty

Definition at line 49 of file Layer.h.

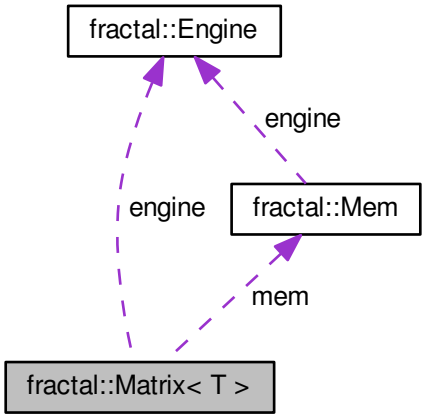
The documentation for this class was generated from the following file:

- src/core/Layer.h

7.13 fractal::Matrix< T > Class Template Reference

`#include <Matrix.h>`

Collaboration diagram for fractal::Matrix< T >:



Public Member Functions

- Matrix (const unsigned long nRows=0, const unsigned long nCols=1)
- Matrix (Matrix< T > &A, const unsigned long a1, const unsigned long a2)
- virtual ~Matrix ()
- void SetEngine (Engine *engine)
- const unsigned long GetOffset () const
- const unsigned long GetNumRows () const
- const unsigned long GetNumCols () const
- const Engine * GetEngine () const
- Mem * GetMem ()
- FLOAT *const GetHostData ()
- void HostPush ()
- void HostPull (PStream &stream)
- void Resize (const unsigned long nRows, const unsigned long nCols)
- void Link (Matrix< T > &src)
- void Unlink ()
- void Import (const std::vector< T > &vec, PStream &stream)
- void Import (const Matrix< T > &mat, PStream &stream)
- void Export (std::vector< T > &vec, PStream &stream) const
- void Export (Matrix< T > &mat, PStream &stream) const
- void Swap (Matrix< T > &mat)
- void Lock ()
- void Unlock ()
- void Save (const std::string &filename)
- void Load (const std::string &filename)

Protected Member Functions

- Matrix (const Matrix< T > &)
- void Malloc ()
- void Clear ()

Protected Attributes

- Mem * mem
- unsigned long offset
- unsigned long nRows
- unsigned long nCols
- bool isSub
- std::recursive_mutex mtx
- Engine * engine

7.13.1 Detailed Description

template<class T>class fractal::Matrix< T >

Definition at line 37 of file Matrix.h.

7.13.2 Constructor & Destructor Documentation

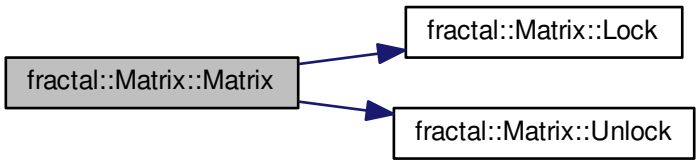
7.13.2.1 template<class T > fractal::Matrix< T >::Matrix (const unsigned long *nRows* = 0, const unsigned long *nCols* = 1)

Definition at line 29 of file Matrix.cc.

7.13.2.2 template<class T> fractal::Matrix< T >::Matrix (Matrix< T > & *A*, const unsigned long *a1*, const unsigned long *a2*)

Definition at line 43 of file Matrix.cc.

Here is the call graph for this function:



7.13.2.3 template<class T > fractal::Matrix< T >::~Matrix ( ) [virtual]

Definition at line 67 of file Matrix.cc.

7.13.2.4 template<class T> fractal::Matrix< T >::Matrix (const Matrix< T > &) [protected]

7.13.3 Member Function Documentation

7.13.3.1 template<class T > void fractal::Matrix< T >::Clear ( ) [protected]

Definition at line 121 of file Matrix.cc.

7.13.3.2 template<class T> void fractal::Matrix< T >::Export (std::vector< T > & *vec,* PStream & *stream*) const

Definition at line 209 of file Matrix.cc.

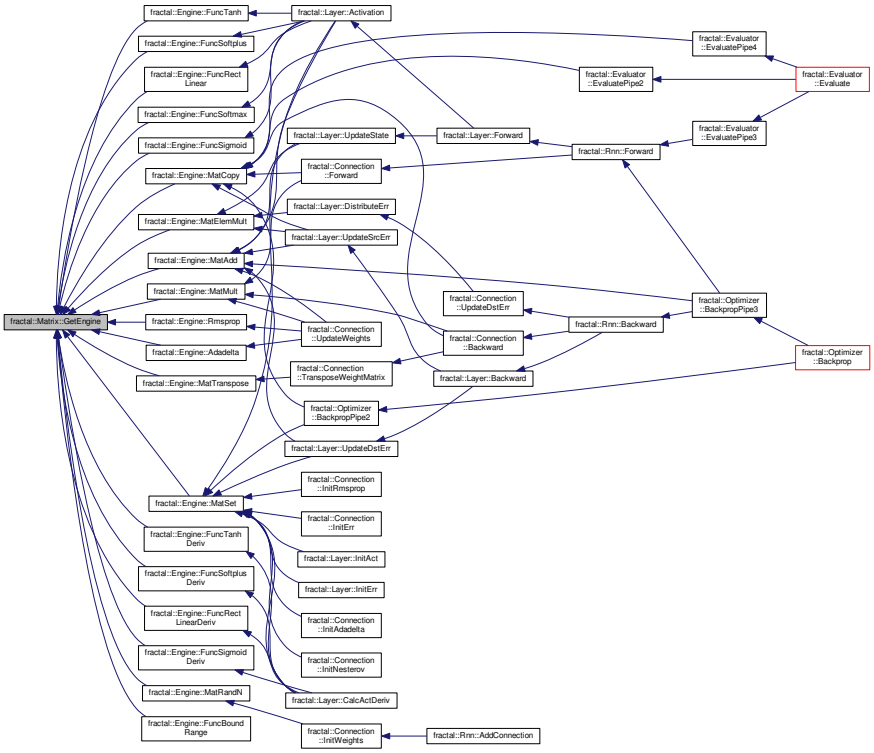
7.13.3.3 template<class T> void fractal::Matrix< T >::Export (Matrix< T > & *mat,* PStream & *stream*) const

Definition at line 218 of file Matrix.cc.

7.13.3.4 template<class T> const Engine* fractal::Matrix< T >::GetEngine ( ) const [inline]

Definition at line 49 of file Matrix.h.

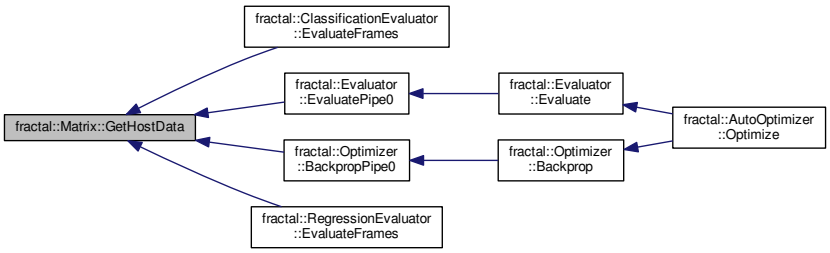
Here is the caller graph for this function:



7.13.3.5 template<class T > FLOAT *const fractal::Matrix< T >::GetHostData ( )

Definition at line 231 of file Matrix.cc.

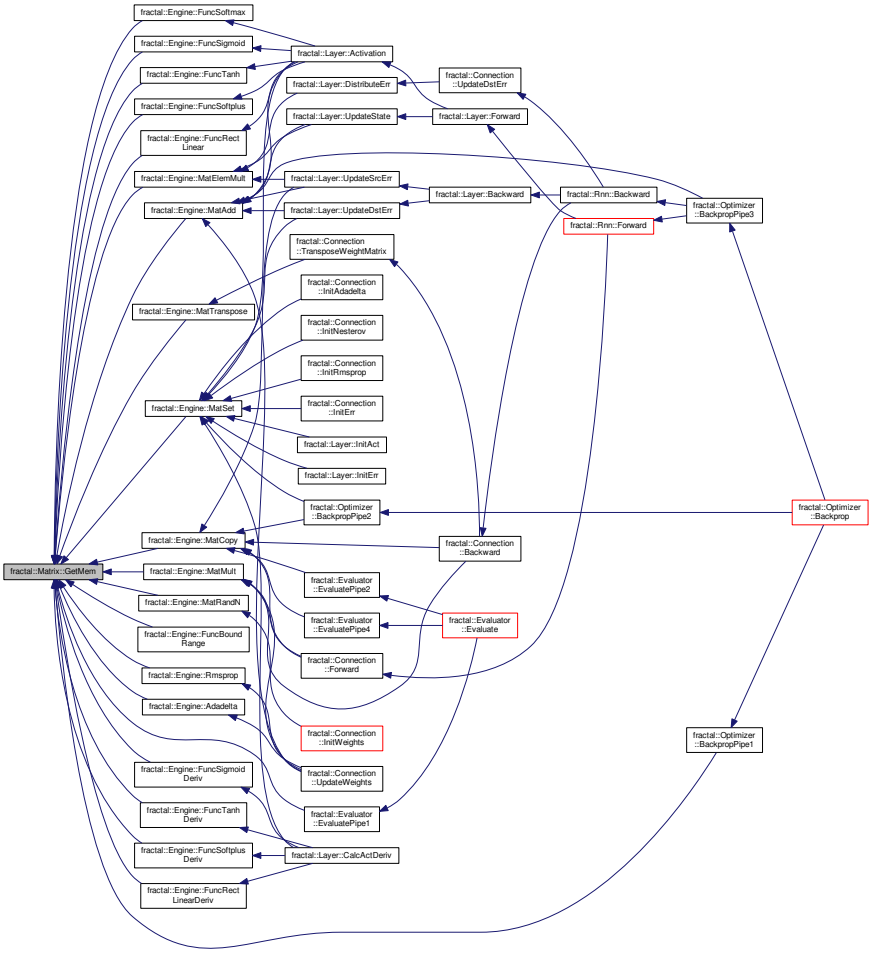
Here is the caller graph for this function:



7.13.3.6 template<class T> Mem* fractal::Matrix< T >::GetMem ( ) [inline]

Definition at line 50 of file Matrix.h.

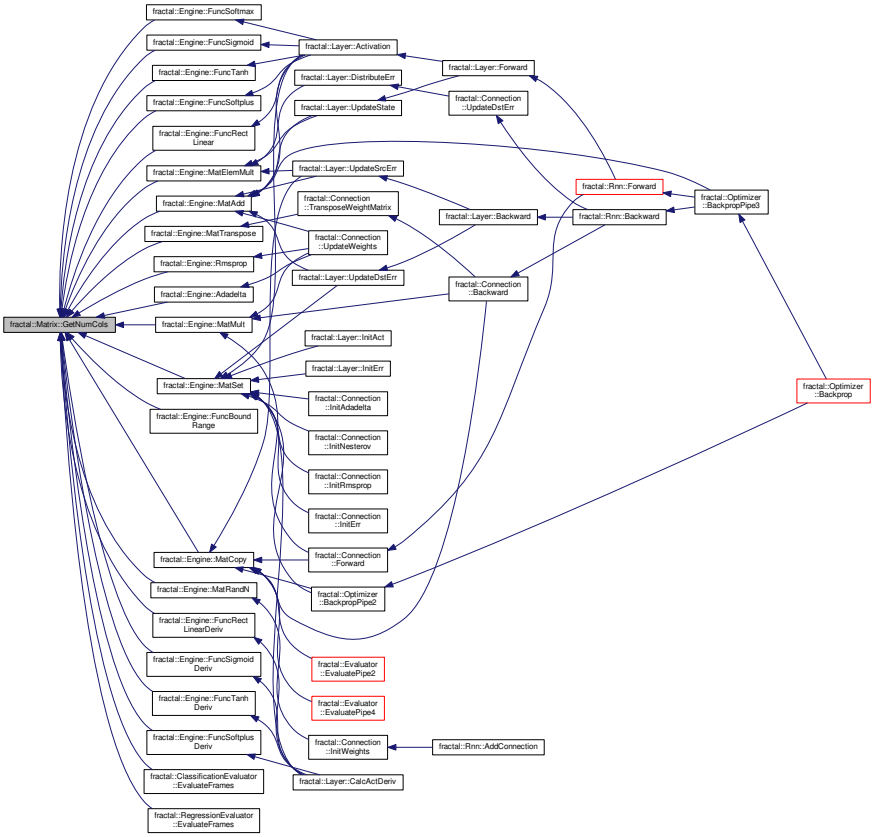
Here is the caller graph for this function:



7.13.3.7 template<class T> const unsigned long fractal::Matrix< T >::GetNumCols ( ) const [inline]

Definition at line 48 of file Matrix.h.

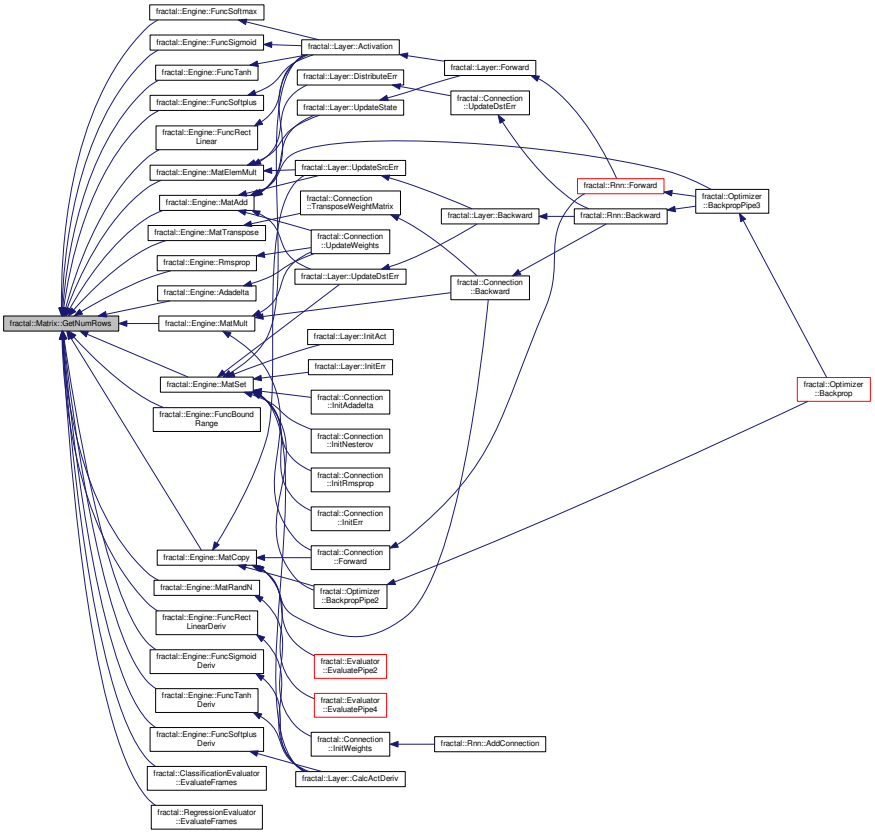
Here is the caller graph for this function:



7.13.3.8 template<class T> const unsigned long fractal::Matrix< T >::GetNumRows ( ) const [inline]

Definition at line 47 of file Matrix.h.

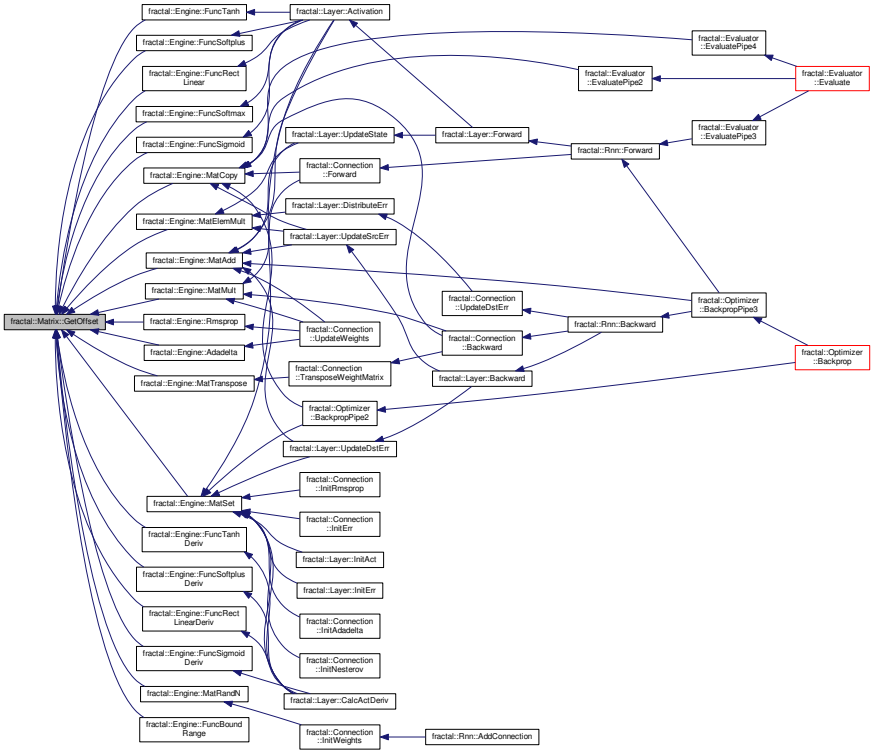
Here is the caller graph for this function:



7.13.3.9 template<class T> const unsigned long fractal::Matrix< T >::GetOffset ( ) const [inline]

Definition at line 46 of file Matrix.h.

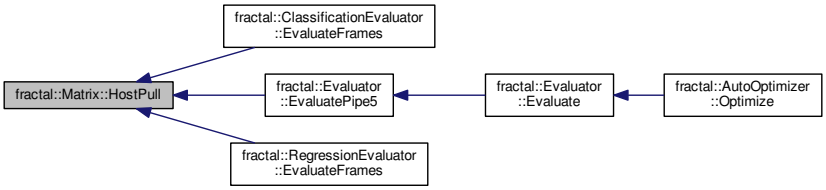
Here is the caller graph for this function:



7.13.3.10 template<class T > void fractal::Matrix< T >::HostPull (PStream & *stream*)

Definition at line 260 of file Matrix.cc.

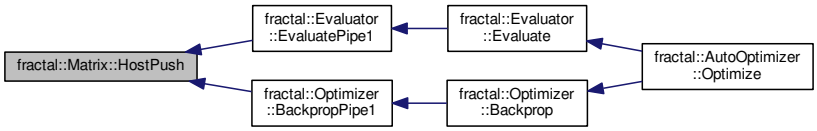
Here is the caller graph for this function:



7.13.3.11 template<class T > void fractal::Matrix< T >::HostPush ( )

Definition at line 250 of file Matrix.cc.

Here is the caller graph for this function:



7.13.3.12 template<class T> void fractal::Matrix< T >::Import (const std::vector< T > & *vec,* PStream & *stream*)

Definition at line 187 of file Matrix.cc.

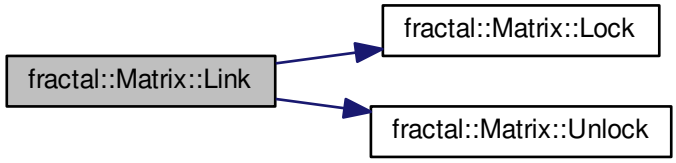
7.13.3.13 template<class T> void fractal::Matrix< T >::Import (const Matrix< T > & *mat,* PStream & *stream*)

Definition at line 196 of file Matrix.cc.

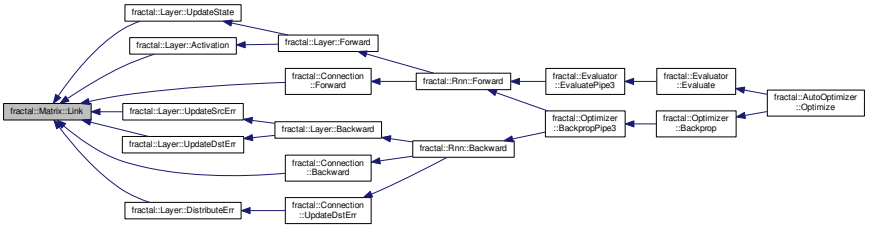
7.13.3.14 template<class T> void fractal::Matrix< T >::Link (Matrix< T > & *src*)

Definition at line 133 of file Matrix.cc.

Here is the call graph for this function:



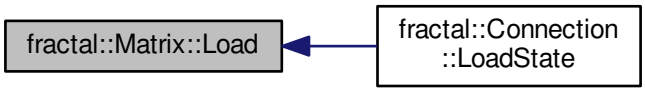
Here is the caller graph for this function:



7.13.3.15 template<class T > void fractal::Matrix< T >::Load (const std::string & *filename*)

Definition at line 334 of file Matrix.cc.

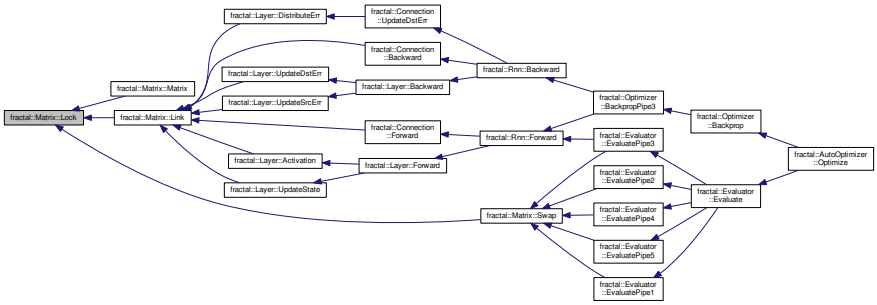
Here is the caller graph for this function:



7.13.3.16 template<class T> void fractal::Matrix< T >::Lock ( ) [inline]

Definition at line 67 of file Matrix.h.

Here is the caller graph for this function:



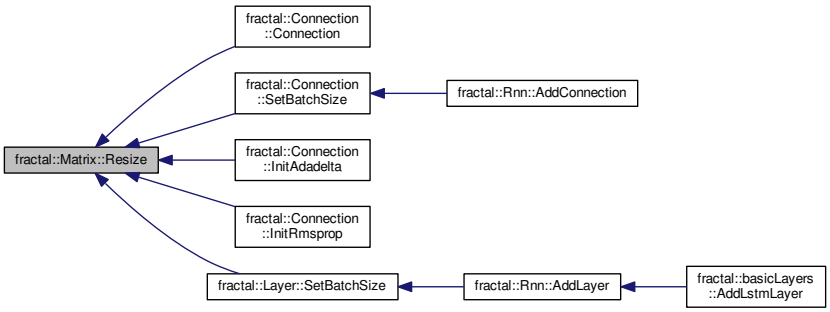
7.13.3.17 template<class T > void fractal::Matrix< T >::Malloc ( ) [protected]

Definition at line 107 of file Matrix.cc.

7.13.3.18 template<class T > void fractal::Matrix< T >::Resize (const unsigned long *nRows,* const unsigned long *nCols*)

Definition at line 89 of file Matrix.cc.

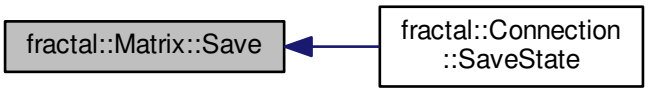
Here is the caller graph for this function:



7.13.3.19 template<class T > void fractal::Matrix< T >::Save (const std::string & *filename*)

Definition at line 294 of file Matrix.cc.

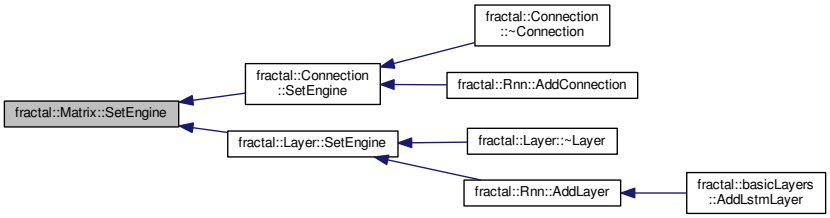
Here is the caller graph for this function:



7.13.3.20 template<class T > void fractal::Matrix< T >::SetEngine (Engine * *engine*)

Definition at line 74 of file Matrix.cc.

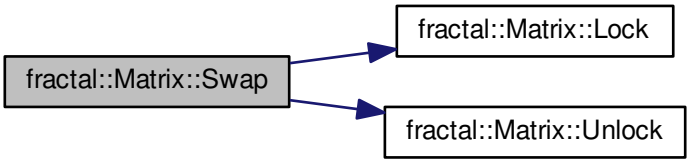
Here is the caller graph for this function:



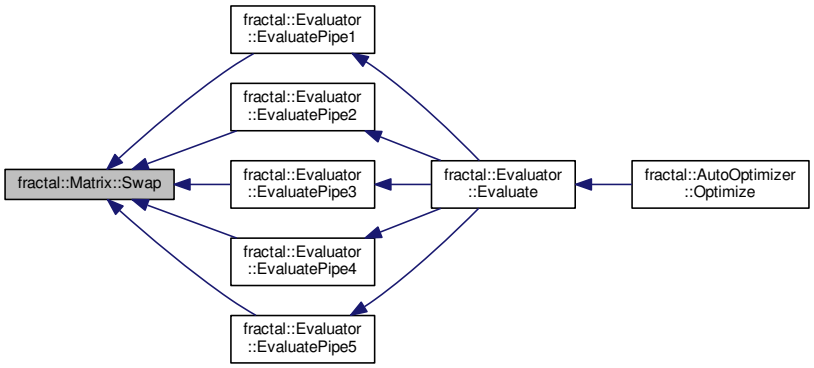
7.13.3.21 template<class T> void fractal::Matrix< T >::Swap (Matrix< T > & *mat*)

Definition at line 270 of file Matrix.cc.

Here is the call graph for this function:



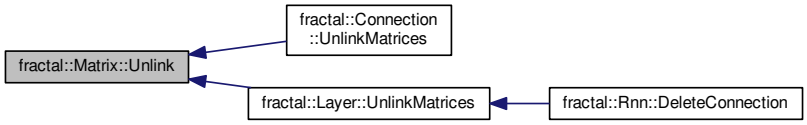
Here is the caller graph for this function:



7.13.3.22 template<class T > void fractal::Matrix< T >::Unlink ( )

Definition at line 164 of file Matrix.cc.

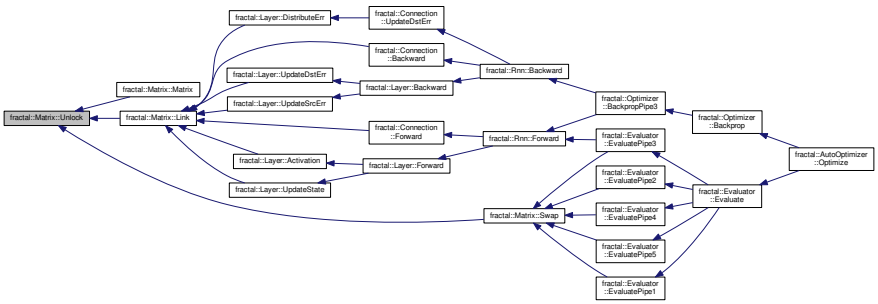
Here is the caller graph for this function:



7.13.3.23 template<class T> void fractal::Matrix< T >::Unlock ( ) [inline]

Definition at line 68 of file Matrix.h.

Here is the caller graph for this function:



7.13.4 Member Data Documentation

7.13.4.1 template<class T> Engine* fractal::Matrix< T >::engine [protected]

Definition at line 86 of file Matrix.h.

7.13.4.2 template<class T> bool fractal::Matrix< T >::isSub [protected]

Definition at line 83 of file Matrix.h.

7.13.4.3 template<class T> Mem* fractal::Matrix< T >::mem [protected]

Definition at line 80 of file Matrix.h.

7.13.4.4 template<class T> std::recursive_mutex fractal::Matrix< T >::mtx [protected]

Definition at line 85 of file Matrix.h.

7.13.4.5 template<class T> unsigned long fractal::Matrix< T >::nCols [protected]

Definition at line 82 of file Matrix.h.

7.13.4.6 template<class T> unsigned long fractal::Matrix< T >::nRows [protected]

Definition at line 82 of file Matrix.h.

7.13.4.7 template<class T> unsigned long fractal::Matrix< T >::offset [protected]

Definition at line 81 of file Matrix.h.

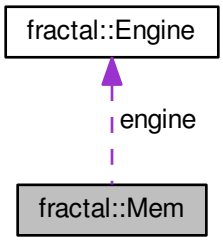
The documentation for this class was generated from the following files:

- src/core/Matrix.h
- src/core/Matrix.cc

7.14 fractal::Mem Class Reference

`#include <Mem.h>`

Collaboration diagram for fractal::Mem:



Public Member Functions

- Mem (Engine ∗const engine, size_t size)
- virtual ∼Mem ()
- const Engine ∗ GetEngine () const
- void ∗const GetPtr (const unsigned long loc) const
- void ∗∗const GetPtrs ()
- const bool IsRealValid (const unsigned long loc)
- const bool IsValid (const unsigned long loc) const
- const unsigned long GetRecentLoc () const
- const size_t GetSize () const
- void SetSize (size_t size)
- void CopyFromHost (const size_t offsetDst, const void ∗ptrSrc, const size_t size, PStream &stream)
- void CopyToHost (const size_t offsetSrc, void ∗ptrDst, const size_t size, PStream &stream) const
- void Validate (const unsigned long loc)
- void Invalidate ()
- void Pull (const unsigned long loc, PStream &stream)
- void Push (const unsigned long loc)
- void Lock ()
- void Unlock ()

Protected Member Functions

- Mem (const Mem &mem)

Protected Attributes

- unsigned long numLoc
- unsigned long recentLoc
- size_t size
- void ∗∗ ptr
- bool ∗ valid
- std::recursive_mutex mtx
- Engine ∗ engine

7.14.1 Detailed Description

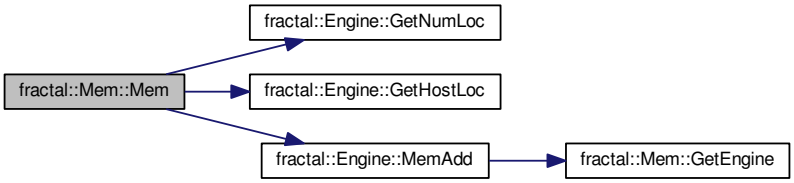
Definition at line 33 of file Mem.h.

7.14.2 Constructor & Destructor Documentation

7.14.2.1 fractal::Mem::Mem (Engine ∗const *engine*, size_t *size*)

Definition at line 25 of file Mem.cc.

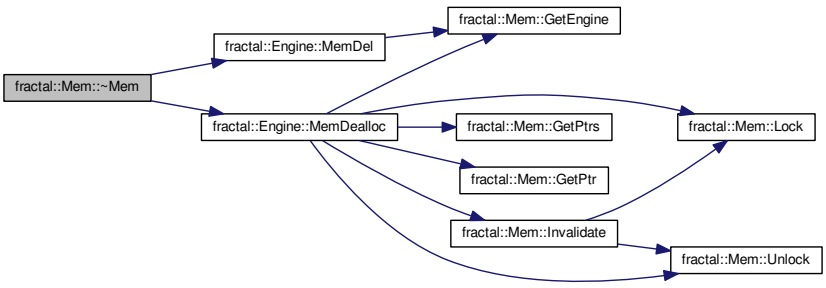
Here is the call graph for this function:



7.14.2.2 fractal::Mem::∼Mem ( ) [virtual]

Definition at line 45 of file Mem.cc.

Here is the call graph for this function:



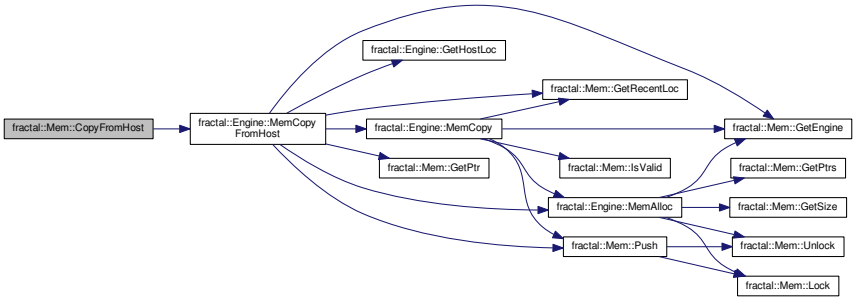
7.14.2.3 fractal::Mem::Mem (const Mem & *mem*) [protected]

7.14.3 Member Function Documentation

7.14.3.1 void fractal::Mem::CopyFromHost (const size_t *offsetDst*, const void ∗ *ptrSrc*, const size_t *size*, PStream & *stream*)

Definition at line 55 of file Mem.cc.

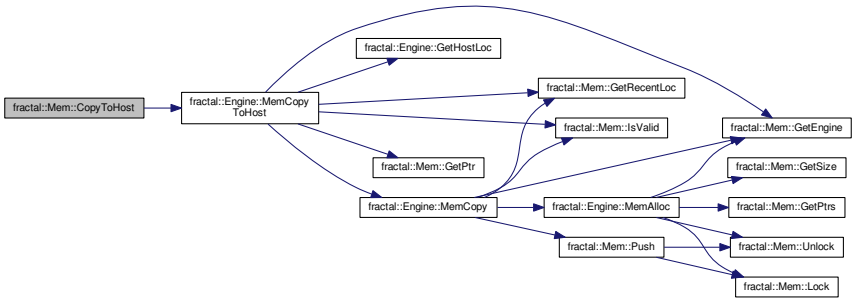
Here is the call graph for this function:



7.14.3.2 void fractal::Mem::CopyToHost (const size_t *offsetSrc,* void * *ptrDst,* const size_t *size,* PStream & *stream*) const

Definition at line 63 of file Mem.cc.

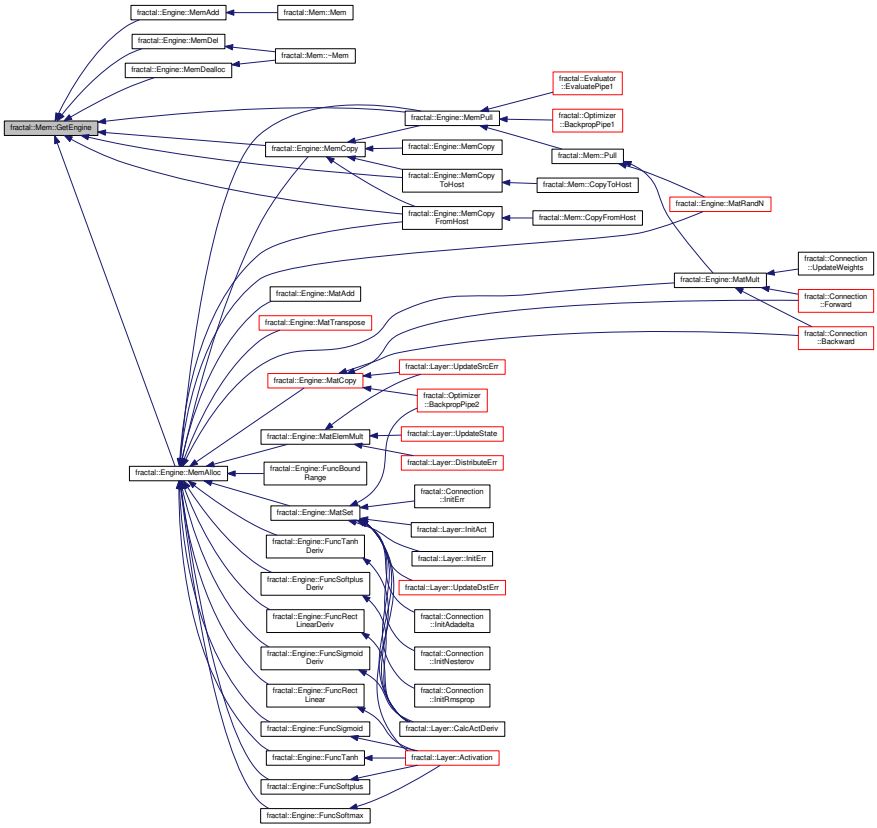
Here is the call graph for this function:



7.14.3.3 const Engine* fractal::Mem::GetEngine ( ) const [inline]

Definition at line 39 of file Mem.h.

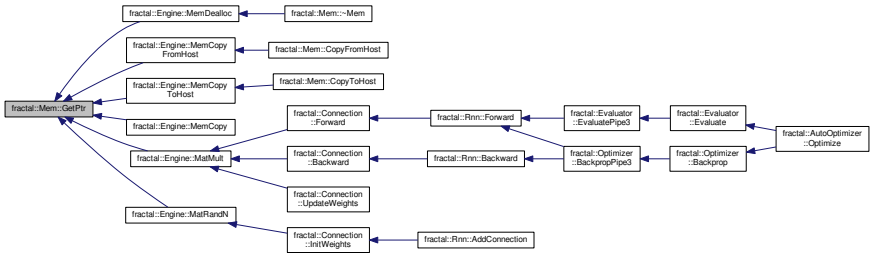
Here is the caller graph for this function:



7.14.3.4 void* const fractal::Mem::GetPtr (const unsigned long *loc*) const [inline]

Definition at line 40 of file Mem.h.

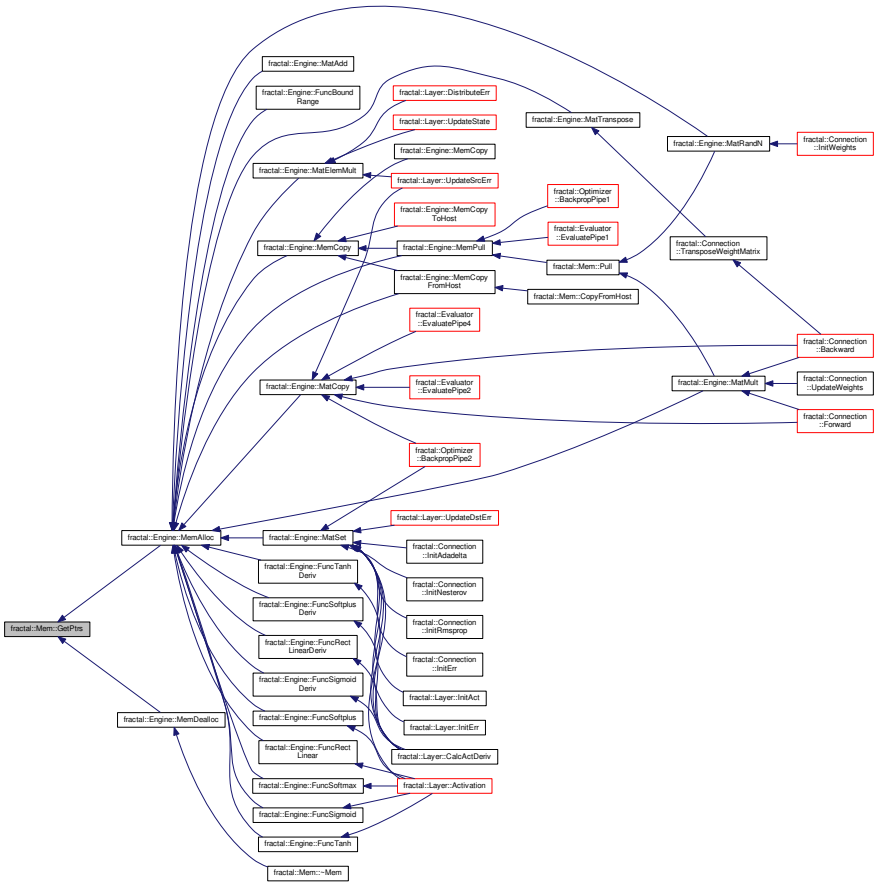
Here is the caller graph for this function:



7.14.3.5 void** const fractal::Mem::GetPtrs ( ) [inline]

Definition at line 41 of file Mem.h.

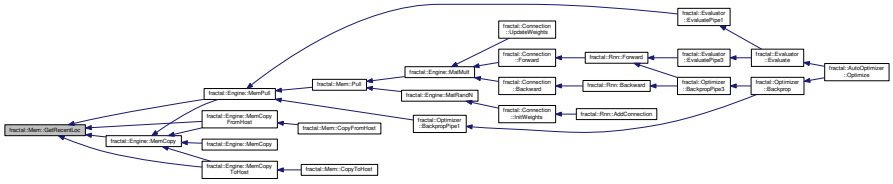
Here is the caller graph for this function:



7.14.3.6 const unsigned long fractal::Mem::GetRecentLoc ( ) const [inline]

Definition at line 44 of file Mem.h.

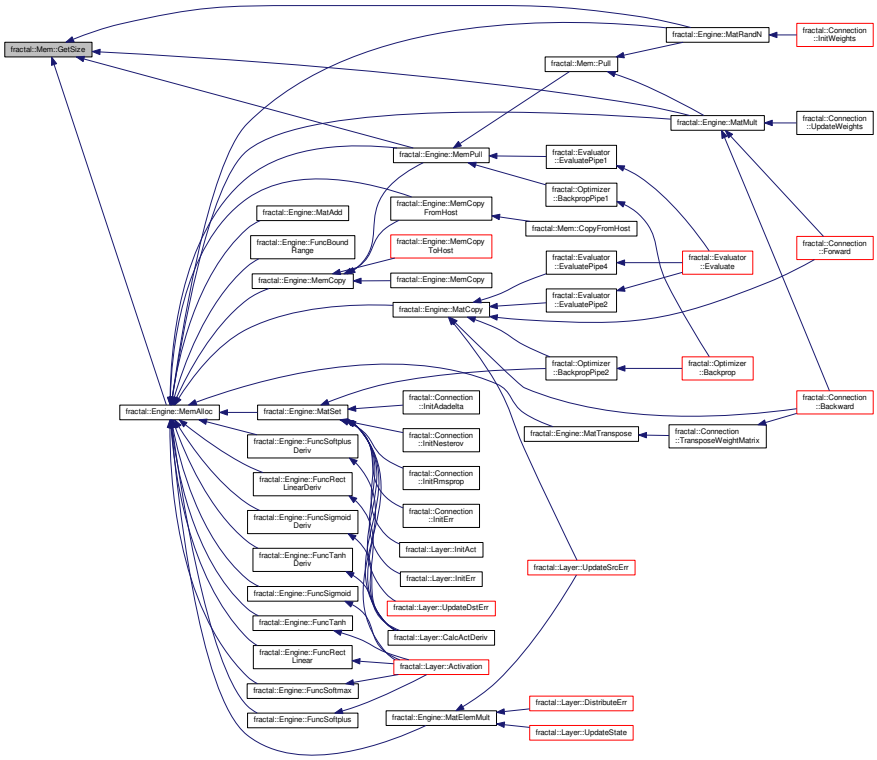
Here is the caller graph for this function:



7.14.3.7 const size_t fractal::Mem::GetSize ( ) const [inline]

Definition at line 45 of file Mem.h.

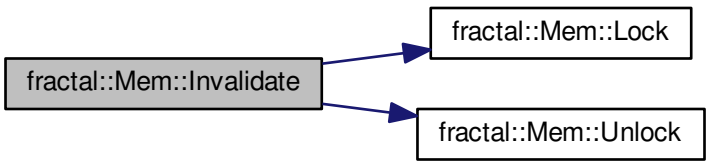
Here is the caller graph for this function:



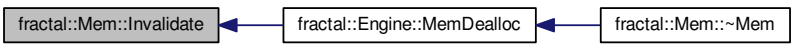
7.14.3.8 void fractal::Mem::Invalidate ( )

Definition at line 82 of file Mem.cc.

Here is the call graph for this function:



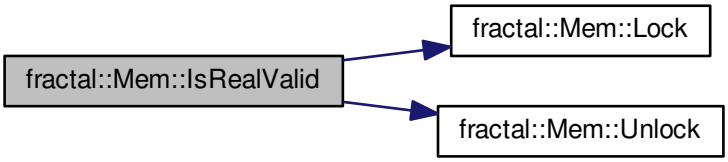
Here is the caller graph for this function:



7.14.3.9 const bool fractal::Mem::IsRealValid (const unsigned long *loc*) [inline]

Definition at line 42 of file Mem.h.

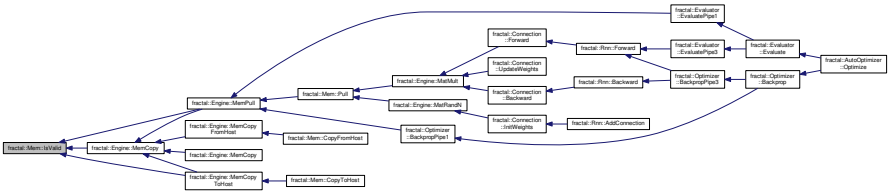
Here is the call graph for this function:



7.14.3.10 const bool fractal::Mem::IsValid (const unsigned long *loc*) const [inline]

Definition at line 43 of file Mem.h.

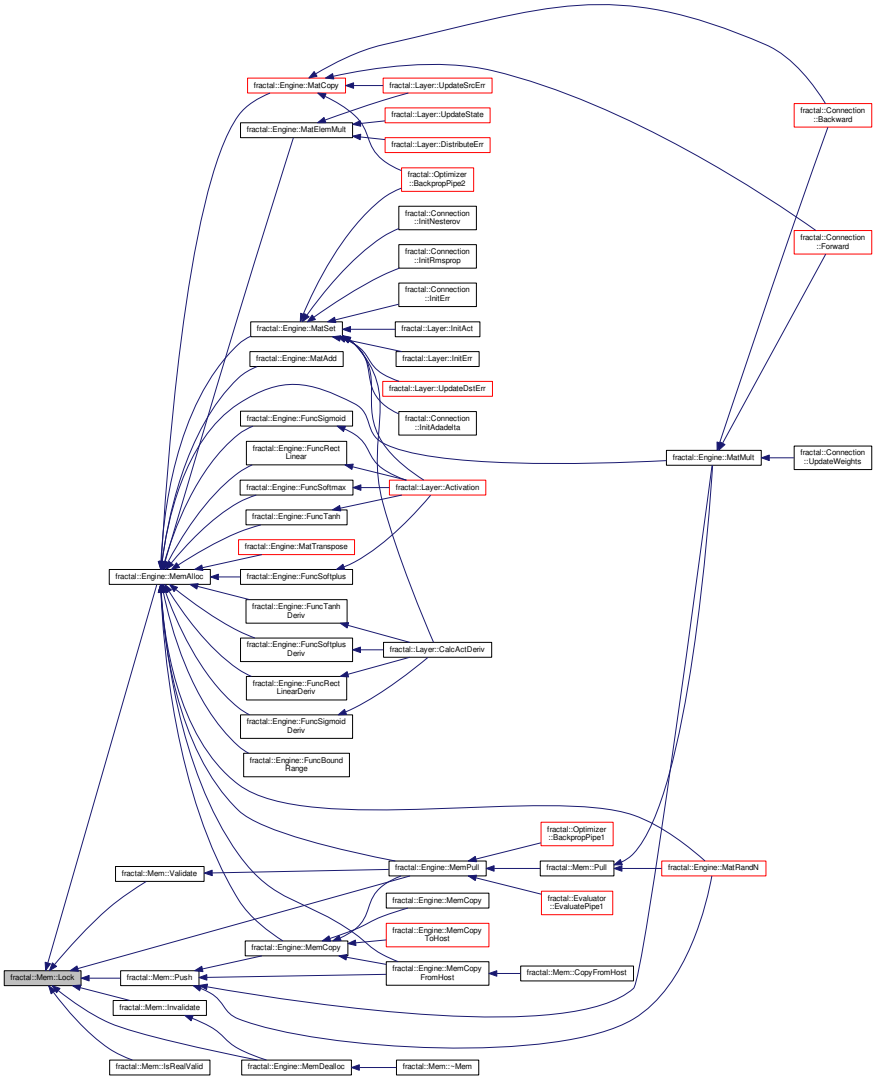
Here is the caller graph for this function:



7.14.3.11 void fractal::Mem::Lock ( ) [inline]

Definition at line 57 of file Mem.h.

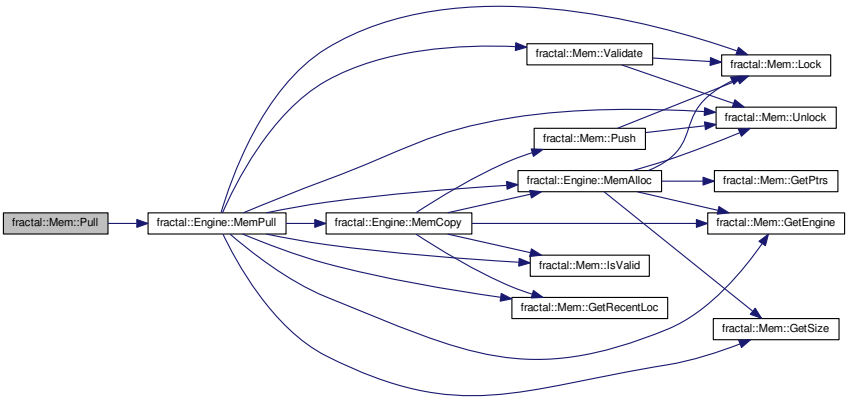
Here is the caller graph for this function:



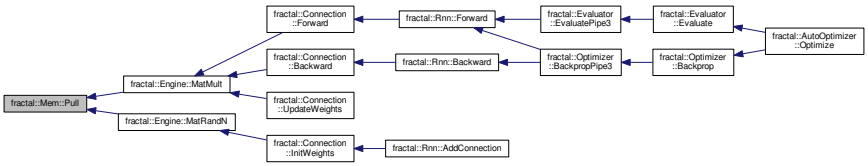
7.14.3.12 void fractal::Mem::Pull (const unsigned long *loc*, PStream & *stream*)

Definition at line 95 of file Mem.cc.

Here is the call graph for this function:



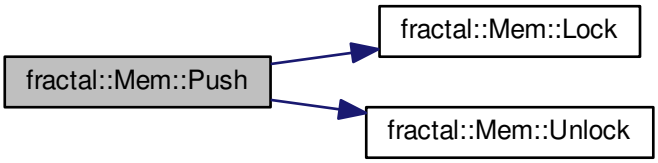
Here is the caller graph for this function:



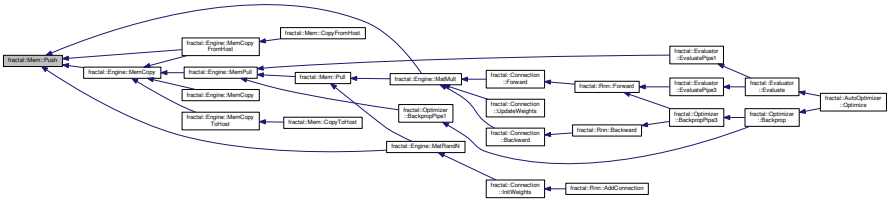
7.14.3.13 void fractal::Mem::Push (const unsigned long *loc*)

Definition at line 101 of file Mem.cc.

Here is the call graph for this function:



Here is the caller graph for this function:



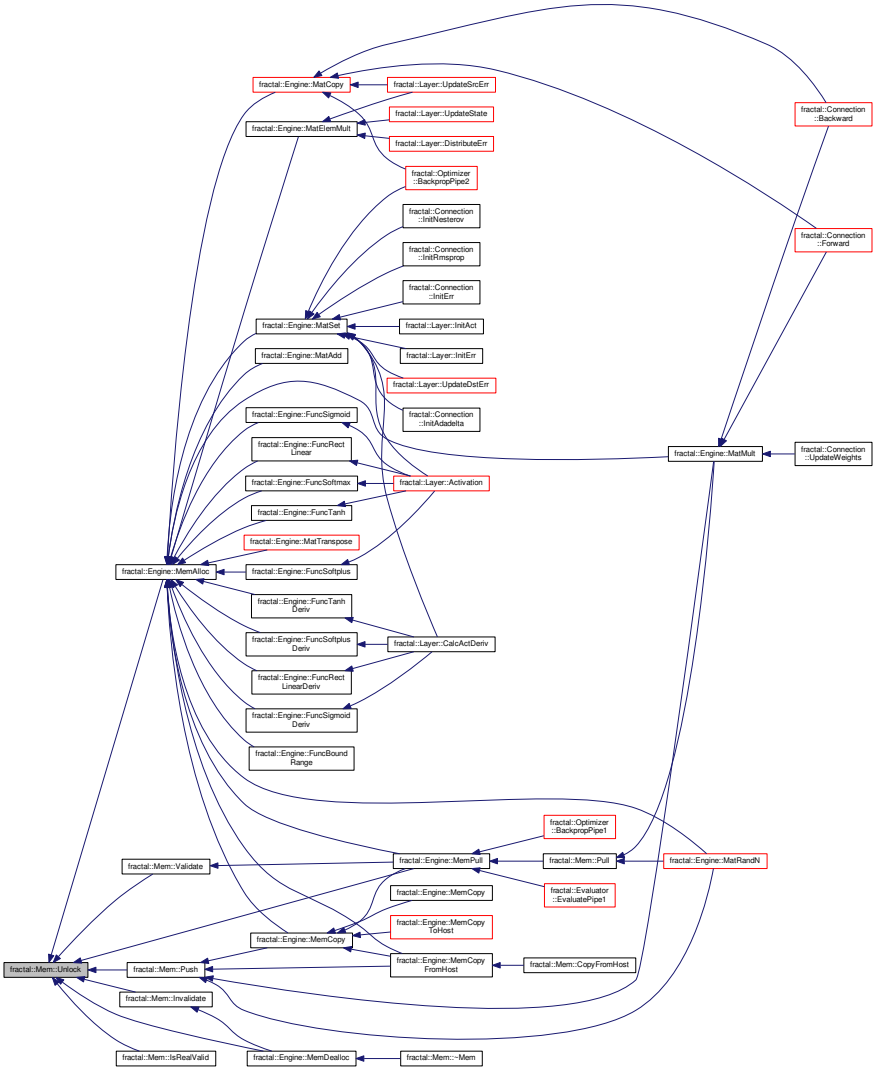
7.14.3.14 void fractal::Mem::SetSize (size_t *size*) [inline]

Definition at line 46 of file Mem.h.

7.14.3.15 void fractal::Mem::Unlock ( ) [inline]

Definition at line 58 of file Mem.h.

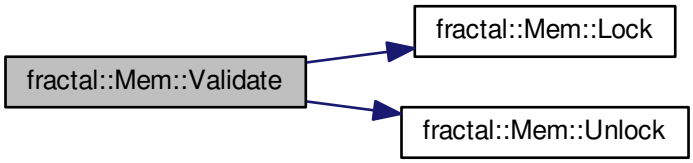
Here is the caller graph for this function:



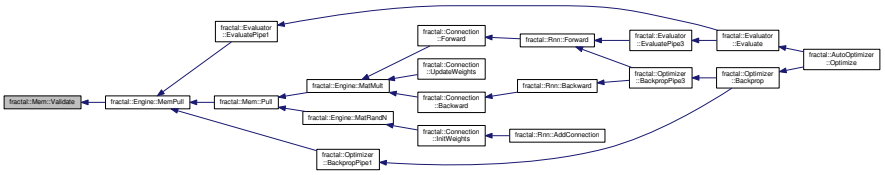
7.14.3.16 void fractal::Mem::Validate (const unsigned long *loc*)

Definition at line 71 of file Mem.cc.

Here is the call graph for this function:



Here is the caller graph for this function:



7.14.4 Member Data Documentation

7.14.4.1 Engine* fractal::Mem::engine [protected]

Definition at line 71 of file Mem.h.

7.14.4.2 std::recursive_mutex fractal::Mem::mtx [protected]

Definition at line 69 of file Mem.h.

7.14.4.3 unsigned long fractal::Mem::numLoc [protected]

Definition at line 63 of file Mem.h.

7.14.4.4 void** fractal::Mem::ptr [protected]

Definition at line 66 of file Mem.h.

7.14.4.5 unsigned long fractal::Mem::recentLoc [protected]

Definition at line 64 of file Mem.h.

7.14.4.6 size_t fractal::Mem::size [protected]

Definition at line 65 of file Mem.h.

7.14.4.7 bool* fractal::Mem::valid [protected]

Definition at line 67 of file Mem.h.

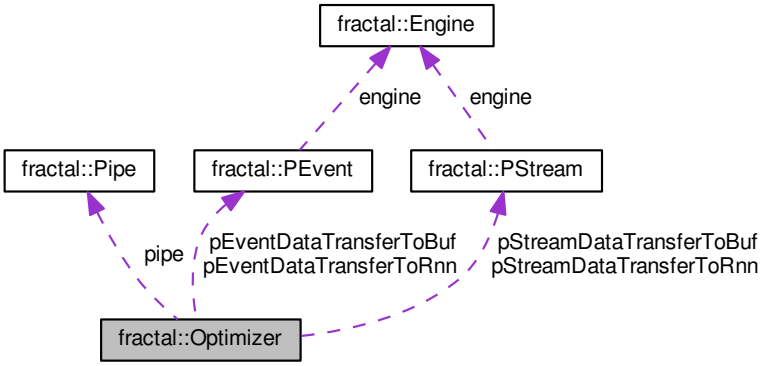
The documentation for this class was generated from the following files:

- src/core/Mem.h
- src/core/Mem.cc

7.15 fractal::Optimizer Class Reference

`#include <Optimizer.h>`

Collaboration diagram for fractal::Optimizer:



Public Member Functions

- Optimizer ()
- virtual ~Optimizer ()
- void Backprop (Rnn &rnn, Stream &stream, const PortMapList &inputPorts, const PortMapList &outputPorts, const unsigned long numFrame, const unsigned long windowSize, const unsigned long stepSize)
- void SetLearningRate (const FLOAT val)
- void SetMomentum (const FLOAT val)
- void SetAdadelta (const bool val)
- void SetRmsprop (const bool val)
- const FLOAT GetLearningRate ()
- const FLOAT GetMomentum ()
- const bool GetAdadelta ()
- const bool GetRmsprop ()

Static Protected Member Functions

- static void BackpropPipe0 (Optimizer *optimizer, BackpropArgs &args)
- static void BackpropPipe1 (Optimizer *optimizer, BackpropArgs &args)
- static void BackpropPipe2 (Optimizer *optimizer, BackpropArgs &args)
- static void BackpropPipe3 (Optimizer *optimizer, BackpropArgs &args)

Protected Attributes

- FLOAT learningRate
- FLOAT momentum
- Pipe pipe [4]
- PStream pStreamDataTransferToBuf
- PStream pStreamDataTransferToRnn
- PEvent pEventDataTransferToBuf
- PEvent pEventDataTransferToRnn
- bool adadelta
- bool rmsprop

7.15.1 Detailed Description

Definition at line 55 of file Optimizer.h.

7.15.2 Constructor & Destructor Documentation

7.15.2.1 fractal::Optimizer::Optimizer ( )

Definition at line 35 of file Optimizer.cc.

7.15.2.2 fractal::Optimizer::~Optimizer ( ) [virtual]

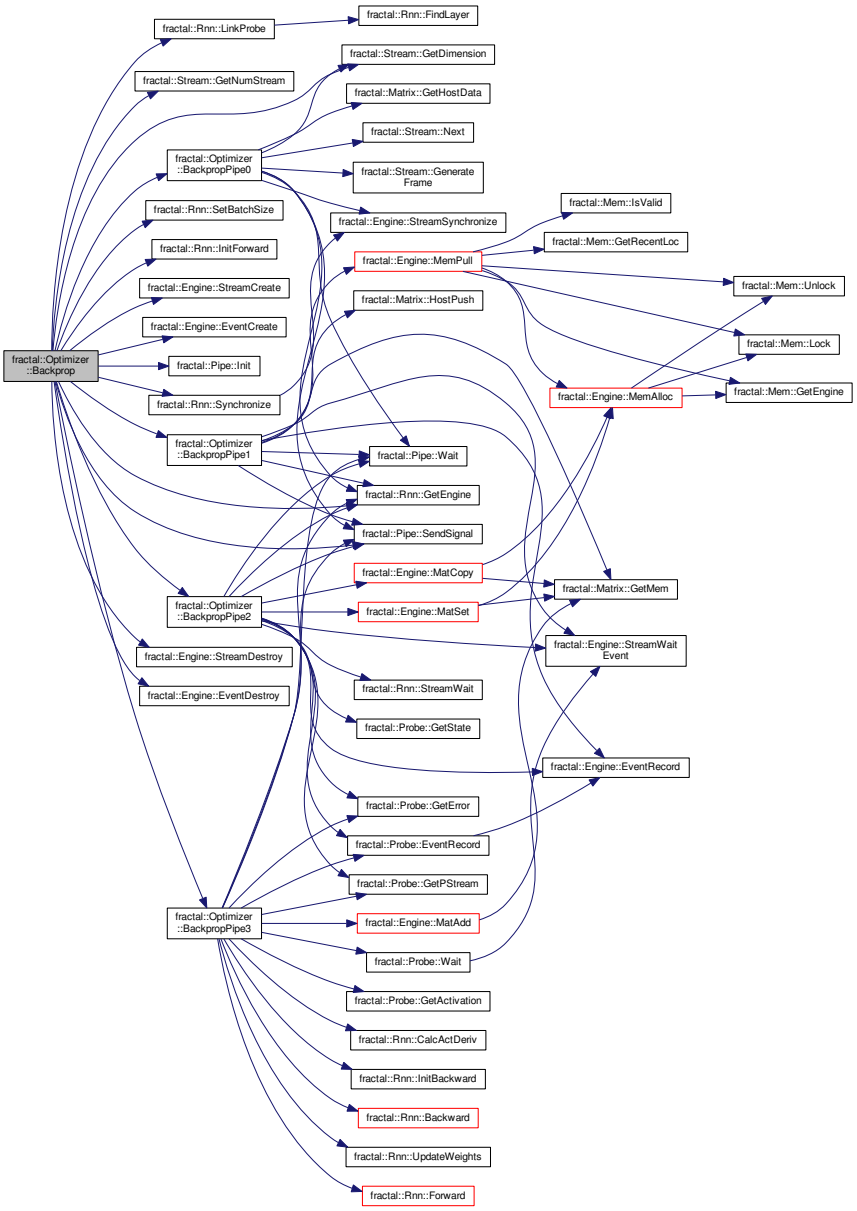
Definition at line 46 of file Optimizer.cc.

7.15.3 Member Function Documentation

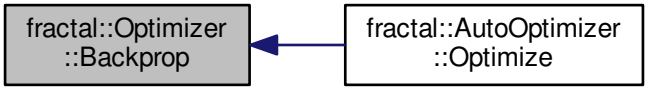
7.15.3.1 void fractal::Optimizer::Backprop (Rnn & *rnn,* Stream & *stream,* const PortMapList & *inputPorts,* const PortMapList & *outputPorts,* const unsigned long *numFrame,* const unsigned long *windowSize,* const unsigned long *stepSize*)

Definition at line 296 of file Optimizer.cc.

Here is the call graph for this function:



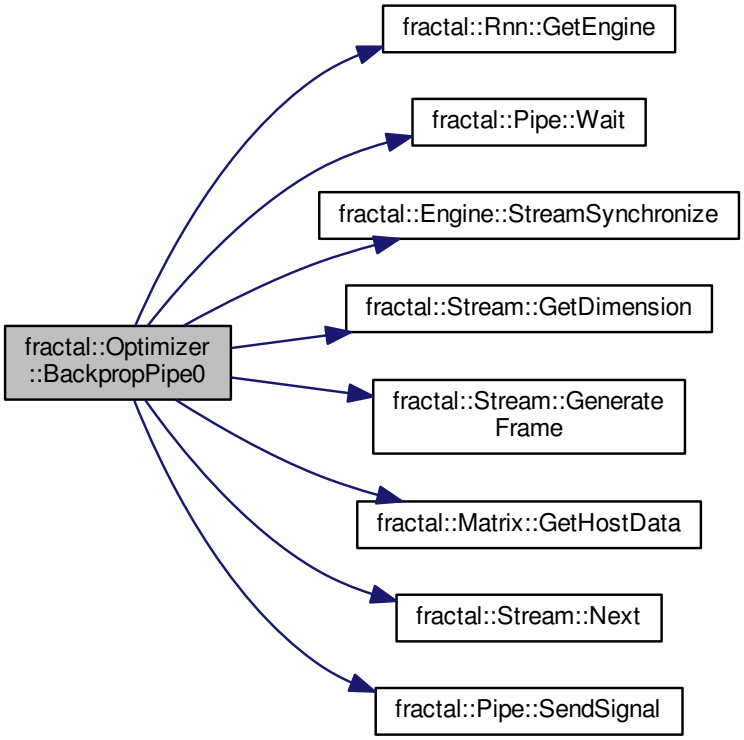
Here is the caller graph for this function:



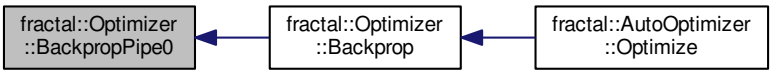
7.15.3.2 void fractal::Optimizer::BackpropPipe0 (Optimizer * *optimizer*, BackpropArgs & *args*) [static], [protected]

Definition at line 413 of file Optimizer.cc.

Here is the call graph for this function:



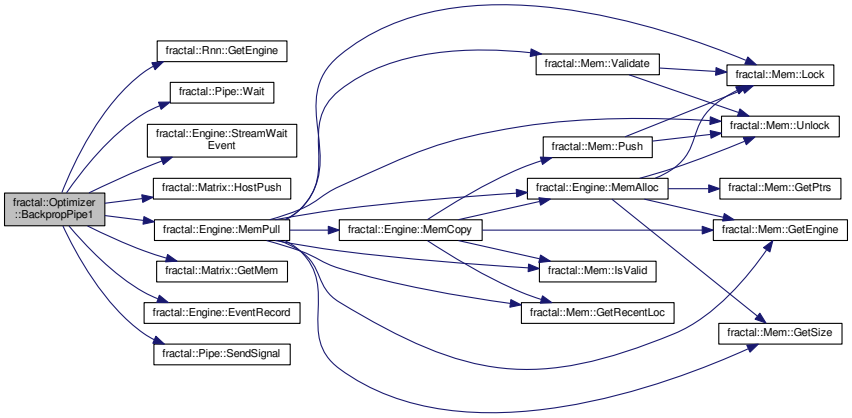
Here is the caller graph for this function:



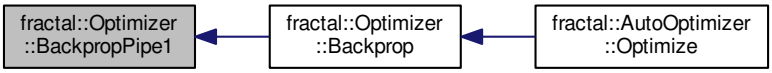
7.15.3.3 void fractal::Optimizer::BackpropPipe1 (Optimizer * *optimizer*, BackpropArgs & *args*) [static], [protected]

Definition at line 460 of file Optimizer.cc.

Here is the call graph for this function:



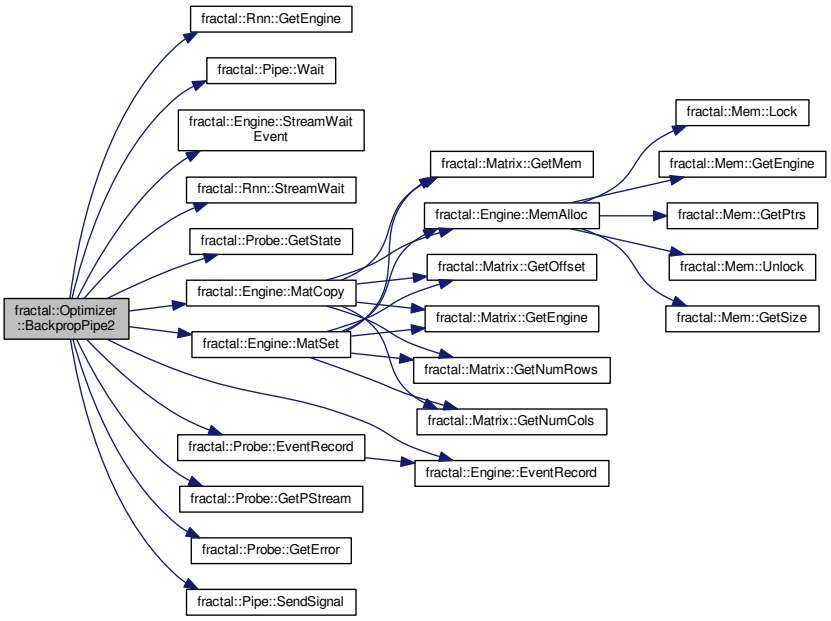
Here is the caller graph for this function:



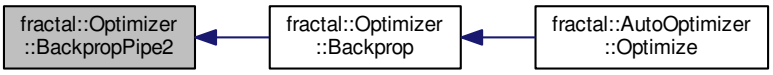
7.15.3.4 void fractal::Optimizer::BackpropPipe2 (Optimizer * *optimizer,* BackpropArgs & *args*) [static], [protected]

Definition at line 494 of file Optimizer.cc.

Here is the call graph for this function:



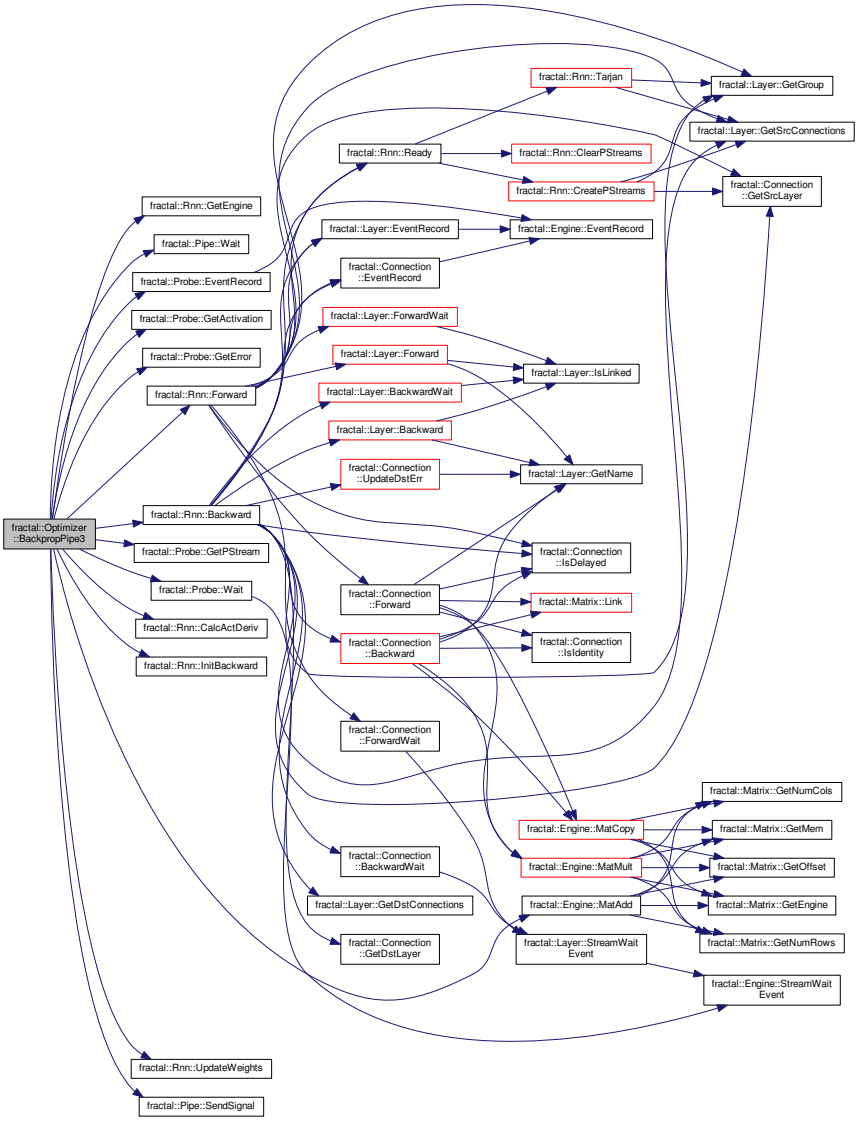
Here is the caller graph for this function:



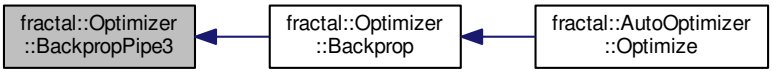
7.15.3.5 void fractal::Optimizer::BackpropPipe3 (Optimizer * *optimizer*, BackpropArgs & *args*) [static], [protected]

Definition at line 545 of file Optimizer.cc.

Here is the call graph for this function:



Here is the caller graph for this function:



7.15.3.6 const bool fractal::Optimizer::GetAdadelta ( ) [inline]

Definition at line 72 of file Optimizer.h.

7.15.3.7 const FLOAT fractal::Optimizer::GetLearningRate ( ) [inline]

Definition at line 70 of file Optimizer.h.

7.15.3.8 const FLOAT fractal::Optimizer::GetMomentum ( ) [inline]

Definition at line 71 of file Optimizer.h.

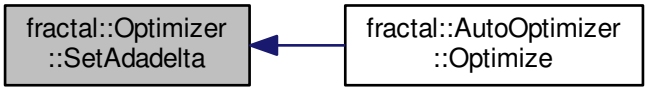
7.15.3.9 const bool fractal::Optimizer::GetRmsprop ( ) [inline]

Definition at line 73 of file Optimizer.h.

7.15.3.10 void fractal::Optimizer::SetAdadelta (const bool *val*) [inline]

Definition at line 67 of file Optimizer.h.

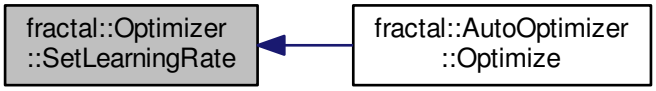
Here is the caller graph for this function:



7.15.3.11 void fractal::Optimizer::SetLearningRate (const FLOAT *val*) [inline]

Definition at line 65 of file Optimizer.h.

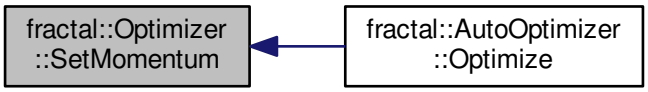
Here is the caller graph for this function:



7.15.3.12 void fractal::Optimizer::SetMomentum (const FLOAT *val*) [inline]

Definition at line 66 of file Optimizer.h.

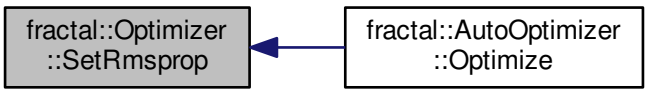
Here is the caller graph for this function:



7.15.3.13 void fractal::Optimizer::SetRmsprop (const bool *val*) [inline]

Definition at line 68 of file Optimizer.h.

Here is the caller graph for this function:



7.15.4 Member Data Documentation

7.15.4.1 bool fractal::Optimizer::adadelta [protected]

Definition at line 90 of file Optimizer.h.

7.15.4.2 FLOAT fractal::Optimizer::learningRate [protected]

Definition at line 81 of file Optimizer.h.

7.15.4.3 FLOAT fractal::Optimizer::momentum [protected]

Definition at line 82 of file Optimizer.h.

7.15.4.4 PEvent fractal::Optimizer::pEventDataTransferToBuf [protected]

Definition at line 87 of file Optimizer.h.

7.15.4.5 PEvent fractal::Optimizer::pEventDataTransferToRnn [protected]

Definition at line 88 of file Optimizer.h.

7.15.4.6 Pipe fractal::Optimizer::pipe[4] [protected]

Definition at line 84 of file Optimizer.h.

7.15.4.7 PStream fractal::Optimizer::pStreamDataTransferToBuf [protected]

Definition at line 85 of file Optimizer.h.

7.15.4.8 PStream fractal::Optimizer::pStreamDataTransferToRnn [protected]

Definition at line 86 of file Optimizer.h.

7.15.4.9 bool fractal::Optimizer::rmsprop [protected]

Definition at line 91 of file Optimizer.h.

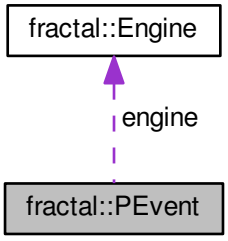
The documentation for this class was generated from the following files:

- src/util/Optimizer.h
- src/util/Optimizer.cc

7.16 fractal::PEvent Class Reference

`#include <Engine.h>`

Collaboration diagram for fractal::PEvent:



Public Member Functions

- PEvent ()

Public Attributes

- unsigned long loc
- cudaEvent_t cudaEvent
- cudaStream_t cudaStream
- Engine * engine

7.16.1 Detailed Description

Definition at line 44 of file Engine.h.

7.16.2 Constructor & Destructor Documentation

7.16.2.1 fractal::PEvent::PEvent ( ) [inline]

Definition at line 47 of file Engine.h.

7.16.3 Member Data Documentation

7.16.3.1 cudaEvent_t fractal::PEvent::cudaEvent

Definition at line 52 of file Engine.h.

7.16.3.2 cudaStream_t fractal::PEvent::cudaStream

Definition at line 53 of file Engine.h.

7.16.3.3 Engine* fractal::PEvent::engine

Definition at line 59 of file Engine.h.

7.16.3.4 unsigned long fractal::PEvent::loc

Definition at line 49 of file Engine.h.

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

- src/core/Engine.h

7.17 fractal::Pipe Class Reference

`#include <Pipe.h>`

Public Member Functions

- Pipe ()
- void Init ()
- void SendSignal ()
- void Wait (const unsigned long count)

Protected Attributes

- unsigned long signalCount
- std::mutex mtx
- std::condition_variable cv

7.17.1 Detailed Description

Definition at line 31 of file Pipe.h.

7.17.2 Constructor & Destructor Documentation

7.17.2.1 fractal::Pipe::Pipe ( )

Definition at line 23 of file Pipe.cc.

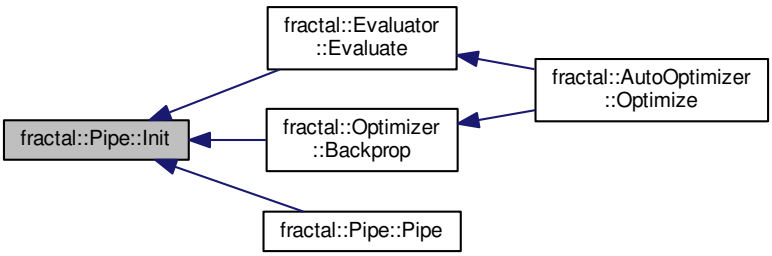
Here is the call graph for this function:

7.17.3 Member Function Documentation

7.17.3.1 void fractal::Pipe::Init ( )

Definition at line 29 of file Pipe.cc.

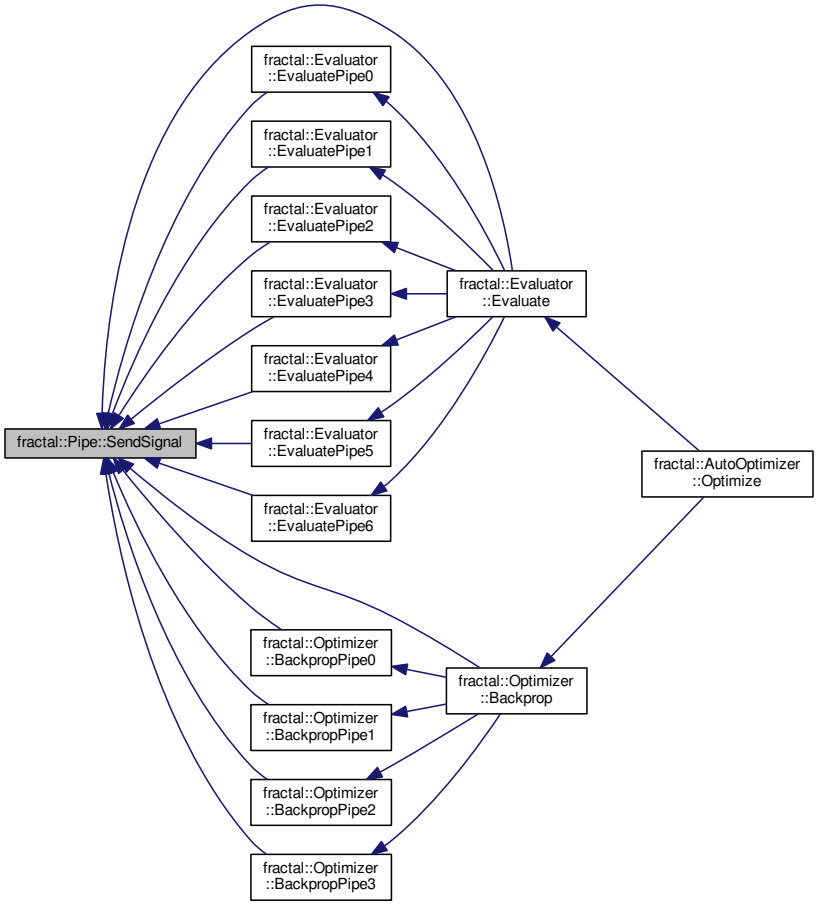
Here is the caller graph for this function:



7.17.3.2 void fractal::Pipe::SendSignal ( )

Definition at line 35 of file Pipe.cc.

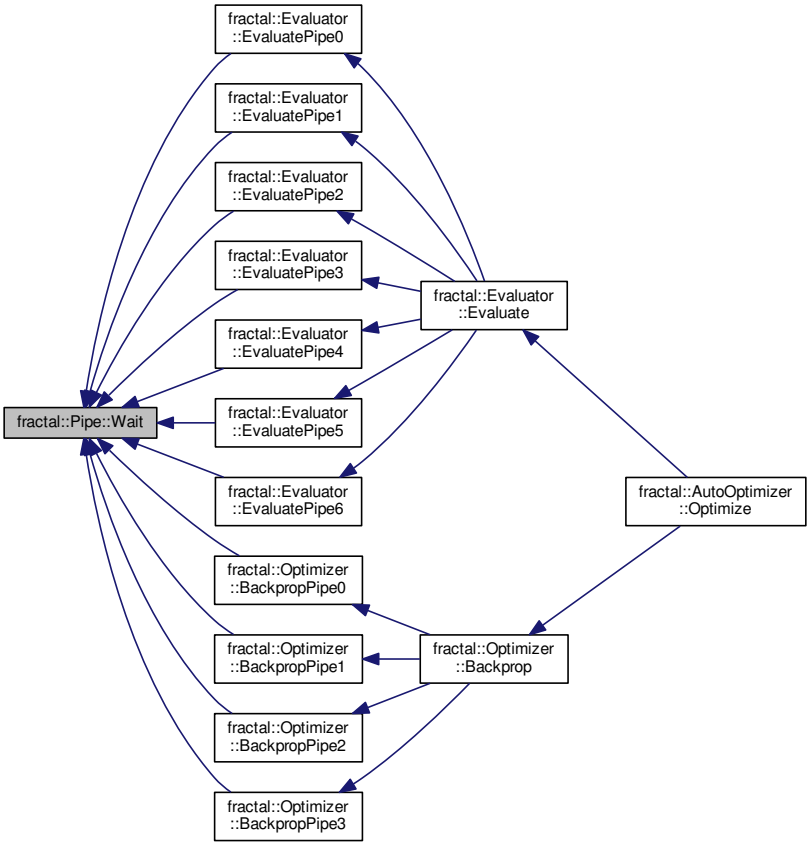
Here is the caller graph for this function:



7.17.3.3 void fractal::Pipe::Wait (const unsigned long *count*)

Definition at line 44 of file Pipe.cc.

Here is the caller graph for this function:



7.17.4 Member Data Documentation

7.17.4.1 std::condition_variable fractal::Pipe::cv [protected]

Definition at line 45 of file Pipe.h.

7.17.4.2 std::mutex fractal::Pipe::mtx [protected]

Definition at line 44 of file Pipe.h.

7.17.4.3 unsigned long fractal::Pipe::signalCount [protected]

Definition at line 42 of file Pipe.h.

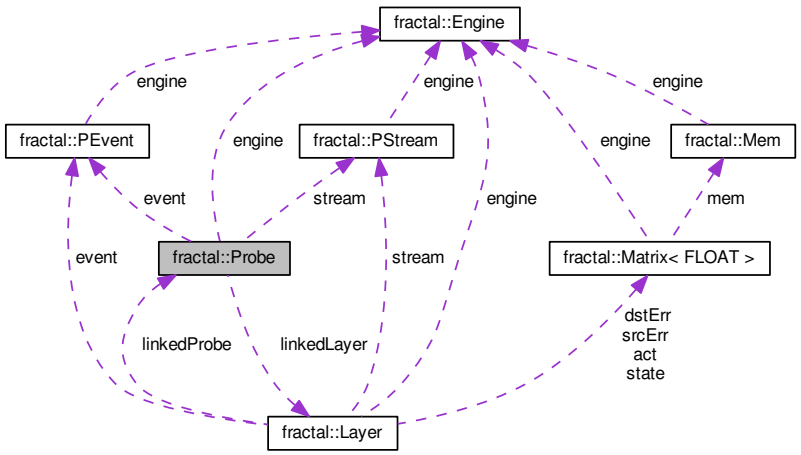
The documentation for this class was generated from the following files:

- src/util/Pipe.h
- src/util/Pipe.cc

7.18 fractal::Probe Class Reference

`#include <Probe.h>`

Collaboration diagram for fractal::Probe:



Public Member Functions

- Probe ()
- virtual ~Probe ()
- void SetEngine (Engine *engine)
- Engine * GetEngine () const
- void SetInput (const bool val)
- void SetOutput (const bool val)
- const bool IsInput () const
- const bool IsOutput () const
- void LinkLayer (Layer *const layer)
- void UnlinkLayer ()
- const bool IsLinked () const
- const unsigned long GetLayerSize () const
- Matrix< FLOAT > & GetActivation ()
- Matrix< FLOAT > & GetState ()
- Matrix< FLOAT > & GetError ()
- PStream & GetPStream ()
- void EventRecord ()
- void EventSynchronize ()
- void StreamWaitEvent (PStream &stream)
- void Wait ()

Protected Attributes

- bool _input
- bool _output
- Engine * engine
- PStream stream
- PEvent event
- Layer * linkedLayer

7.18.1 Detailed Description

Definition at line 32 of file Probe.h.

7.18.2 Constructor & Destructor Documentation

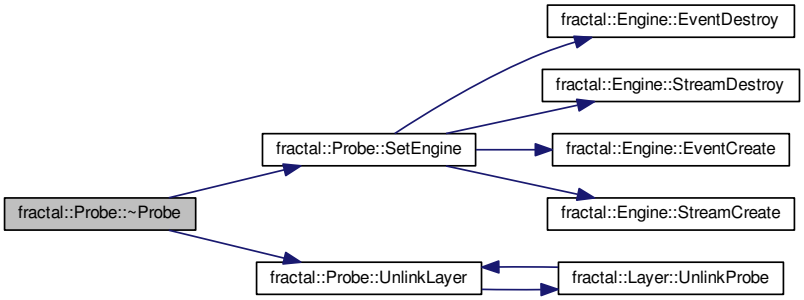
7.18.2.1 fractal::Probe::Probe ( )

Definition at line 26 of file Probe.cc.

7.18.2.2 fractal::Probe::~Probe ( ) [virtual]

Definition at line 35 of file Probe.cc.

Here is the call graph for this function:



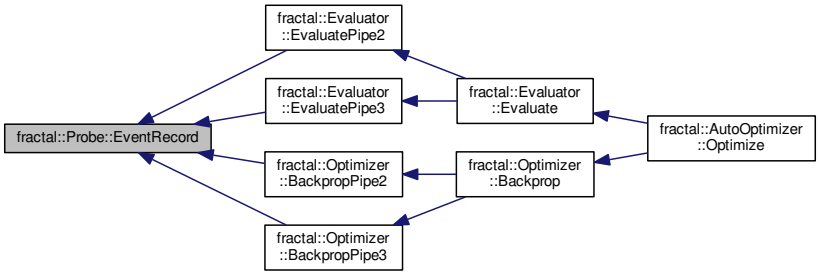
7.18.3 Member Function Documentation

7.18.3.1 void fractal::Probe::EventRecord ( )

Definition at line 143 of file Probe.cc.

Here is the call graph for this function:

Here is the caller graph for this function:



7.18.3.2 void fractal::Probe::EventSynchronize ( )

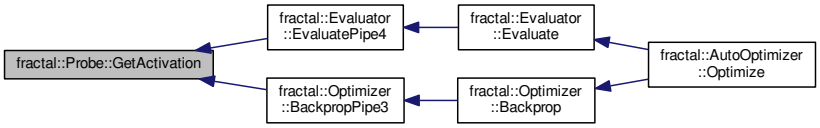
Definition at line 150 of file Probe.cc.

Here is the call graph for this function:

7.18.3.3 Matrix< FLOAT > & fractal::Probe::GetActivation ( )

Definition at line 115 of file Probe.cc.

Here is the caller graph for this function:



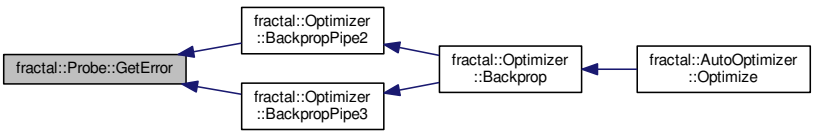
7.18.3.4 Engine * fractal::Probe::GetEngine ( ) const

Definition at line 60 of file Probe.cc.

7.18.3.5 Matrix< FLOAT > & fractal::Probe::GetError ( )

Definition at line 129 of file Probe.cc.

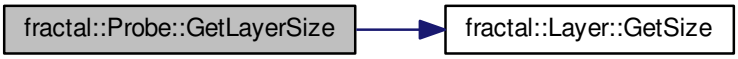
Here is the caller graph for this function:



7.18.3.6 const unsigned long fractal::Probe::GetLayerSize ( ) const

Definition at line 108 of file Probe.cc.

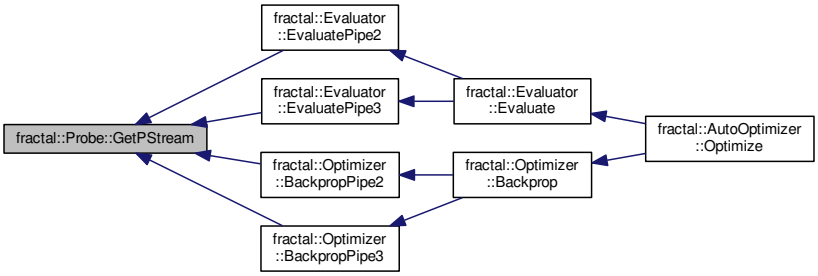
Here is the call graph for this function:



7.18.3.7 PStream & fractal::Probe::GetPStream ( )

Definition at line 136 of file Probe.cc.

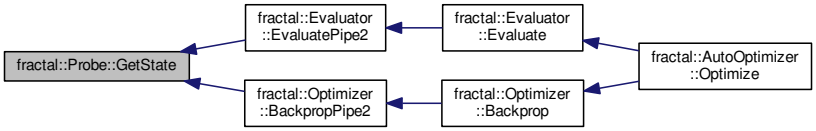
Here is the caller graph for this function:



7.18.3.8 Matrix< FLOAT > & fractal::Probe::GetState ( )

Definition at line 122 of file Probe.cc.

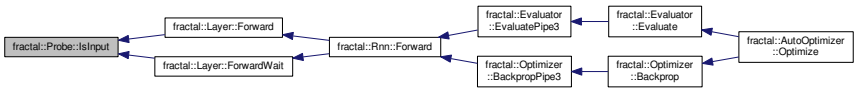
Here is the caller graph for this function:



7.18.3.9 const bool fractal::Probe::IsInput ( ) const [inline]

Definition at line 43 of file Probe.h.

Here is the caller graph for this function:



7.18.3.10 const bool fractal::Probe::IsLinked ( ) const

Definition at line 102 of file Probe.cc.

7.18.3.11 const bool fractal::Probe::IsOutput ( ) const [inline]

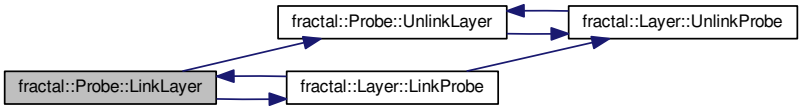
Definition at line 44 of file Probe.h.

Here is the caller graph for this function:

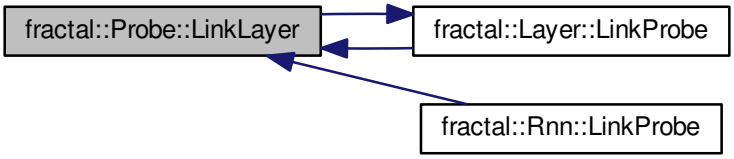
7.18.3.12 void fractal::Probe::LinkLayer (Layer ∗const *layer*)

Definition at line 78 of file Probe.cc.

Here is the call graph for this function:



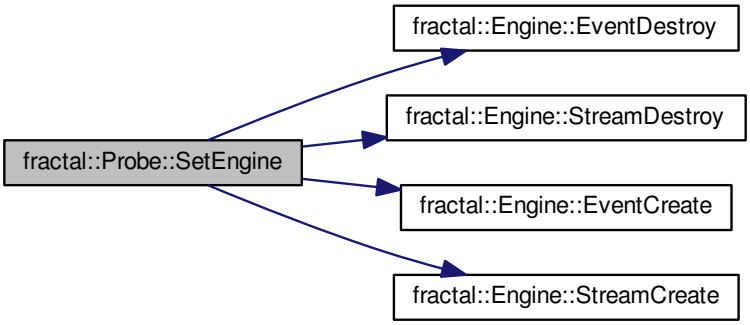
Here is the caller graph for this function:



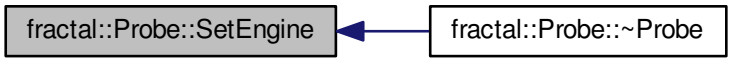
7.18.3.13 void fractal::Probe::SetEngine (Engine * *engine*)

Definition at line 42 of file Probe.cc.

Here is the call graph for this function:



Here is the caller graph for this function:



7.18.3.14 void fractal::Probe::SetInput (const bool *val*)

Definition at line 66 of file Probe.cc.

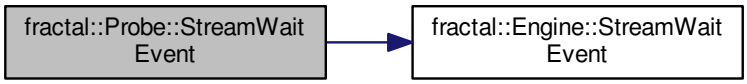
7.18.3.15 void fractal::Probe::SetOutput (const bool *val*)

Definition at line 72 of file Probe.cc.

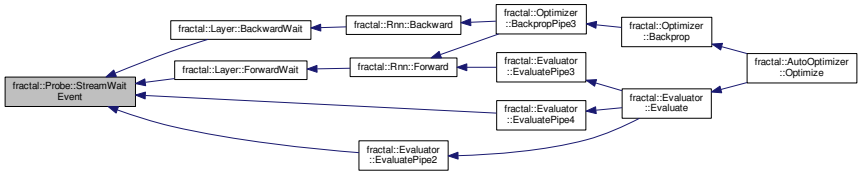
7.18.3.16 void fractal::Probe::StreamWaitEvent (PStream & *stream*)

Definition at line 157 of file Probe.cc.

Here is the call graph for this function:



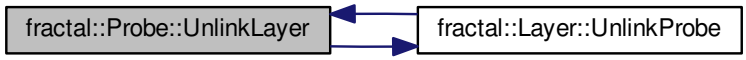
Here is the caller graph for this function:



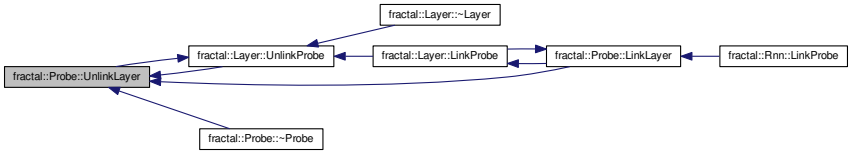
7.18.3.17 void fractal::Probe::UnlinkLayer ( )

Definition at line 89 of file Probe.cc.

Here is the call graph for this function:



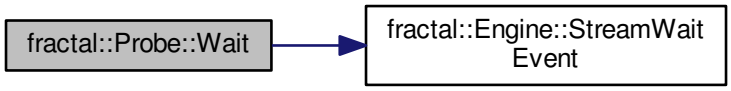
Here is the caller graph for this function:



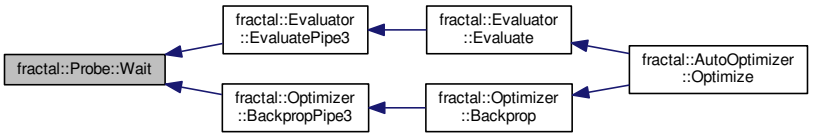
7.18.3.18 void fractal::Probe::Wait ( )

Definition at line 164 of file Probe.cc.

Here is the call graph for this function:



Here is the caller graph for this function:



7.18.4 Member Data Documentation

7.18.4.1 bool fractal::Probe::_input [protected]

Definition at line 65 of file Probe.h.

7.18.4.2 bool fractal::Probe::_output [protected]

Definition at line 65 of file Probe.h.

7.18.4.3 Engine* fractal::Probe::engine [protected]

Definition at line 66 of file Probe.h.

7.18.4.4 PEvent fractal::Probe::event [protected]

Definition at line 68 of file Probe.h.

7.18.4.5 Layer* fractal::Probe::linkedLayer [protected]

Definition at line 69 of file Probe.h.

7.18.4.6 PStream fractal::Probe::stream [protected]

Definition at line 67 of file Probe.h.

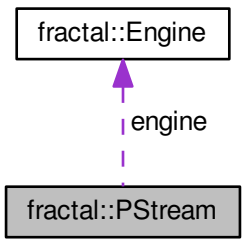
The documentation for this class was generated from the following files:

- src/core/Probe.h
- src/core/Probe.cc

7.19 fractal::PStream Class Reference

`#include <Engine.h>`

Collaboration diagram for fractal::PStream:



Public Member Functions

- PStream ()

Public Attributes

- unsigned long loc
- cudaStream_t cudaStream
- Engine * engine

7.19.1 Detailed Description

Definition at line 63 of file Engine.h.

7.19.2 Constructor & Destructor Documentation

7.19.2.1 fractal::PStream::PStream ( ) [inline]

Definition at line 66 of file Engine.h.

7.19.3 Member Data Documentation

7.19.3.1 cudaStream_t fractal::PStream::cudaStream

Definition at line 71 of file Engine.h.

7.19.3.2 Engine* fractal::PStream::engine

Definition at line 76 of file Engine.h.

7.19.3.3 unsigned long fractal::PStream::loc

Definition at line 68 of file Engine.h.

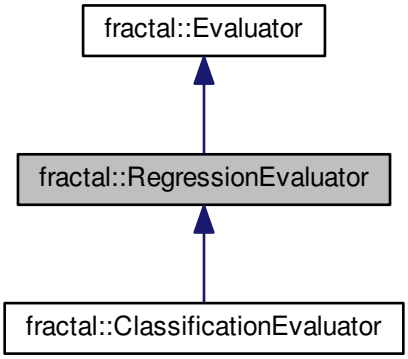
The documentation for this class was generated from the following file:

- src/core/Engine.h

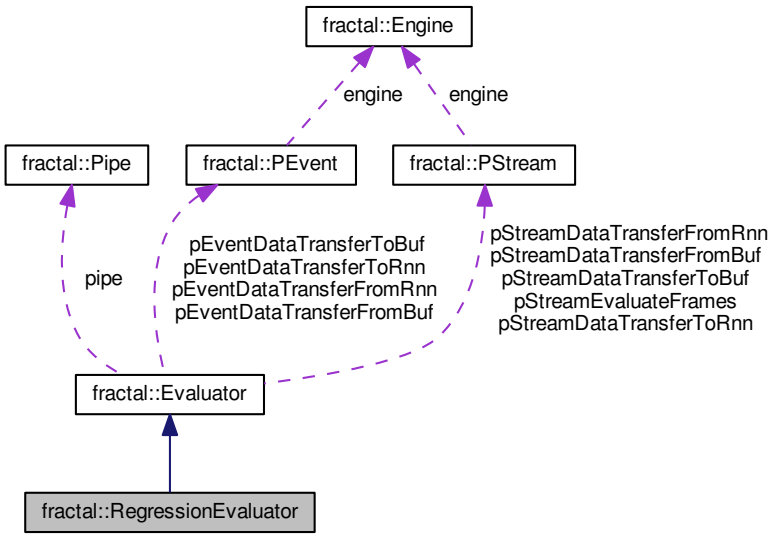
7.20 fractal::RegressionEvaluator Class Reference

`#include <RegressionEvaluator.h>`

Inheritance diagram for fractal::RegressionEvaluator:



Collaboration diagram for fractal::RegressionEvaluator:



Public Member Functions

- RegressionEvaluator ()
- virtual const double GetLoss (const unsigned long outputIdx) const
- const double GetMeanSquaredError (const unsigned long outputIdx) const

Protected Member Functions

- virtual void Reset ()

- virtual void EvaluateFrames (const unsigned long outputIdx, Matrix< FLOAT > &target, Matrix< FLOAT > &output, const unsigned long nStream, PStream &stream)
- virtual void MemAlloc ()

Protected Attributes

- std::vector< unsigned long > nSample
- std::vector< double > seSum

Additional Inherited Members

7.20.1 Detailed Description

Definition at line 28 of file RegressionEvaluator.h.

7.20.2 Constructor & Destructor Documentation

7.20.2.1 fractal::RegressionEvaluator::RegressionEvaluator ( ) [inline]

Definition at line 31 of file RegressionEvaluator.h.

7.20.3 Member Function Documentation

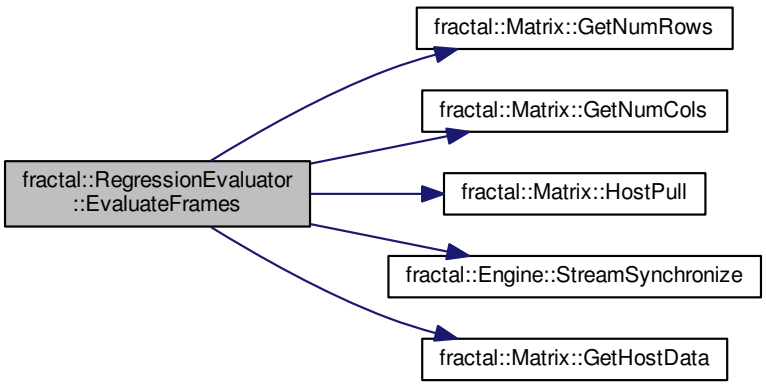
7.20.3.1 void fractal::RegressionEvaluator::EvaluateFrames (const unsigned long *outputIdx*, Matrix< FLOAT > & *target*, Matrix< FLOAT > & *output*, const unsigned long *nStream*, PStream & *stream*) [protected], [virtual]

Implements fractal::Evaluator.

Reimplemented in fractal::ClassificationEvaluator.

Definition at line 48 of file RegressionEvaluator.cc.

Here is the call graph for this function:



7.20.3.2 const double fractal::RegressionEvaluator::GetLoss (const unsigned long *outputIdx*) const [virtual]

Implements fractal::Evaluator.

Reimplemented in fractal::ClassificationEvaluator.

Definition at line 23 of file RegressionEvaluator.cc.

Here is the call graph for this function:

7.20.3.3 const double fractal::RegressionEvaluator::GetMeanSquaredError (const unsigned long *outputIdx*) const

Definition at line 28 of file RegressionEvaluator.cc.

Here is the caller graph for this function:

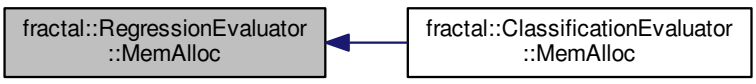
7.20.3.4 void fractal::RegressionEvaluator::MemAlloc ( ) [protected], [virtual]

Implements fractal::Evaluator.

Reimplemented in fractal::ClassificationEvaluator.

Definition at line 89 of file RegressionEvaluator.cc.

Here is the caller graph for this function:



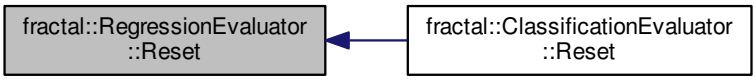
7.20.3.5 void fractal::RegressionEvaluator::Reset ( ) [protected],[virtual]

Implements fractal::Evaluator.

Reimplemented in fractal::ClassificationEvaluator.

Definition at line 36 of file RegressionEvaluator.cc.

Here is the caller graph for this function:



7.20.4 Member Data Documentation

7.20.4.1 std::vector<unsigned long> fractal::RegressionEvaluator::nSample [protected]

Definition at line 43 of file RegressionEvaluator.h.

7.20.4.2 std::vector<double> fractal::RegressionEvaluator::seSum [protected]

Definition at line 44 of file RegressionEvaluator.h.

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

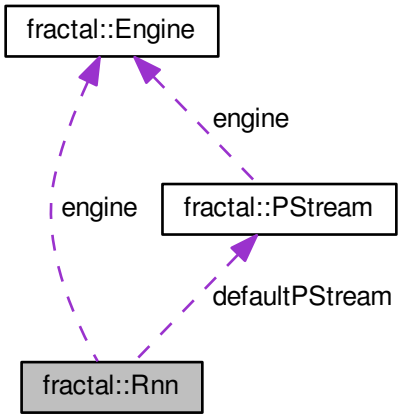
- src/util/RegressionEvaluator.h
- src/util/RegressionEvaluator.cc

7.21 fractal::Rnn Class Reference

A network structure container.

`#include <Rnn.h>`

Collaboration diagram for fractal::Rnn:



Public Member Functions

- Rnn ()

  The constructor.
- virtual ~Rnn ()

  The destructor.
- void SetEngine (Engine *engine)

  Set a computation engine.
- Engine * GetEngine () const

  Get the computation engine.
- void AddLayer (const std::string &name, ActType actType, StateType stateType, const unsigned long size, const LayerParam &param=LayerParam())

  Add a layer.
- void AddConnection (const std::string &from, const std::string &to, const unsigned long delayAmount, const bool isIdentity, const InitWeightParam &initWeightParam=InitWeightParam())

  Add a connection.
- void DeleteLayer (const std::string &name)

  Delete a Layer.
- void DeleteConnection (const std::string &from, const std::string &to)

  Delete a connection.
- void LinkProbe (Probe &probe, const std::string &layerName)

  Link a probe to a layer.
- void SetBatchSize (const unsigned long batchSize)

  Set the mini-batch size.
- const unsigned long GetBatchSize () const

  Get the mini-batch size.
- void InitForward (const unsigned long batchFrom, const unsigned long batchTo)

  Initialize layer activations for forward propagation.
- void InitBackward (const unsigned long batchFrom, const unsigned long batchTo)

  Initialize errors in connections for backward propagation.

- void InitWeights (const InitWeightParam &param)

  Initialize all weights.
- void InitAdadelta (const FLOAT decayRate)

  Initialize AdaDelta.
- void InitNesterov ()

  Initialize Nesterov momentum.
- void InitRmsprop (const FLOAT decayRate)

  Initialize RMSprop.
- void Forward (const unsigned long batchFrom, const unsigned long batchTo, const unsigned long nStream)

  Forward propagation.
- void Backward (const unsigned long batchFrom, const unsigned long batchTo, const unsigned long nStream)

  Backward propagation.
- void CalcActDeriv (const unsigned long batchFrom, const unsigned long batchTo)

  Compute the derivatives of the activation functions with respect to the layer states.
- void UpdateWeights (const unsigned long batchFrom, const unsigned long batchTo, const unsigned long nFrame, const FLOAT rate, const FLOAT momentum, const bool adadelta, const bool rmsprop)

  Update the weights.
- void Synchronize ()

  Wait until all asynchronous operations are finished.
- void StreamWait (PStream &stream)

  Force a stream to wait until all asynchronous operations of this RNN are finished.
- void Ready ()

  Get ready to perform forward and backward propagation.
- void Clear ()

  Free resources.
- void SaveState (const std::string &path)

  Save the current network states.
- void LoadState (const std::string &path)

  Load the previously saved network states.
- const unsigned long GetNumWeights ()

  Get the total number of weights.

Protected Types

- typedef std::list< Layer * > Scc

  Structure for strongly connected components.
- typedef std::list< Layer * > LayerList

  List structure of Layer objects.
- typedef std::unordered_map < std::string, Layer * > LayerMap

  Map structure that maps each layer name to the corresponding Layer object.
- typedef std::unordered_set < Connection * > ConnSet

  Set structure that contains the pointers to all connections.
- typedef std::list< Scc * > SccList

  List structure of Scc objects.
- typedef std::list< PStream * > PStreamList

  List structure of PStream objects.

Protected Member Functions

- Layer * FindLayer (const std::string &layerName)

  Find a layer using its name.
- void AddConnection (Layer *const from, Layer *const to, const unsigned long delayAmount, const bool isIdentity, const InitWeightParam &initWeightParam)

  Add a connection.
- void DeleteConnection (Layer *const from, Layer *const to)

  Delete a connection.
- void LinkProbe (Probe &probe, Layer *const layer)

  Link a probe to a layer.
- void ClearLayers ()

  Clear all layers.
- void ClearConnections ()

  Clear all connections.
- void ClearSccList ()

  Clear all SCCs.
- void ClearPStreams ()

  Clear all PStreams.
- void Tarjan ()

  Perform Tarjan's strongly connected component (SCC) algorithm.
- Scc *const CreateScc (std::stack< Layer * > &sccStack, const Layer *const root, const long group)

  Create an SCC and perform topological sort.
- void CreatePStreams (const unsigned long loc)

  Create a PStream object with the location loc.
- void CreateDefaultPStream (const unsigned long loc)

  Create the default PStream object with the location loc.
- void DestroyDefaultPStream ()

  Destroy the default PStream object.

Protected Attributes

- Engine * engine

  Engine pointer.
- LayerMap layerMap

  Layer map.
- ConnSet connSet

  Connection set.
- SccList sccList

  Scc list.
- PStreamList pStreamList

  PStream list.
- PStream * defaultPStream

  Default PStream.
- unsigned long batchSize

  Mini-batch size.
- bool isReady

  Indicates whether the network is analyzed or not.

7.21.1 Detailed Description

A network structure container.

This class contains a neural network structure with graph-based representation. Nodes and edges correspond to layers and connections respectively.

Definition at line 44 of file Rnn.h.

7.21.2 Member Typedef Documentation

7.21.2.1 typedef std::unordered_set<Connection *> fractal::Rnn::ConnSet [protected]

Set structure that contains the pointers to all connections.

Definition at line 234 of file Rnn.h.

7.21.2.2 typedef std::list<Layer *> fractal::Rnn::LayerList [protected]

List structure of Layer objects.

Definition at line 228 of file Rnn.h.

7.21.2.3 typedef std::unordered_map<std::string, Layer *> fractal::Rnn::LayerMap [protected]

Map structure that maps each layer name to the corresponding Layer object.

Definition at line 231 of file Rnn.h.

7.21.2.4 typedef std::list<PStream *> fractal::Rnn::PStreamList [protected]

List structure of PStream objects.

Definition at line 240 of file Rnn.h.

7.21.2.5 typedef std::list<Layer *> fractal::Rnn::Scc [protected]

Structure for strongly connected components.

Definition at line 225 of file Rnn.h.

7.21.2.6 typedef std::list<Scc *> fractal::Rnn::SccList [protected]

List structure of Scc objects.

Definition at line 237 of file Rnn.h.

7.21.3 Constructor & Destructor Documentation

7.21.3.1 fractal::Rnn::Rnn ( )

The constructor.

Initialize variables.

Definition at line 39 of file Rnn.cc.

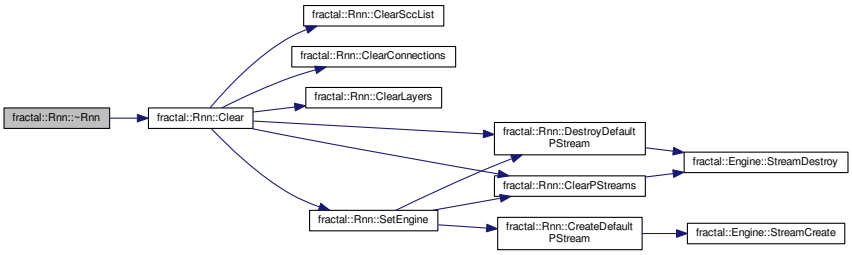
7.21.3.2 fractal::Rnn::~Rnn ( ) [virtual]

The destructor.

Free resources.

Definition at line 48 of file Rnn.cc.

Here is the call graph for this function:



7.21.4 Member Function Documentation

7.21.4.1 void fractal::Rnn::AddConnection (const std::string & *from,* const std::string & *to,* const unsigned long *delayAmount,* const bool *isIdentity,* const InitWeightParam & *initWeightParam* = InitWeightParam ( ))

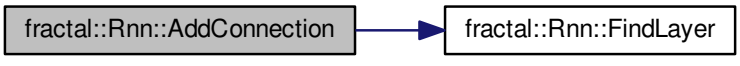
Add a connection.

Parameters

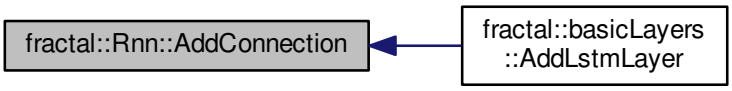
from	The name of the anterior layer.
to	The name of the posterior layer.
delayAmount	Delay amount.
isIdentity	If set to true, an identity matrix is used as the weight matrix.
initWeightParam	Weight initialization parameter.

Definition at line 104 of file Rnn.cc.

Here is the call graph for this function:



Here is the caller graph for this function:



7.21.4.2 void fractal::Rnn::AddConnection (Layer *const *from, Layer *const *to, const unsigned long *delayAmount, const bool *isIdentity, const InitWeightParam & *initWeightParam*) [protected]

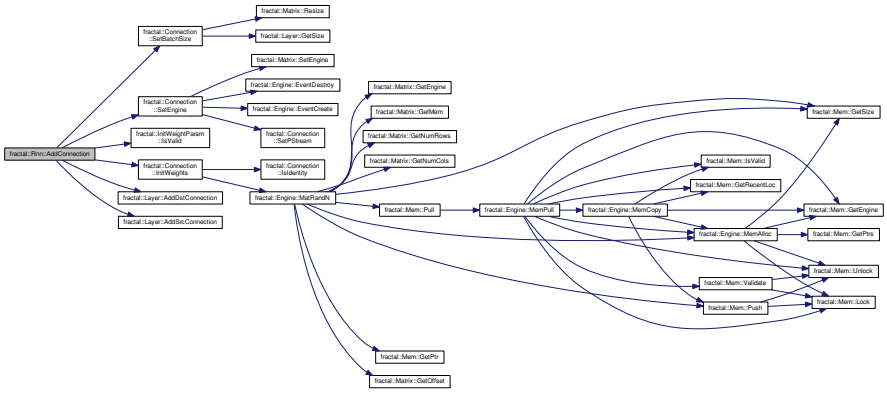
Add a connection.

Parameters

from	The pointer to the anterior layer.
to	The pointer to the posterior layer.
delayAmount	Delay amount.
isIdentity	If set to true, an identity matrix is used as the weight matrix.
initWeightParam	Weight initialization parameter.

Definition at line 1173 of file Rnn.cc.

Here is the call graph for this function:



7.21.4.3 void fractal::Rnn::AddLayer (const std::string & *name, ActType *actType, StateType *stateType, const unsigned long *size, const LayerParam & *param* = LayerParam())

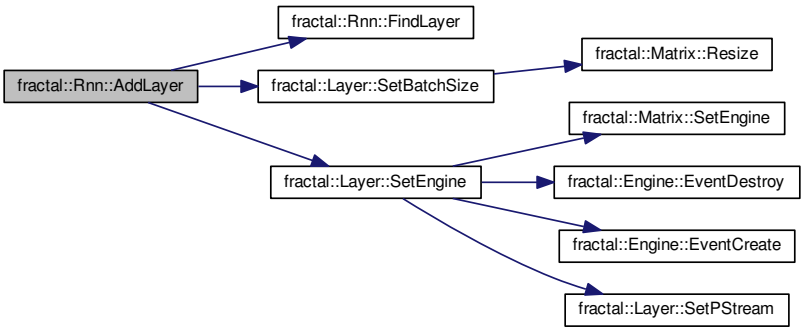
Add a layer.

Parameters

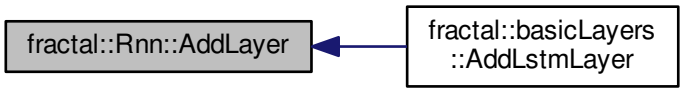
name	Layer name.
actType	Activation function type.
stateType	Aggregation function type.
size	Layer size.
param	Extra layer parameter.

Definition at line 89 of file Rnn.cc.

Here is the call graph for this function:



Here is the caller graph for this function:



7.21.4.4 void fractal::Rnn::Backward (const unsigned long *batchFrom*, const unsigned long *batchTo*, const unsigned long *nStream*)

Backward propagation.

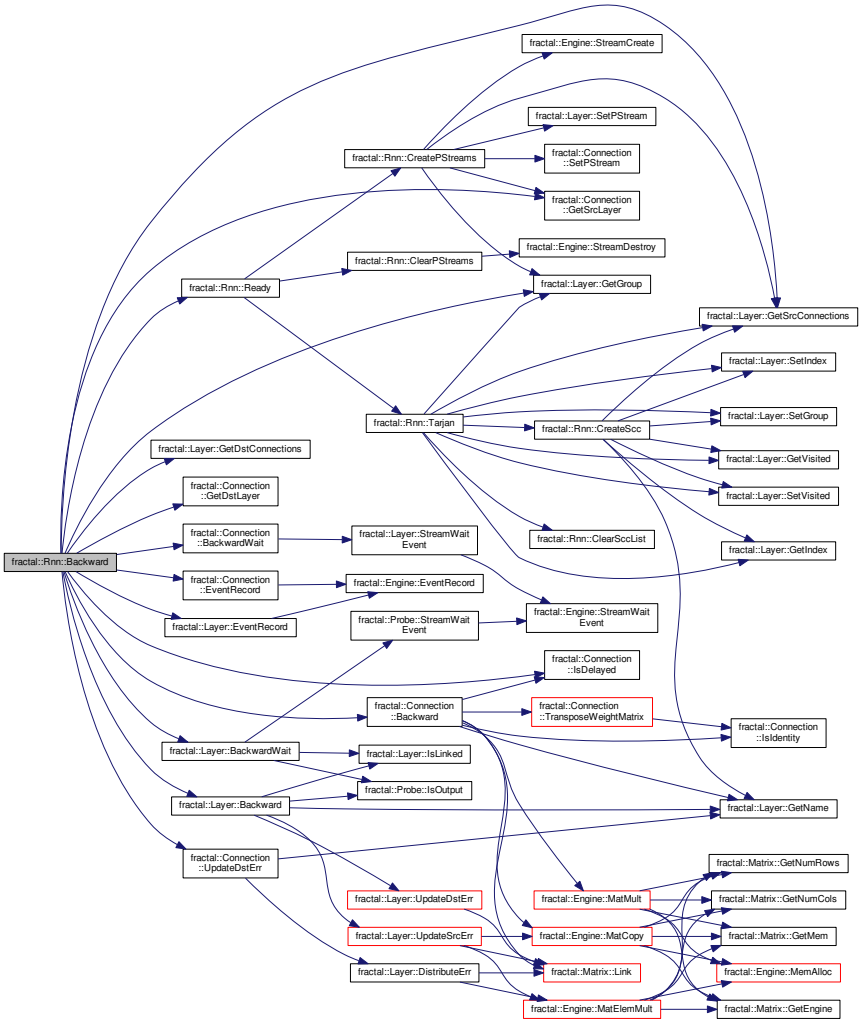
Perform backward propagation from *batchTo* to *batchFrom* with the stride of *nStream*. The total number of backward steps per data stream is (*batchTo* - *batchFrom* + 1) / *nStream*.

Parameters

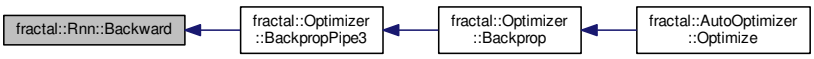
batchFrom	The start index of the mini-batch to perform the operation.
batchTo	The end index of the mini-batch to perform the operation.
nStream	The number of data streams.

Definition at line 448 of file Rnn.cc.

Here is the call graph for this function:



Here is the caller graph for this function:



7.21.4.5 void fractal::Rnn::CalcActDeriv (const unsigned long *batchFrom*, const unsigned long *batchTo*)

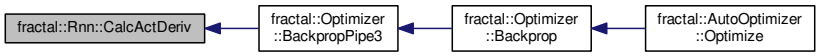
Compute the derivatives of the activation functions with respect to the layer states.

Parameters

batchFrom	The start index of the mini-batch to perform the operation.
batchTo	The end index of the mini-batch to perform the operation.

Definition at line 603 of file Rnn.cc.

Here is the caller graph for this function:

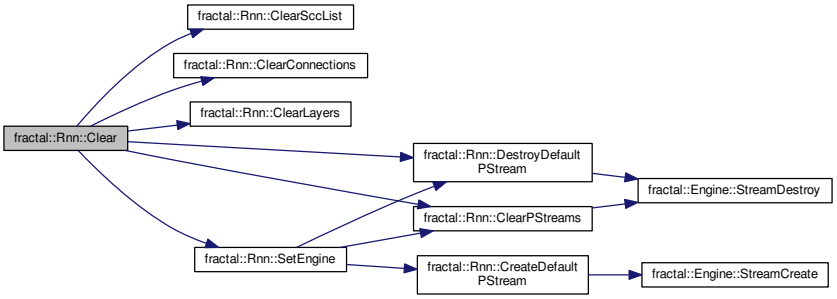


7.21.4.6 void fractal::Rnn::Clear ( )

Free resources.

Definition at line 1148 of file Rnn.cc.

Here is the call graph for this function:



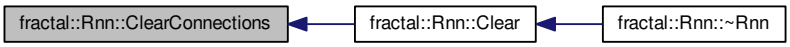
Here is the caller graph for this function:

7.21.4.7 void fractal::Rnn::ClearConnections ( ) [protected]

Clear all connections.

Definition at line 1238 of file Rnn.cc.

Here is the caller graph for this function:

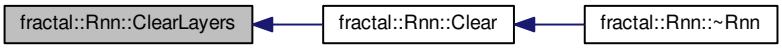


7.21.4.8 void fractal::Rnn::ClearLayers ( ) [protected]

Clear all layers.

Definition at line 1224 of file Rnn.cc.

Here is the caller graph for this function:



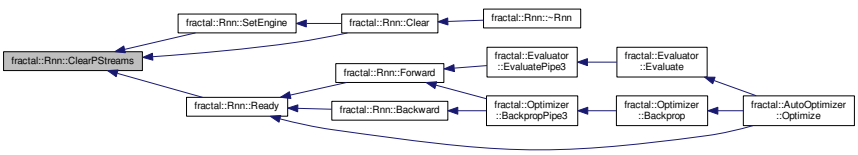
7.21.4.9 void fractal::Rnn::ClearPStreams ( ) [protected]

Clear all PStreams.

Definition at line 1266 of file Rnn.cc.

Here is the call graph for this function:

Here is the caller graph for this function:

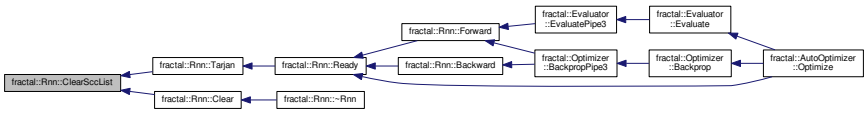


7.21.4.10 void fractal::Rnn::ClearSccList ( ) [protected]

Clear all SCCs.

Definition at line 1252 of file Rnn.cc.

Here is the caller graph for this function:



7.21.4.11 void fractal::Rnn::CreateDefaultPStream (const unsigned long *loc*) [protected]

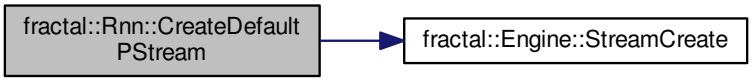
Create the default PStream object with the location *loc*.

Parameters

loc	Location.

Definition at line 1090 of file Rnn.cc.

Here is the call graph for this function:



Here is the caller graph for this function:

7.21.4.12 void fractal::Rnn::CreatePStreams (const unsigned long *loc*) [protected]

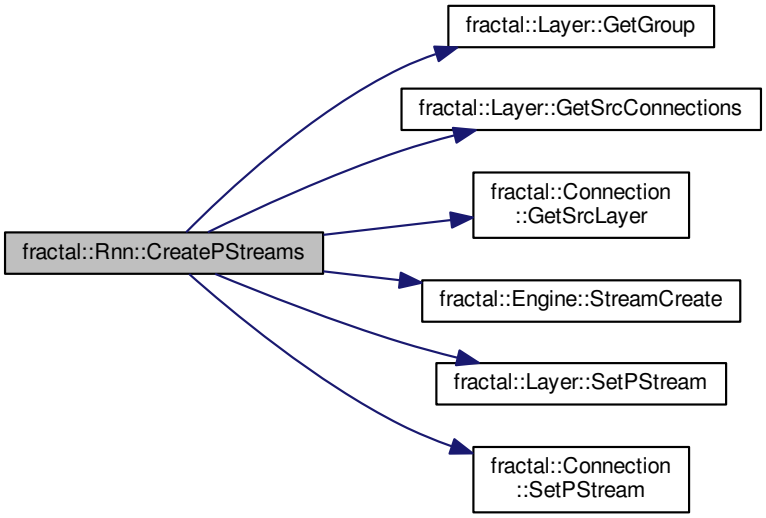
Create a PStream object with the location *loc*.

Parameters

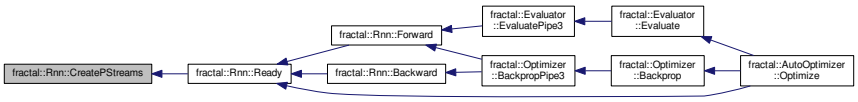
loc	Location.

Definition at line 975 of file Rnn.cc.

Here is the call graph for this function:



Here is the caller graph for this function:



7.21.4.13 Rnn::Scc *const fractal::Rnn::CreateScc (std::stack< Layer * > & *sccStack*, const Layer *const *root*, const long *group*) [protected]

Create an SCC and perform topological sort.

Subroutine of Tarjan().

Parameters

sccStack	Stack of nodes (Layer objects).
root	The pointer to the root node (Layer object).
group	The group index or SCC index.

Returns

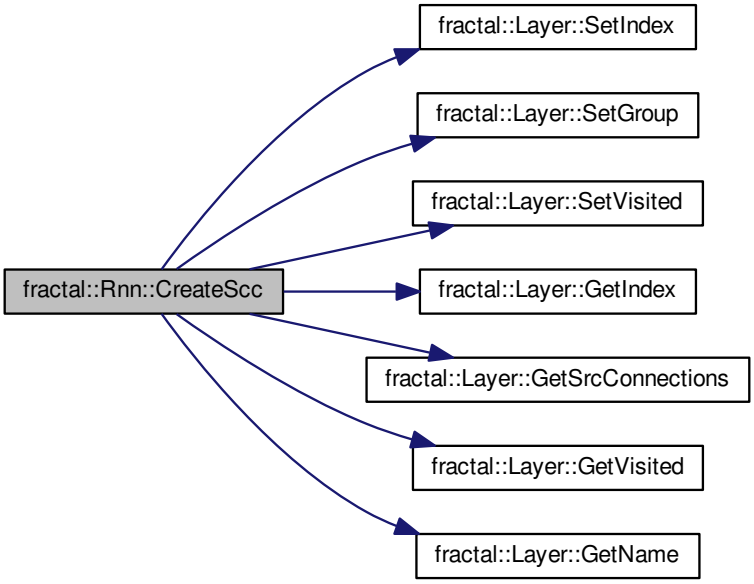
The pointer to the newly created SCC object.

Note

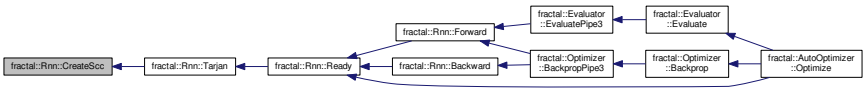
Memory space is dynamically allocated for the returned SCC object.

Definition at line 755 of file Rnn.cc.

Here is the call graph for this function:



Here is the caller graph for this function:



7.21.4.14 void fractal::Rnn::DeleteConnection (const std::string & *from*, const std::string & *to*)

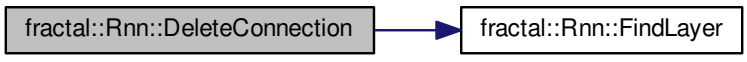
Delete a connection.

Parameters

from	The name of the anterior layer of the connection to be deleted.
to	The name of the posterior layer of the connection to be deleted.

Definition at line 158 of file Rnn.cc.

Here is the call graph for this function:



7.21.4.15 void fractal::Rnn::DeleteConnection (Layer *const *from*, Layer *const *to*) [protected]

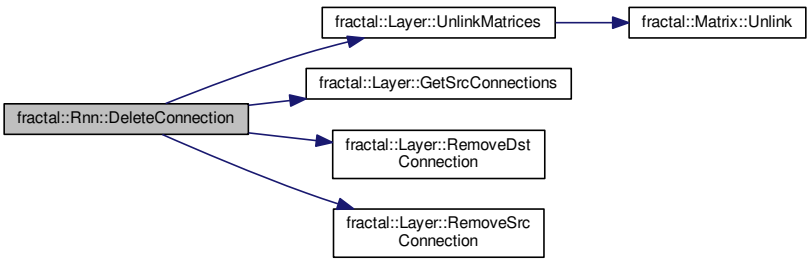
Delete a connection.

Parameters

from	The pointer to the anterior layer of the connection to be deleted.
to	The pointer to the posterior layer of the connection to be deleted.

Definition at line 1192 of file Rnn.cc.

Here is the call graph for this function:



7.21.4.16 void fractal::Rnn::DeleteLayer (const std::string & *name*)

Delete a Layer.

All connections from or to this layer is automatically removed.

Parameters

name	The name of the layer to be deleted.

Definition at line 120 of file Rnn.cc.

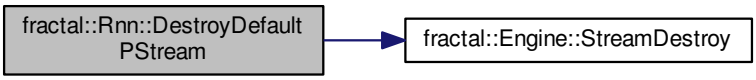
Here is the call graph for this function:

7.21.4.17 void fractal::Rnn::DestroyDefaultPStream ( ) [protected]

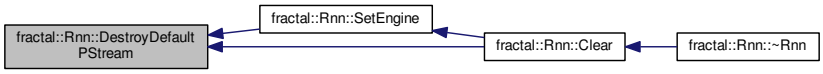
Destroy the default PStream object.

Definition at line 1099 of file Rnn.cc.

Here is the call graph for this function:



Here is the caller graph for this function:



7.21.4.18 Layer * fractal::Rnn::FindLayer (const std::string & *layerName*) [protected]

Find a layer using its name.

Parameters

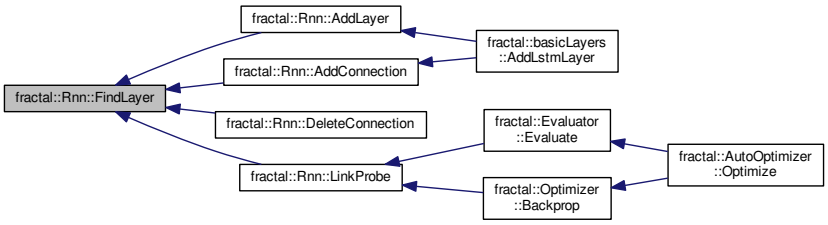
layerName	Layer name.

Returns

The pointer to the corresponding layer. If there is no matching layer, NULL is returned.

Definition at line 1162 of file Rnn.cc.

Here is the caller graph for this function:



7.21.4.19 void fractal::Rnn::Forward (const unsigned long *batchFrom,* const unsigned long *batchTo,* const unsigned long *nStream*)

Forward propagation.

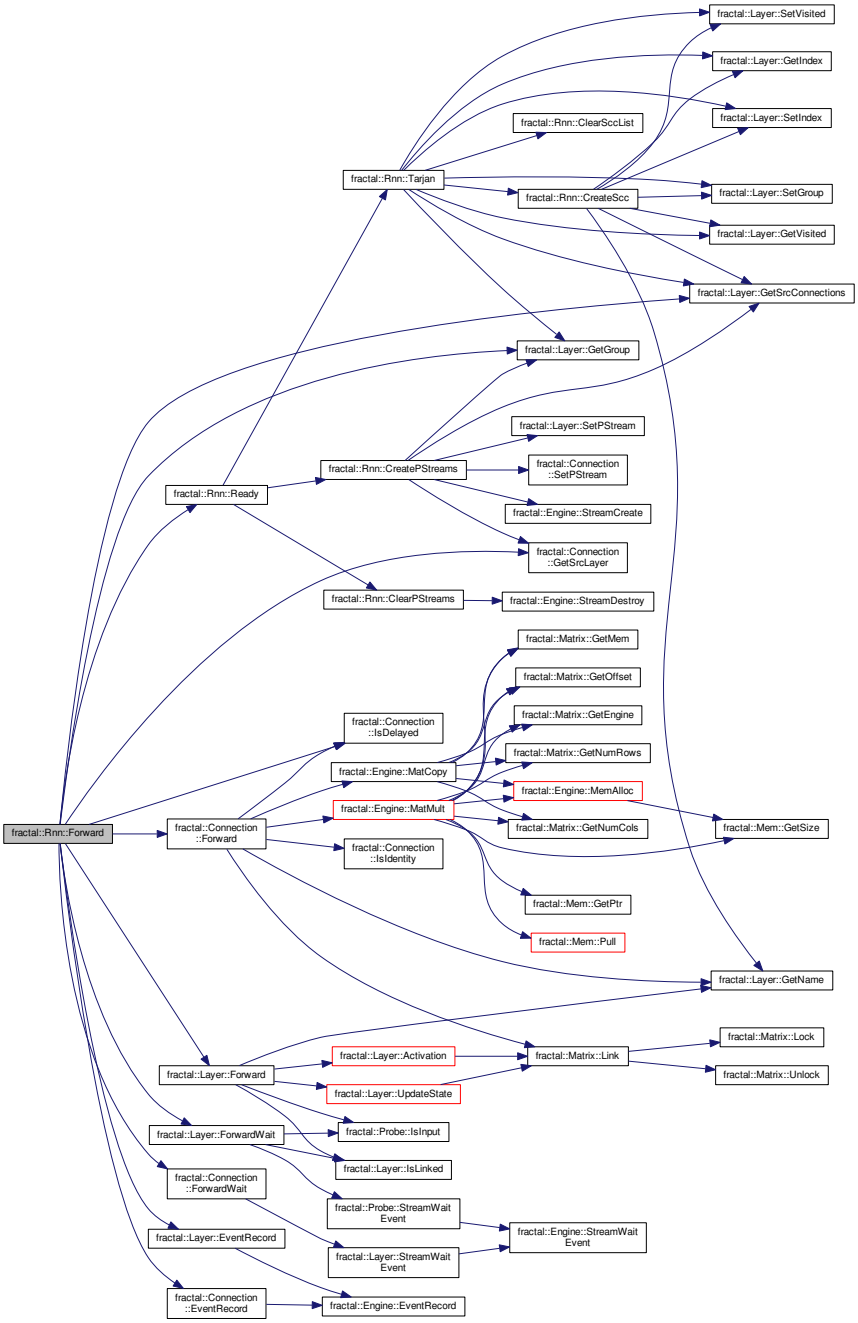
Perform forward propagation from *batchFrom* to *batchTo* with the stride of *nStream*. The total number of forward steps per data stream is (*batchTo* - *batchFrom* + 1) / *nStream*.

Parameters

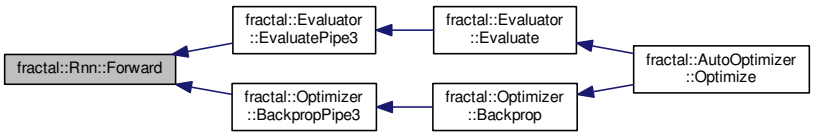
batchFrom	The start index of the mini-batch to perform the operation.
batchTo	The end index of the mini-batch to perform the operation.
nStream	The number of data streams.

Definition at line 308 of file Rnn.cc.

Here is the call graph for this function:



Here is the caller graph for this function:



7.21.4.20 const unsigned long fractal::Rnn::GetBatchSize ( ) const

Get the mini-batch size.

Returns

Mini-batch size.

Definition at line 210 of file Rnn.cc.

7.21.4.21 Engine * fractal::Rnn::GetEngine ( ) const

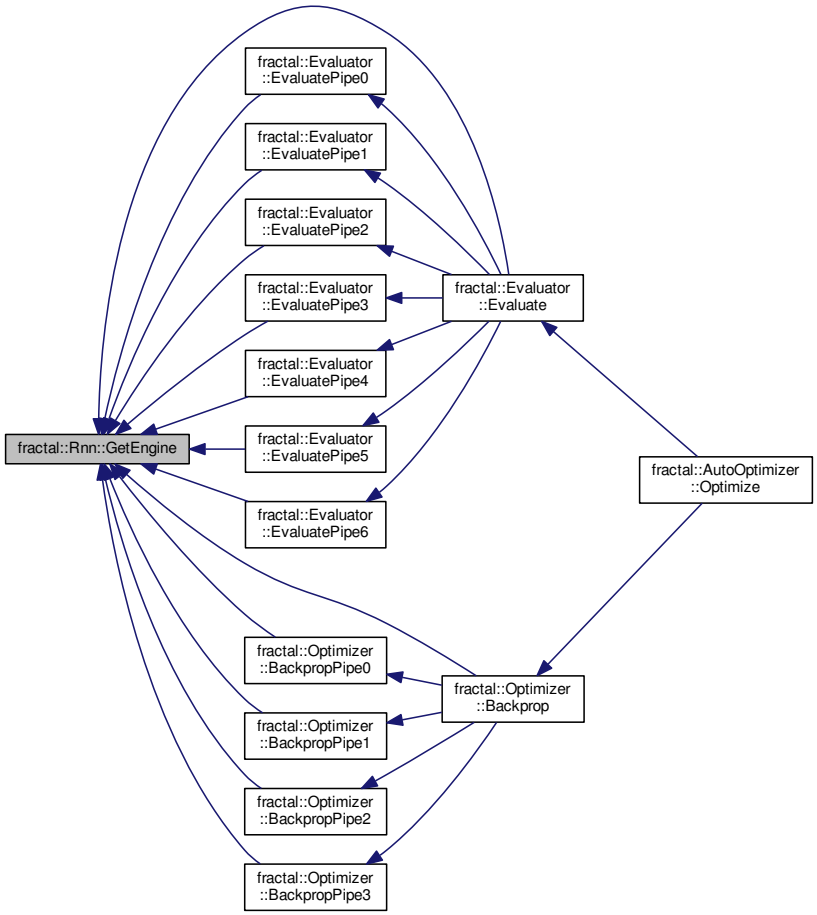
Get the computation engine.

Returns

A pointer to the Engine instance of this RNN. The return value is NULL when the engine is not set.

Definition at line 83 of file Rnn.cc.

Here is the caller graph for this function:



7.21.4.22 const unsigned long fractal::Rnn::GetNumWeights ( )

Get the total number of weights.

Definition at line 1341 of file Rnn.cc.

7.21.4.23 void fractal::Rnn::InitAdadelta (const FLOAT *decayRate*)

Initialize AdaDelta.

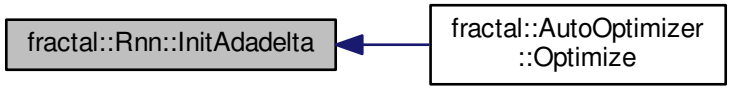
Initialize mean-square variables.

Parameters

decayRate	Decay rate of exponential averaging.

Definition at line 280 of file Rnn.cc.

Here is the caller graph for this function:



7.21.4.24 void fractal::Rnn::InitBackward (const unsigned long *batchFrom,* const unsigned long *batchTo*)

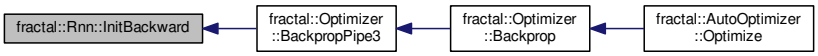
Initialize errors in connections for backward propagation.

Parameters

batchFrom	The start index of the mini-batch to perform the operation.
batchTo	The end index of the mini-batch to perform the operation.

Definition at line 234 of file Rnn.cc.

Here is the caller graph for this function:



7.21.4.25 void fractal::Rnn::InitForward (const unsigned long *batchFrom,* const unsigned long *batchTo*)

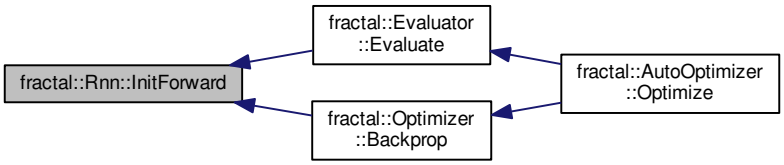
Initialize layer activations for forward propagation.

Parameters

batchFrom	The start index of the mini-batch to perform the operation.
batchTo	The end index of the mini-batch to perform the operation.

Definition at line 216 of file Rnn.cc.

Here is the caller graph for this function:



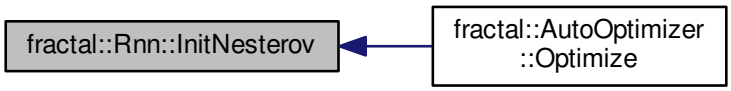
7.21.4.26 void fractal::Rnn::InitNesterov ( )

Initialize Nesterov momentum.

Initialize the velocities to zero.

Definition at line 268 of file Rnn.cc.

Here is the caller graph for this function:



7.21.4.27 void fractal::Rnn::InitRmsprop (const FLOAT *decayRate*)

Initialize RMSprop.

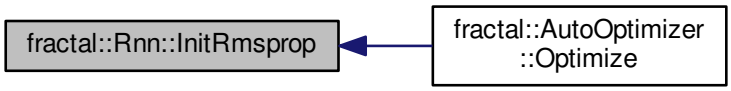
Initialize the mean-square variables to one.

Parameters

decayRate	Decay rate of exponential averaging.

Definition at line 294 of file Rnn.cc.

Here is the caller graph for this function:



7.21.4.28 void fractal::Rnn::InitWeights (const InitWeightParam & *param*)

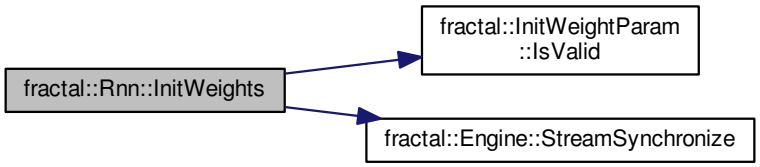
Initialize all weights.

Parameters

param	Weight initialization parameter.

Definition at line 252 of file Rnn.cc.

Here is the call graph for this function:



7.21.4.29 void fractal::Rnn::LinkProbe (Probe & *probe,* const std::string & *layerName*)

Link a probe to a layer.

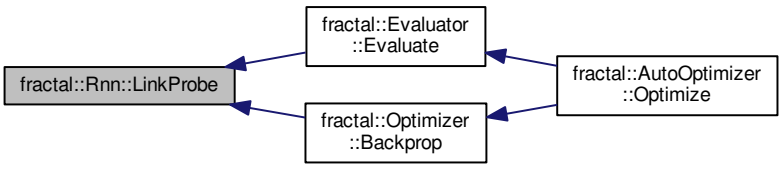
Parameters

probe	The probe instance.
layerName	Layer name.

Definition at line 174 of file Rnn.cc.

Here is the call graph for this function:

Here is the caller graph for this function:



7.21.4.30 void fractal::Rnn::LinkProbe (Probe & *probe,* Layer *const *layer*) [protected]

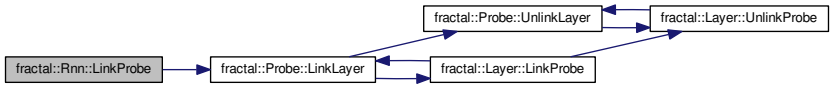
Link a probe to a layer.

Parameters

probe	The probe instance.
layer	The pointer to the layer.

Definition at line 1218 of file Rnn.cc.

Here is the call graph for this function:



7.21.4.31 void fractal::Rnn::LoadState (const std::string & *path*)

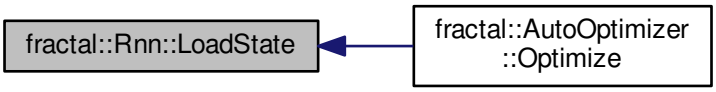
Load the previously saved network states.

Parameters

path	Path to load.

Definition at line 1313 of file Rnn.cc.

Here is the caller graph for this function:



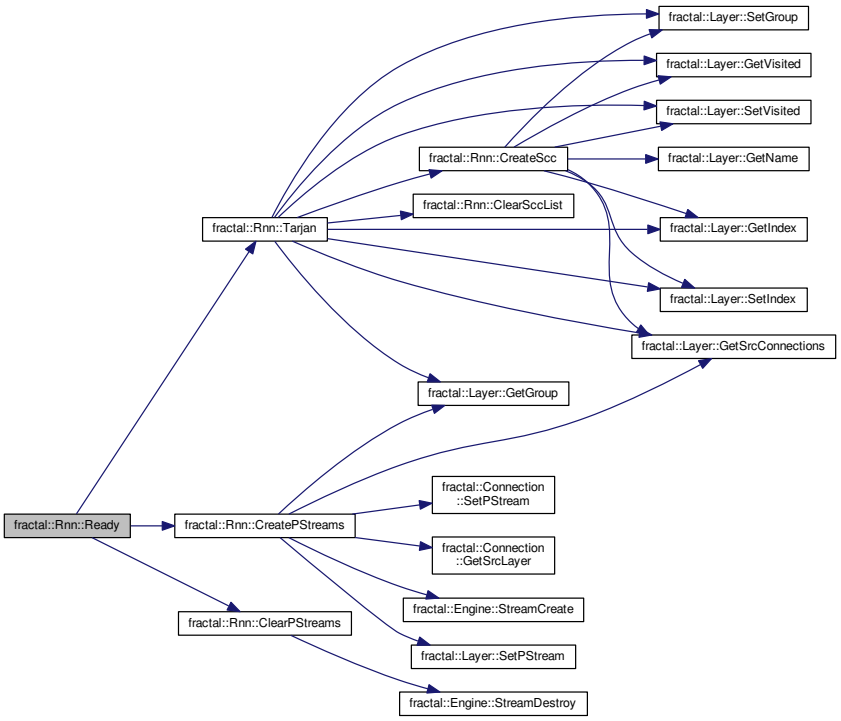
7.21.4.32 void fractal::Rnn::Ready ( )

Get ready to perform forward and backward propagation.

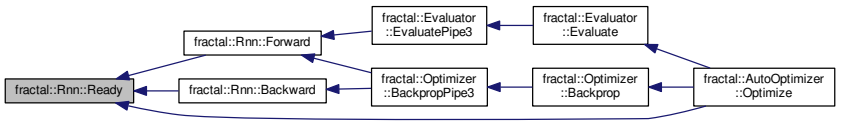
Analyze the network structure to find strongly connected components (loops) and determine the activation order. Automatically called when needed.

Definition at line 632 of file Rnn.cc.

Here is the call graph for this function:



Here is the caller graph for this function:



7.21.4.33 void fractal::Rnn::SaveState (const std::string & *path*)

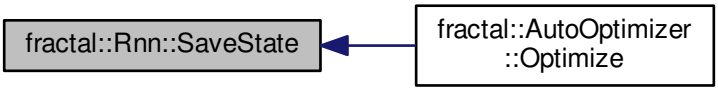
Save the current network states.

Parameters

path	Path to save.

Definition at line 1281 of file Rnn.cc.

Here is the caller graph for this function:



7.21.4.34 void fractal::Rnn::SetBatchSize (const unsigned long *batchSize*)

Set the mini-batch size.

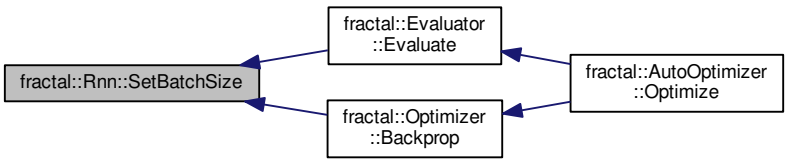
Unroll the network *batchSize* times with a circular buffer.

Parameters

batchSize	Mini-batch size.

Definition at line 186 of file Rnn.cc.

Here is the caller graph for this function:



7.21.4.35 void fractal::Rnn::SetEngine (Engine * *engine*)

Set a computation engine.

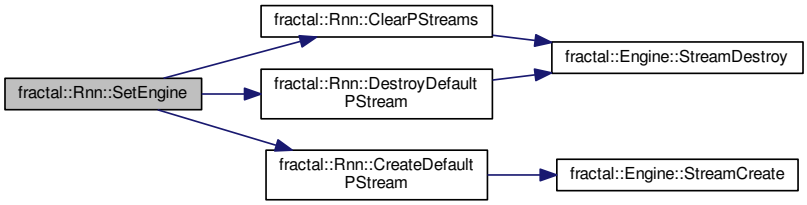
If *engine* is set to NULL, free resources.

Parameters

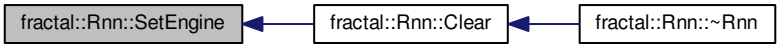
engine	A pointer to an Engine instance.

Definition at line 54 of file Rnn.cc.

Here is the call graph for this function:



Here is the caller graph for this function:



7.21.4.36 void fractal::Rnn::StreamWait (PStream & *stream*)

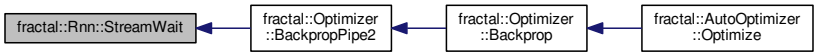
Force a *stream* to wait until all asynchronous operations of this RNN are finished.

Parameters

stream	PStream object.

Definition at line 1124 of file Rnn.cc.

Here is the caller graph for this function:



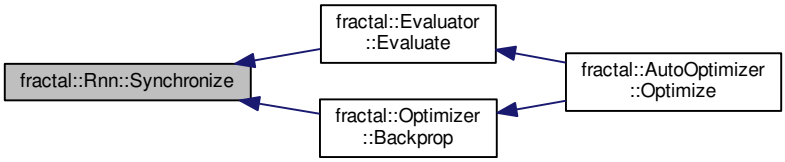
7.21.4.37 void fractal::Rnn::Synchronize ( )

Wait until all asynchronous operations are finished.

Definition at line 1110 of file Rnn.cc.

Here is the call graph for this function:

Here is the caller graph for this function:



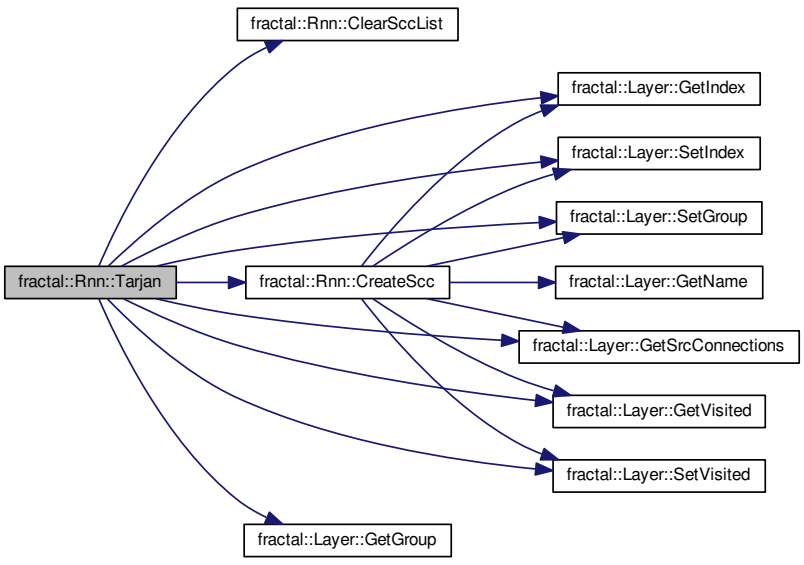
7.21.4.38 void fractal::Rnn::Tarjan ( ) [protected]

Perform Tarjan's strongly connected component (SCC) algorithm.

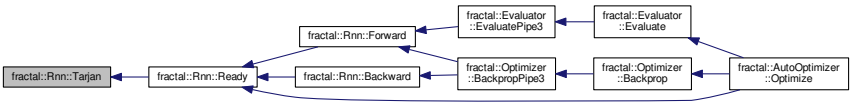
This function analyzes the network structure to find SCCs and loops.

Definition at line 665 of file Rnn.cc.

Here is the call graph for this function:



Here is the caller graph for this function:



7.21.4.39 void fractal::Rnn::UpdateWeights (const unsigned long *batchFrom,* const unsigned long *batchTo,* const unsigned long *nFrame,* const **FLOAT** *rate,* const **FLOAT** *momentum,* const bool *adadelta,* const bool *rmsprop*)

Update the weights.

Parameters

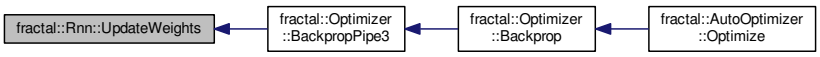
batchFrom	The start index of the mini-batch to perform the operation.
batchTo	The end index of the mini-batch to perform the operation.
nFrame	The total number of forward frames.
rate	Learning rate.
momentum	Momentum.
adadelta	Set to true to perform AdaDelta.
rmsprop	Set to true to perform RMSprop.

Note

AdaDelta and *RMSprop* cannot be set to true at the same time.

Definition at line 616 of file Rnn.cc.

Here is the caller graph for this function:



7.21.5 Member Data Documentation

7.21.5.1 unsigned long fractal::Rnn::batchSize [protected]

Mini-batch size.

The network is unrolled *batchSize* times.

Definition at line 331 of file Rnn.h.

7.21.5.2 ConnSet fractal::Rnn::connSet [protected]

Connection set.

Definition at line 317 of file Rnn.h.

7.21.5.3 PStream* fractal::Rnn::defaultPStream [protected]

Default PStream.

Definition at line 326 of file Rnn.h.

7.21.5.4 Engine* fractal::Rnn::engine [protected]

Engine pointer.

Definition at line 311 of file Rnn.h.

7.21.5.5 bool fractal::Rnn::isReady [protected]

Indicates whether the network is analyzed or not.

Definition at line 334 of file Rnn.h.

7.21.5.6 LayerMap fractal::Rnn::layerMap [protected]

Layer map.

Definition at line 314 of file Rnn.h.

7.21.5.7 PStreamList fractal::Rnn::pStreamList [protected]

PStream list.

Definition at line 323 of file Rnn.h.

7.21.5.8 SccList fractal::Rnn::sccList [protected]

Scc list.

Definition at line 320 of file Rnn.h.

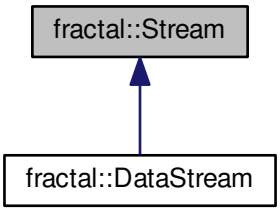
The documentation for this class was generated from the following files:

- src/core/Rnn.h
- src/core/Rnn.cc

7.22 fractal::Stream Class Reference

`#include <Stream.h>`

Inheritance diagram for fractal::Stream:



Public Member Functions

- virtual void SetNumStream (const unsigned long nStream)=0
- virtual const unsigned long GetNumStream () const =0
- virtual const unsigned long GetNumChannel () const =0
- virtual const unsigned long GetDimension (const unsigned long channelIdx) const =0
- virtual void Reset ()=0
- virtual void Next (const unsigned long streamIdx)=0
- virtual void GenerateFrame (const unsigned long streamIdx, const unsigned long channelIdx, FLOAT *const frame)=0

7.22.1 Detailed Description

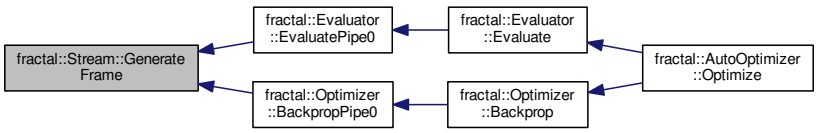
Definition at line 30 of file Stream.h.

7.22.2 Member Function Documentation

7.22.2.1 virtual void fractal::Stream::GenerateFrame (const unsigned long *streamIdx*, const unsigned long *channelIdx*, FLOAT *const *frame*) [pure virtual]

Implemented in fractal::DataStream.

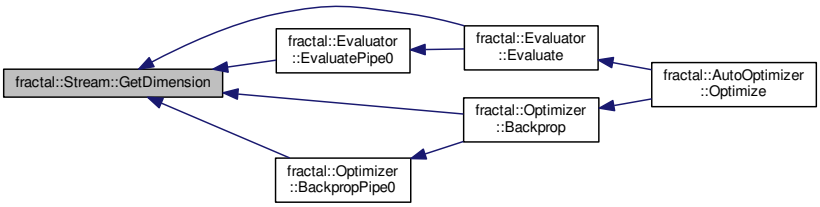
Here is the caller graph for this function:



7.22.2.2 virtual const unsigned long fractal::Stream::GetDimension (const unsigned long *channelIdx*) const [pure virtual]

Implemented in fractal::DataStream.

Here is the caller graph for this function:



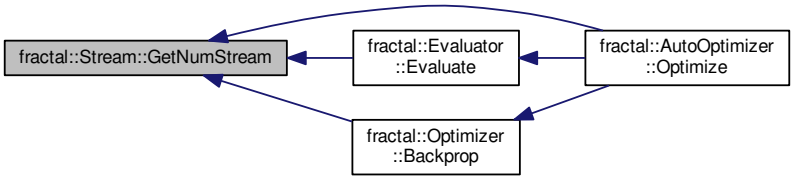
7.22.2.3 virtual const unsigned long fractal::Stream::GetNumChannel ( ) const [pure virtual]

Implemented in fractal::DataStream.

7.22.2.4 virtual const unsigned long fractal::Stream::GetNumStream ( ) const [pure virtual]

Implemented in fractal::DataStream.

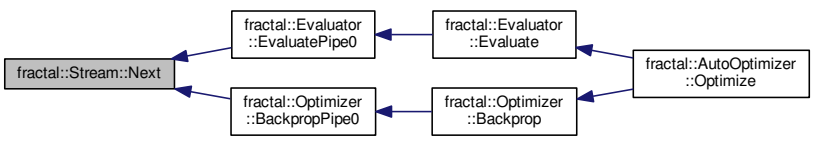
Here is the caller graph for this function:



7.22.2.5 virtual void fractal::Stream::Next (const unsigned long *streamIdx*) [pure virtual]

Implemented in fractal::DataStream.

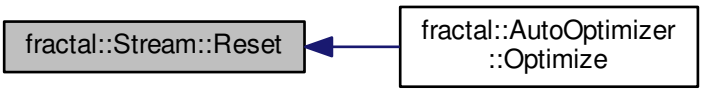
Here is the caller graph for this function:



7.22.2.6 virtual void fractal::Stream::Reset ( ) [pure virtual]

Implemented in fractal::DataStream.

Here is the caller graph for this function:



7.22.2.7 virtual void fractal::Stream::SetNumStream (const unsigned long *nStream*) [pure virtual]

Implemented in fractal::DataStream.

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

- src/util/Stream.h

Chapter 8

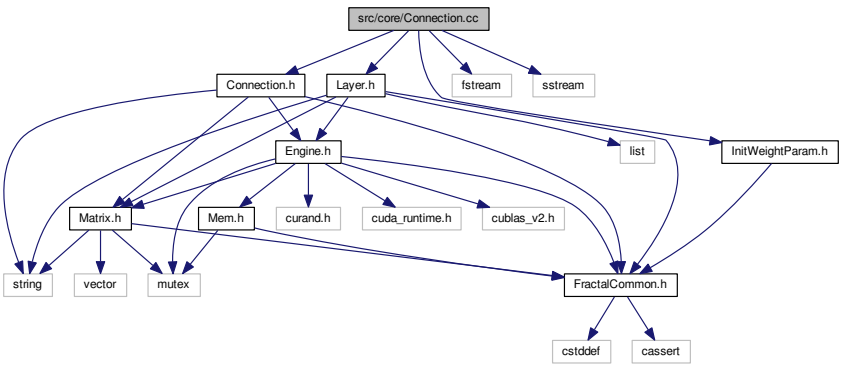
File Documentation

8.1 doc/Mainpage.h File Reference

8.2 src/core/Connection.cc File Reference

```
#include "Connection.h"
#include <fstream>
#include <sstream>
#include "Layer.h"
#include "InitWeightParam.h"
```

Include dependency graph for Connection.cc:



Namespaces

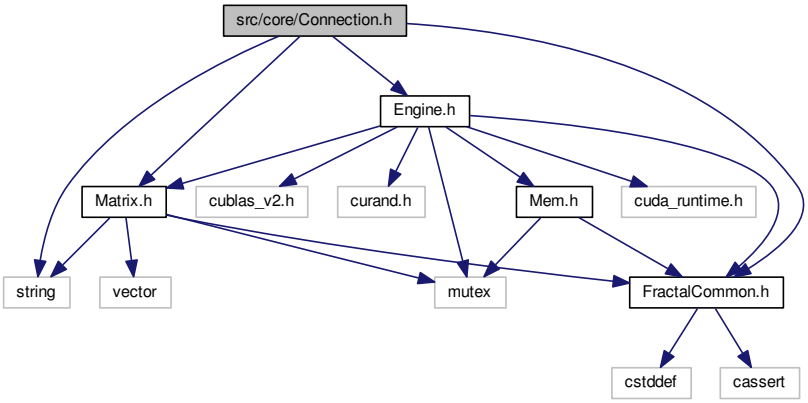
- fractal

  The topmost namespace of libfractal.

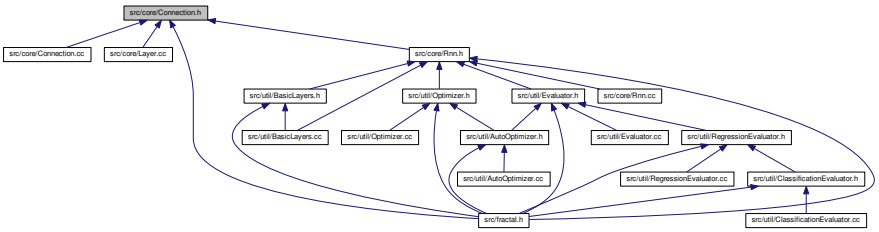
8.3 src/core/Connection.h File Reference

```
#include <string>
#include "Engine.h"
#include "Matrix.h"
#include "FractalCommon.h"
```

Include dependency graph for Connection.h:



This graph shows which files directly or indirectly include this file:



Classes

- class fractal::Connection

Namespaces

- fractal

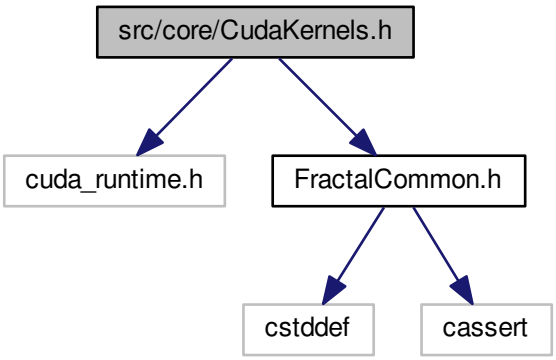
  The topmost namespace of libfractal.

8.4 src/core/CudaKernels.cu File Reference

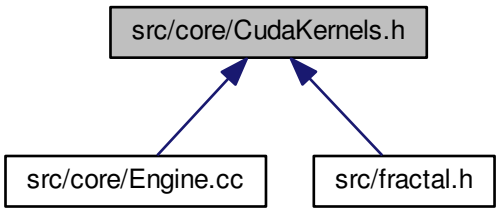
8.5 src/core/CudaKernels.h File Reference

```
#include <cuda_runtime.h>
#include "FractalCommon.h"
```

Include dependency graph for CudaKernels.h:



This graph shows which files directly or indirectly include this file:



Namespaces

- fractal

  The topmost namespace of libfractal.
- fractal::cudaKernels

Functions

- template<class T >
  void fractal::cudaKernels::MemSet (T *_x, const T val, const unsigned long n, const cudaStream_t stream)
- template<class T >
  void fractal::cudaKernels::ElemMult (const T *_x, const T *_y, T *_z, const unsigned long n, const cuda↩
  Stream_t stream)
- template<class T >
  void fractal::cudaKernels::Add (const T *_x, const T *_y, T *_z, const unsigned long n, const cudaStream_t stream)

- template<class T >
  void fractal::cudaKernels::FuncSigmoid (const T *_x, T *_y, const unsigned long n, const cudaStream_t stream)

- template<class T >
  void fractal::cudaKernels::FuncTanh (const T *_x, T *_y, const unsigned long n, const cudaStream_t stream)

- template<class T >
  void fractal::cudaKernels::FuncSoftplus (const T *_x, T *_y, const unsigned long n, const cudaStream_t stream)

- template<class T >
  void fractal::cudaKernels::FuncRectLinear (const T *_x, T *_y, const unsigned long n, const cudaStream_t stream)

- template<class T >
  void fractal::cudaKernels::FuncSoftmax (const T *_x, T *_y, const unsigned long layerSize, const unsigned long batchSize, const cudaStream_t stream)

- template<class T >
  void fractal::cudaKernels::FuncBoundRange (const T *_x, T *_y, const T min, const T max, const unsigned long n, const cudaStream_t stream)

- template<class T >
  void fractal::cudaKernels::FuncSigmoidDeriv (const T *_x, T *_y, const unsigned long n, const cudaStream_t stream)

- template<class T >
  void fractal::cudaKernels::FuncTanhDeriv (const T *_x, T *_y, const unsigned long n, const cudaStream_t stream)

- template<class T >
  void fractal::cudaKernels::FuncSoftplusDeriv (const T *_x, T *_y, const unsigned long n, const cudaStream_t stream)

- template<class T >
  void fractal::cudaKernels::FuncRectLinearDeriv (const T *_x, T *_y, const unsigned long n, const cudaStream_t stream)

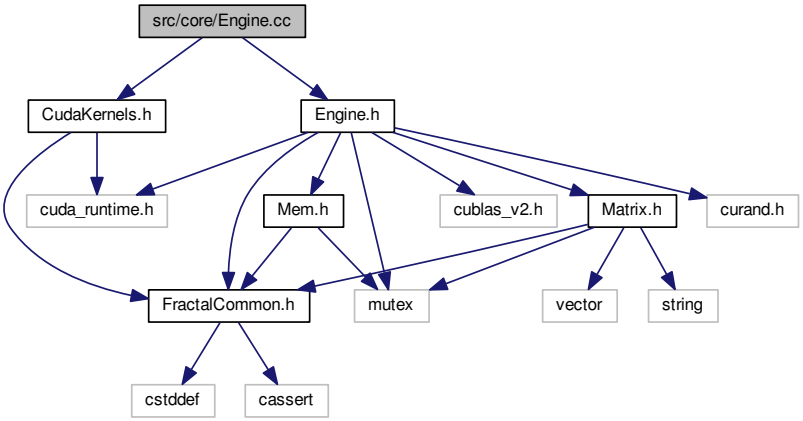
- template<class T >
  void fractal::cudaKernels::Rmsprop (T *_newDerivs, const T *_derivs, T *_msDeriv, const T decayRate, const unsigned long n, const cudaStream_t stream)

- template<class T >
  void fractal::cudaKernels::Adadelta (T *_deltas, const T *_derivs, T *_msDeriv, T *_msDelta, const T learningRate, const T decayRate, const unsigned long n, const cudaStream_t stream)

8.6 src/core/Engine.cc File Reference

```
#include "Engine.h"
#include "CudaKernels.h"
```

Include dependency graph for Engine.cc:



Namespaces

- fractal

  The topmost namespace of libfractal.

Macros

- #define CUDA_CHUNK_SIZE (2 ∗ sizeof(FLOAT)) /∗ In bytes. curandGenerateNormal requires even number of elements ∗/
- #define GEAM cublasSgeam
- #define GEMV cublasSgemv
- #define GEMM cublasSgemm
- #define AXPY cublasSaxpy
- #define COPY cublasScopy
- #define RANDN curandGenerateNormal

8.6.1 Macro Definition Documentation

8.6.1.1 #define AXPY cublasSaxpy

Definition at line 37 of file Engine.cc.

8.6.1.2 #define COPY cublasScopy

Definition at line 38 of file Engine.cc.

8.6.1.3 #define CUDA_CHUNK_SIZE (2 ∗ sizeof(FLOAT)) /∗ In bytes. curandGenerateNormal requires even number of elements ∗/

Definition at line 24 of file Engine.cc.

8.6.1.4 #define GEAM cublasSgeam

Definition at line 34 of file Engine.cc.

8.6.1.5 #define GEMM cublasSgemm

Definition at line 36 of file Engine.cc.

8.6.1.6 #define GEMV cublasSgemv

Definition at line 35 of file Engine.cc.

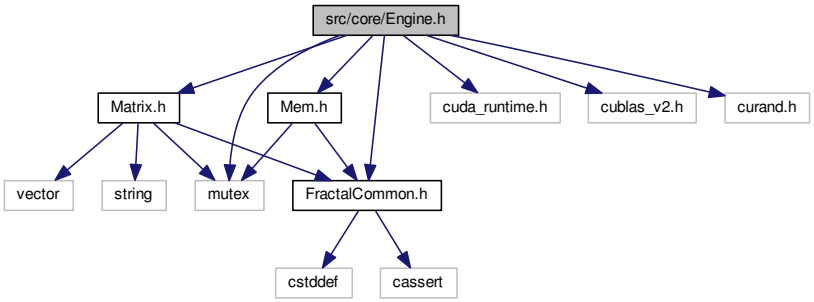
8.6.1.7 #define RANDN curandGenerateNormal

Definition at line 39 of file Engine.cc.

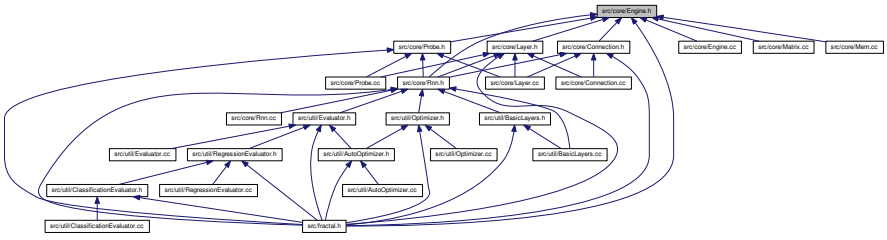
8.7 src/core/Engine.h File Reference

```
#include <mutex>
#include <cuda_runtime.h>
#include <cublas_v2.h>
#include <curand.h>
#include "Matrix.h"
#include "Mem.h"
#include "FractalCommon.h"
```

Include dependency graph for Engine.h:



This graph shows which files directly or indirectly include this file:



Classes

- class fractal::PEvent
- class fractal::PStream
- class fractal::Engine

Namespaces

- fractal

  The topmost namespace of libfractal.

Macros

- #define FRACTAL_USE_CUDA /∗ For now, always use CUDA ∗/

8.7.1 Macro Definition Documentation

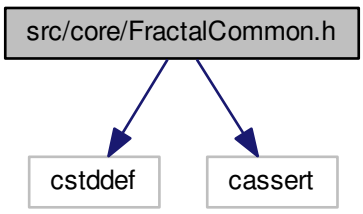
8.7.1.1 #define FRACTAL_USE_CUDA /∗ For now, always use CUDA ∗/

Definition at line 21 of file Engine.h.

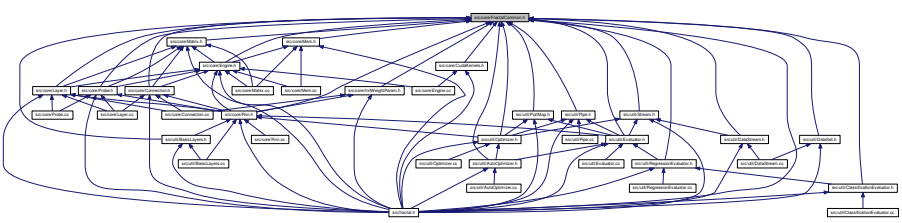
8.8 src/core/FractalCommon.h File Reference

```
#include <cstddef>
#include <cassert>
```

Include dependency graph for FractalCommon.h:



This graph shows which files directly or indirectly include this file:



Namespaces

- fractal

  The topmost namespace of libfractal.

Macros

- #define FRACTAL_SINGLE_PRECISION
- #define verify(expression) assert(expression)

Typedefs

- typedef float fractal::FLOAT

8.8.1 Macro Definition Documentation

8.8.1.1 #define FRACTAL_SINGLE_PRECISION

Definition at line 33 of file FractalCommon.h.

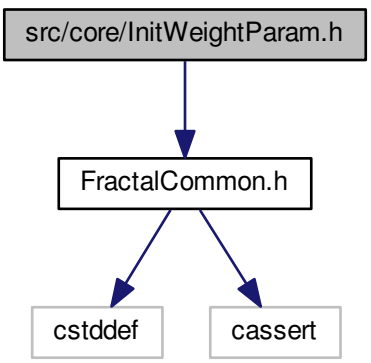
8.8.1.2 #define verify(*expression*) assert(expression)

Definition at line 43 of file FractalCommon.h.

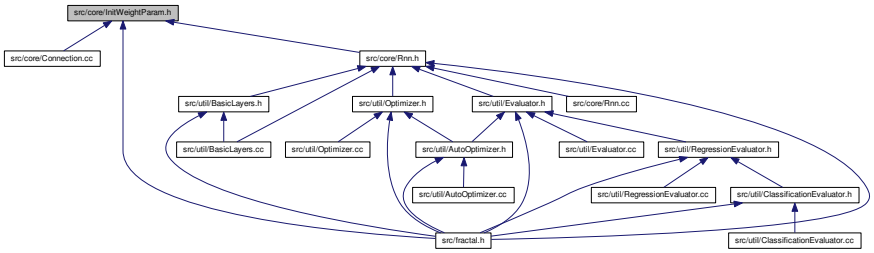
8.9 src/core/InitWeightParam.h File Reference

```
#include "FractalCommon.h"
```

Include dependency graph for InitWeightParam.h:



This graph shows which files directly or indirectly include this file:



Classes

- class fractal::InitWeightParam

Namespaces

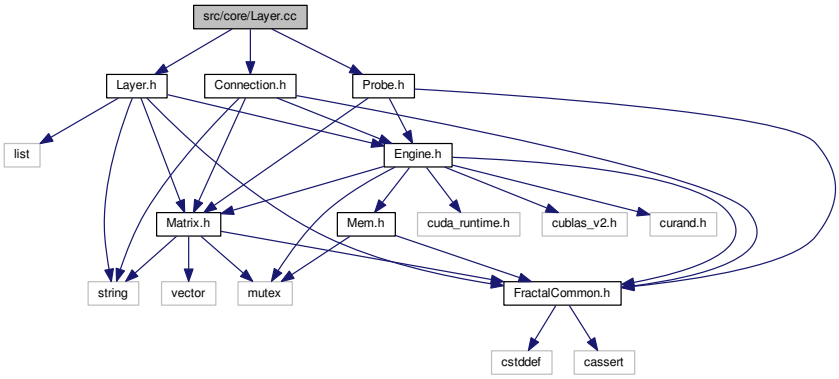
- fractal

  The topmost namespace of libfractal.

8.10 src/core/Layer.cc File Reference

```
#include "Layer.h"
#include "Connection.h"
#include "Probe.h"
```

Include dependency graph for Layer.cc:



Namespaces

- fractal

  The topmost namespace of libfractal.

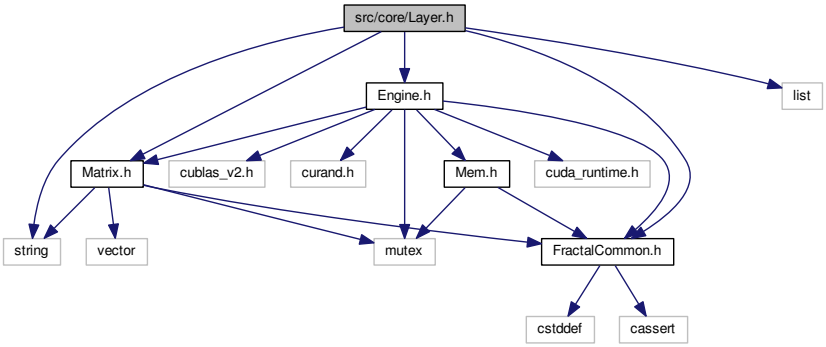
Variables

- const FLOAT fractal::NO_STATE_PENALTY = (FLOAT) -1

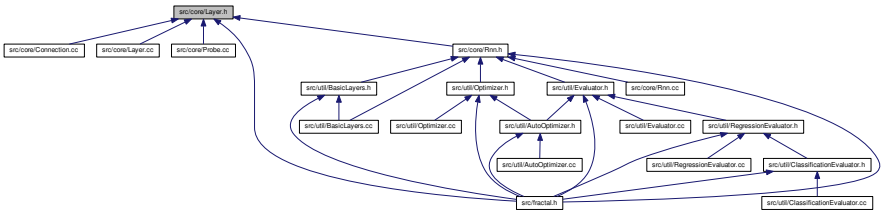
8.11 src/core/Layer.h File Reference

```
#include <string>
#include <list>
#include "Engine.h"
#include "Matrix.h"
#include "FractalCommon.h"
```

Include dependency graph for Layer.h:



This graph shows which files directly or indirectly include this file:



Classes

- class fractal::LayerParam
- class fractal::Layer

Namespaces

- fractal

  The topmost namespace of libfractal.

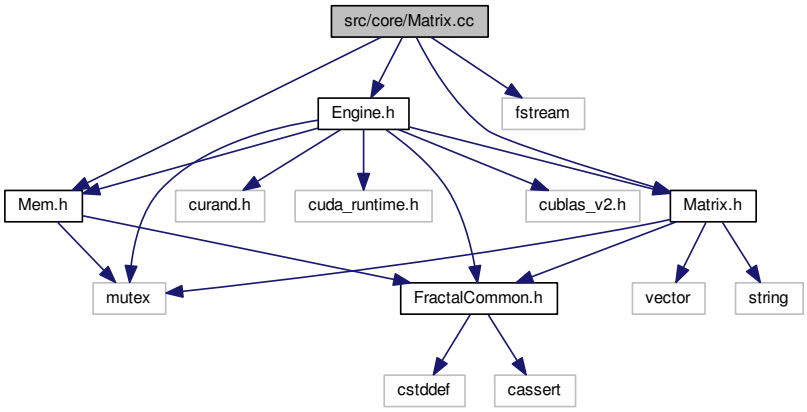
Enumerations

- enum fractal::ActType {
  fractal::ACT_BIAS, fractal::ACT_SIGMOID, fractal::ACT_TANH, fractal::ACT_SOFTPLUS,
  fractal::ACT_RECTLINEAR, fractal::ACT_LINEAR, fractal::ACT_ONE_MINUS_LINEAR, fractal::ACT_INV↩
  ERSE,
  fractal::ACT_SOFTMAX }
- enum fractal::StateType { fractal::AGG_DONTCARE, fractal::AGG_SUM, fractal::AGG_MULT }

8.12 src/core/Matrix.cc File Reference

```
#include "Matrix.h"
#include <fstream>
#include "Mem.h"
#include "Engine.h"
```

Include dependency graph for Matrix.cc:



Namespaces

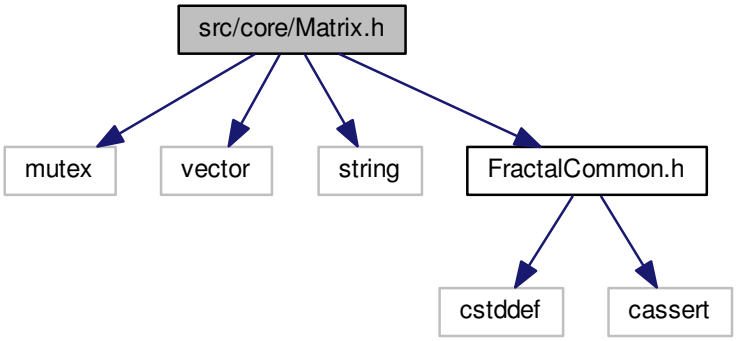
- fractal

  The topmost namespace of libfractal.

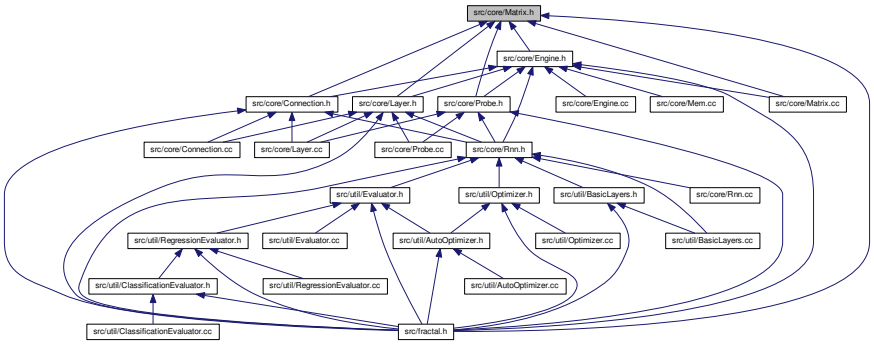
8.13 src/core/Matrix.h File Reference

```
#include <mutex>
#include <vector>
#include <string>
#include "FractalCommon.h"
```

Include dependency graph for Matrix.h:



This graph shows which files directly or indirectly include this file:



Classes

- class fractal::Matrix< T >

Namespaces

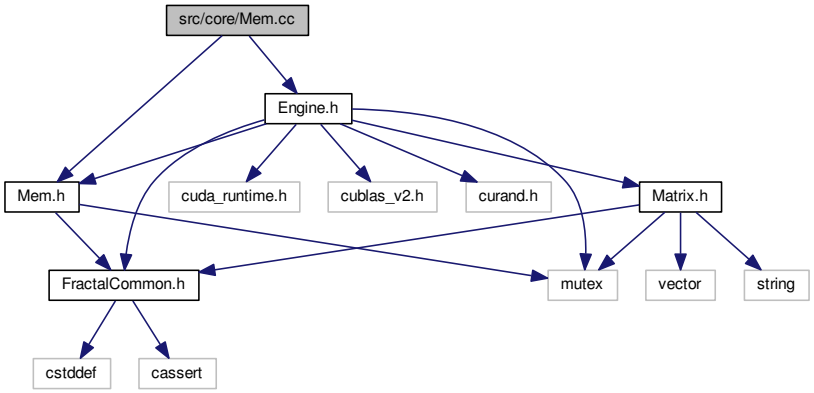
- fractal

  The topmost namespace of libfractal.

8.14 src/core/Mem.cc File Reference

```
#include "Mem.h"
#include "Engine.h"
```

Include dependency graph for Mem.cc:



Namespaces

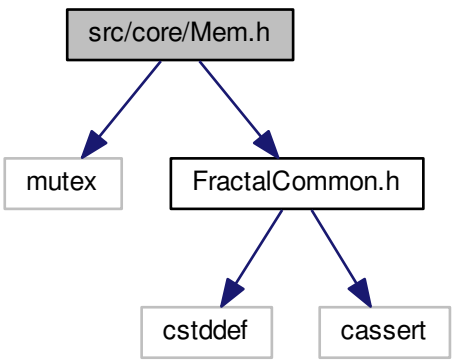
- fractal

  The topmost namespace of libfractal.

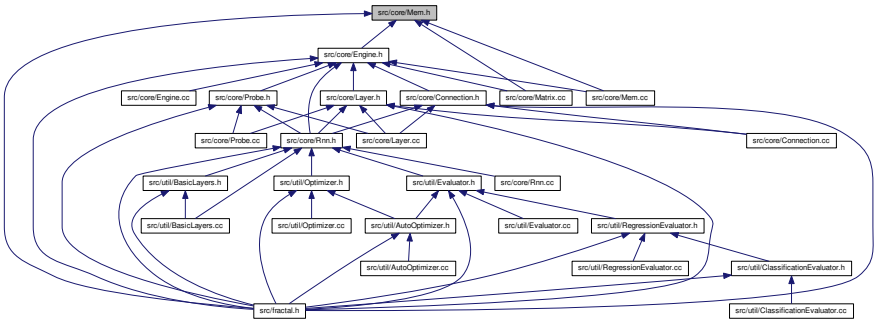
8.15 src/core/Mem.h File Reference

```
#include <mutex>
#include "FractalCommon.h"
```

Include dependency graph for Mem.h:



This graph shows which files directly or indirectly include this file:



Classes

- class fractal::Mem

Namespaces

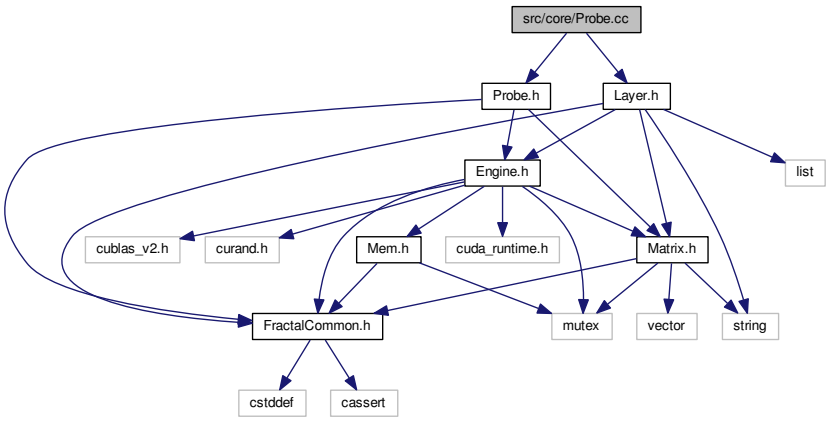
- fractal

  The topmost namespace of libfractal.

8.16 src/core/Probe.cc File Reference

```
#include "Probe.h"
#include "Layer.h"
```

Include dependency graph for Probe.cc:



Namespaces

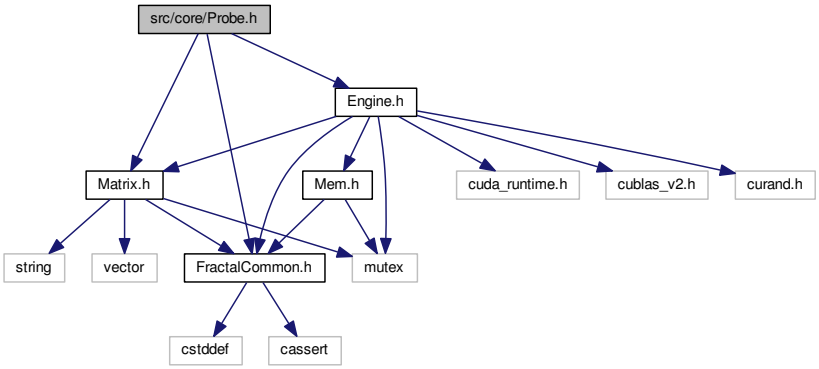
- fractal

  The topmost namespace of libfractal.

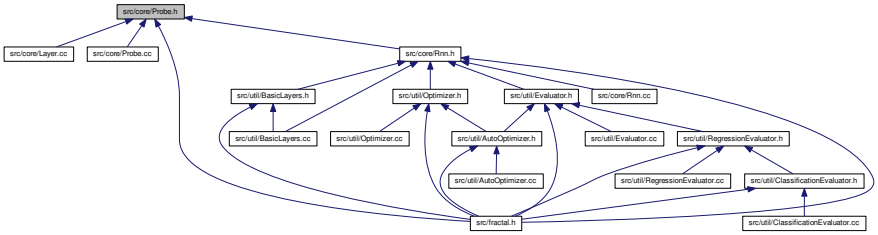
8.17 src/core/Probe.h File Reference

```
#include "Engine.h"
#include "Matrix.h"
#include "FractalCommon.h"
```

Include dependency graph for Probe.h:



This graph shows which files directly or indirectly include this file:



Classes

- class fractal::Probe

Namespaces

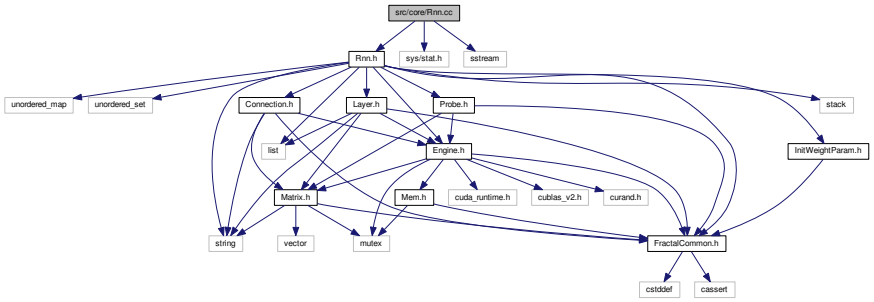
- fractal

  The topmost namespace of libfractal.

8.18 src/core/Rnn.cc File Reference

```
#include "Rnn.h"
#include <sys/stat.h>
#include <sstream>
```

Include dependency graph for Rnn.cc:



Namespaces

- fractal

  The topmost namespace of libfractal.

Macros

- #define MAX_NUM_PSTREAM 4

Variables

- static const long fractal::UNTOUCHED = -1
- static const long fractal::TOUCHED = -2
- static const long fractal::SCC_DETERMINED = -3

8.18.1 Macro Definition Documentation

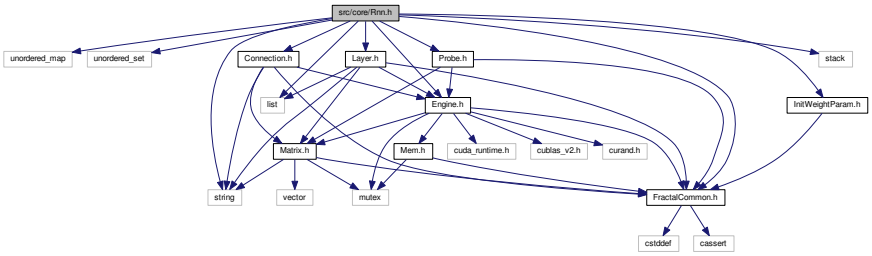
8.18.1.1 #define MAX_NUM_PSTREAM 4

Definition at line 28 of file Rnn.cc.

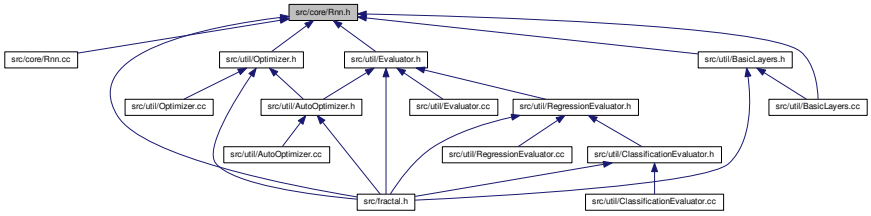
8.19 src/core/Rnn.h File Reference

```
#include <unordered_map>
#include <unordered_set>
#include <list>
#include <stack>
#include <string>
#include "InitWeightParam.h"
#include "Engine.h"
#include "Layer.h"
#include "Probe.h"
#include "Connection.h"
#include "FractalCommon.h"
```

Include dependency graph for Rnn.h:



This graph shows which files directly or indirectly include this file:



Classes

- class fractal::Rnn

  A network structure container.

Namespaces

- fractal

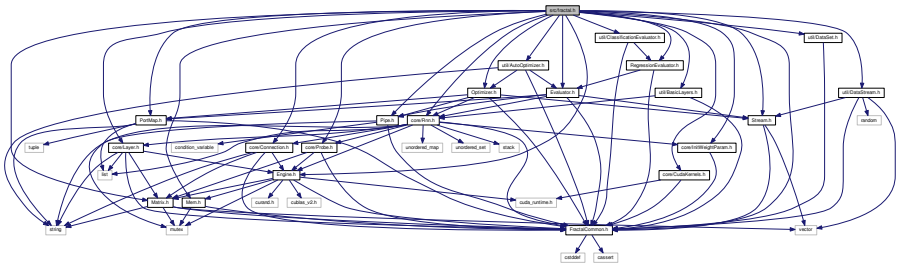
  The topmost namespace of libfractal.

8.20 src/fractal.h File Reference

```
#include "core/Connection.h"
```

```
#include "core/CudaKernels.h"
#include "core/Engine.h"
#include "core/FractalCommon.h"
#include "core/InitWeightParam.h"
#include "core/Layer.h"
#include "core/Matrix.h"
#include "core/Mem.h"
#include "core/Probe.h"
#include "core/Rnn.h"
#include "util/AutoOptimizer.h"
#include "util/BasicLayers.h"
#include "util/ClassificationEvaluator.h"
#include "util/DataSet.h"
#include "util/DataStream.h"
#include "util/Evaluator.h"
#include "util/Optimizer.h"
#include "util/Pipe.h"
#include "util/PortMap.h"
#include "util/RegressionEvaluator.h"
#include "util/Stream.h"
```

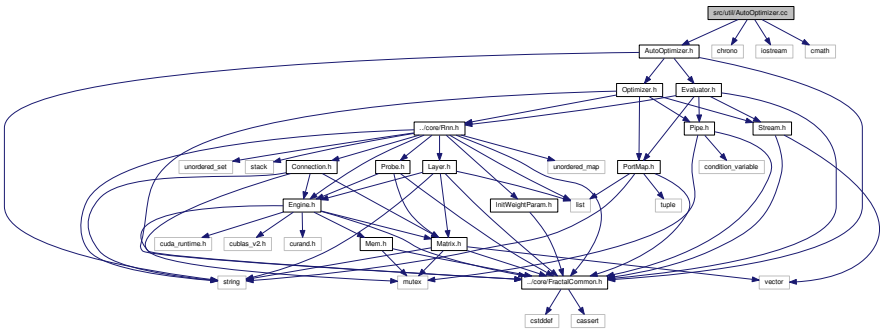
Include dependency graph for fractal.h:



8.21 src/util/AutoOptimizer.cc File Reference

```
#include "AutoOptimizer.h"
#include <chrono>
#include <iostream>
#include <cmath>
```

Include dependency graph for AutoOptimizer.cc:



Namespaces

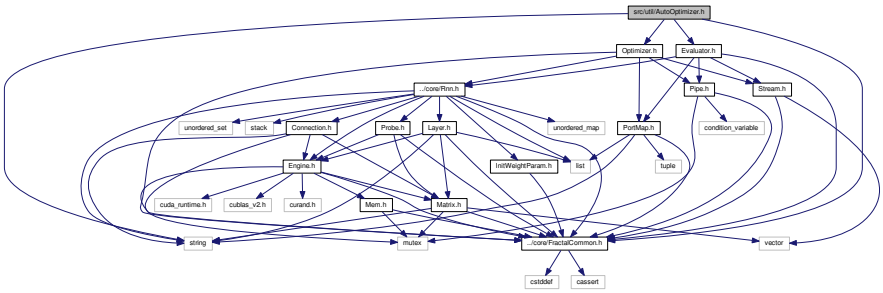
- fractal

  The topmost namespace of libfractal.

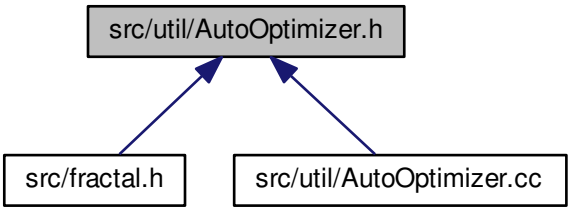
8.22 src/util/AutoOptimizer.h File Reference

```
#include <string>
#include "Optimizer.h"
#include "Evaluator.h"
#include "../core/FractalCommon.h"
```

Include dependency graph for AutoOptimizer.h:



This graph shows which files directly or indirectly include this file:



Classes

- class fractal::AutoOptimizer

Namespaces

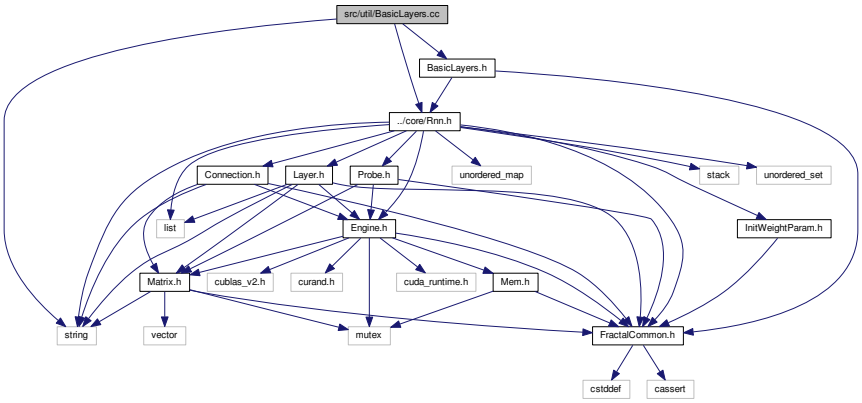
- fractal

  The topmost namespace of libfractal.

8.23 src/util/BasicLayers.cc File Reference

```
#include "BasicLayers.h"
#include <string>
#include "../core/Rnn.h"
```

Include dependency graph for BasicLayers.cc:



Namespaces

- fractal

  The topmost namespace of libfractal.
- fractal::basicLayers

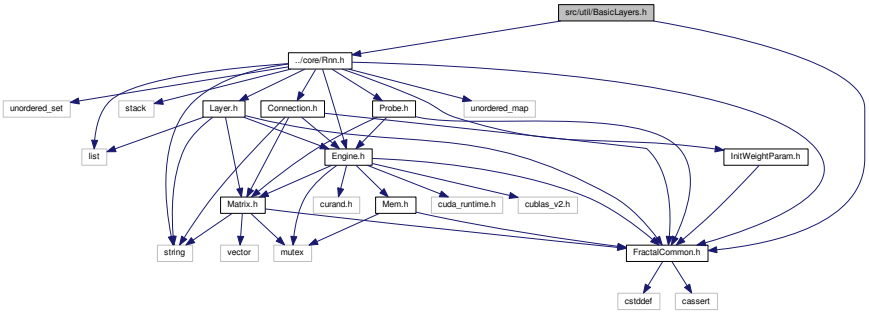
Functions

- void fractal::basicLayers::AddLstmLayer (Rnn &rnn, const std::string name, const std::string biasLayer, const unsigned long delayAmount, const unsigned long size, const InitWeightParam &initWeightParam, const FL↩ OAT initForgetGateBias)

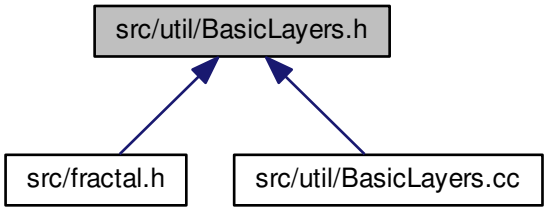
8.24 src/util/BasicLayers.h File Reference

```
#include "../core/Rnn.h"
#include "../core/FractalCommon.h"
```

Include dependency graph for BasicLayers.h:



This graph shows which files directly or indirectly include this file:



Namespaces

- fractal

  The topmost namespace of libfractal.
- fractal::basicLayers

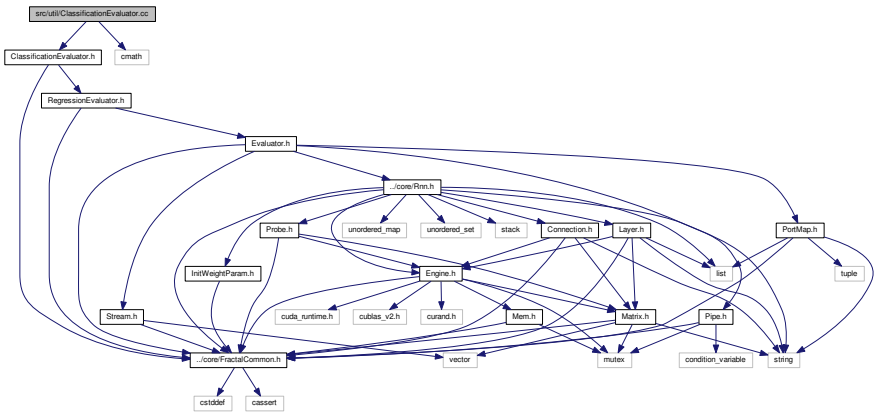
Functions

- void fractal::basicLayers::AddLstmLayer (Rnn &rnn, const std::string name, const std::string biasLayer, const unsigned long delayAmount, const unsigned long size, const InitWeightParam &initWeightParam, const FL↩ OAT initForgetGateBias)

8.25 src/util/ClassificationEvaluator.cc File Reference

```
#include "ClassificationEvaluator.h"
#include <cmath>
```

Include dependency graph for ClassificationEvaluator.cc:



Namespaces

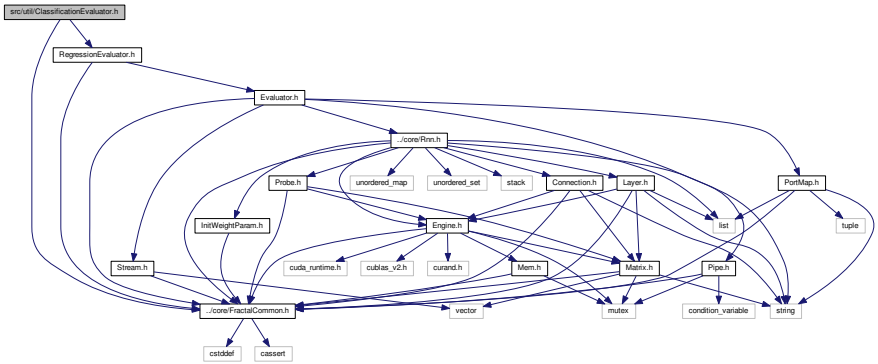
- fractal

The topmost namespace of libfractal.

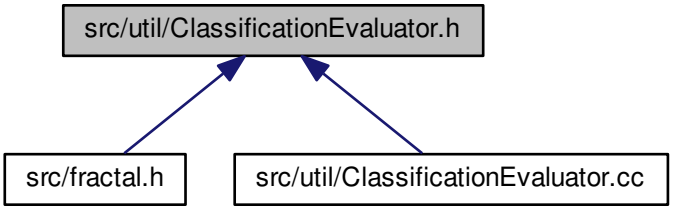
8.26 src/util/ClassificationEvaluator.h File Reference

```
#include "RegressionEvaluator.h"
#include "../core/FractalCommon.h"
```

Include dependency graph for ClassificationEvaluator.h:



This graph shows which files directly or indirectly include this file:



Classes

- class fractal::ClassificationEvaluator

Namespaces

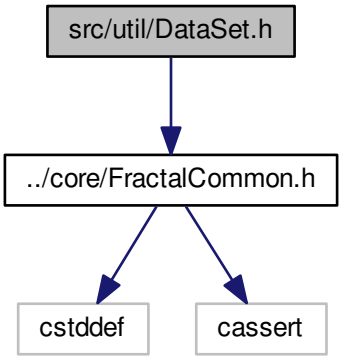
- fractal

  The topmost namespace of libfractal.

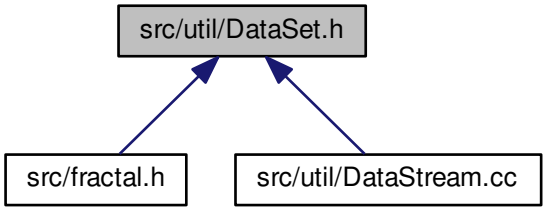
8.27 src/util/DataSet.h File Reference

```
#include "../core/FractalCommon.h"
```

Include dependency graph for DataSet.h:



This graph shows which files directly or indirectly include this file:



Classes

- class fractal::DataSet

Namespaces

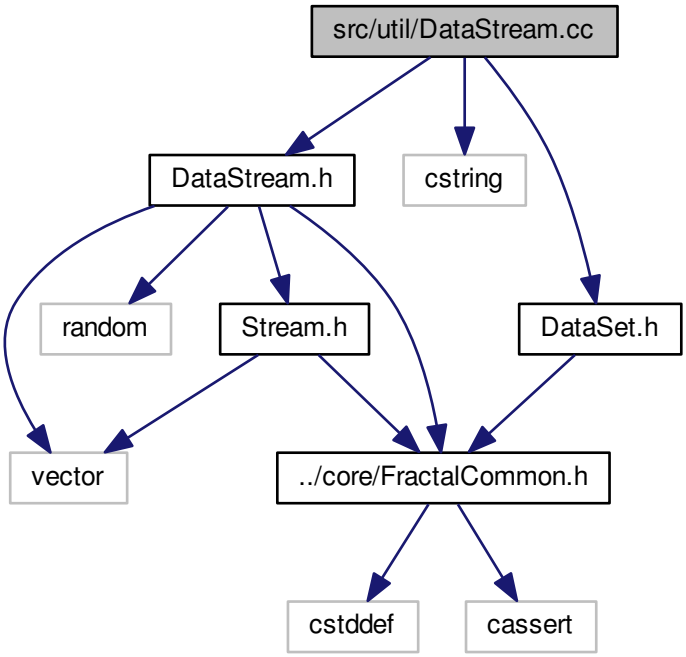
- fractal

  The topmost namespace of libfractal.

8.28 src/util/DataStream.cc File Reference

```
#include "DataStream.h"
#include <cstring>
#include "DataSet.h"
```

Include dependency graph for DataStream.cc:



Namespaces

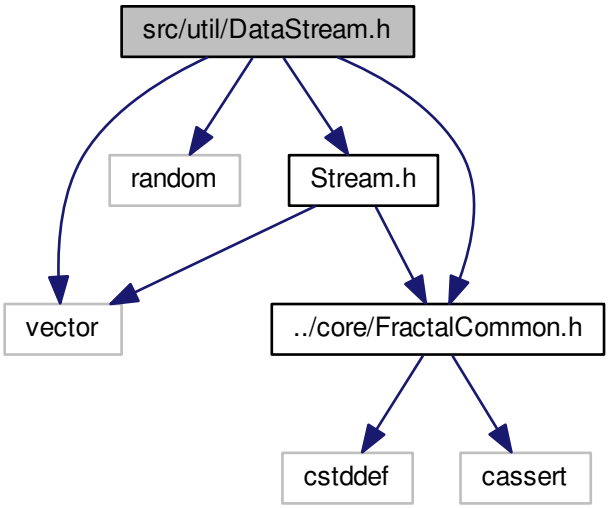
- fractal

  The topmost namespace of libfractal.

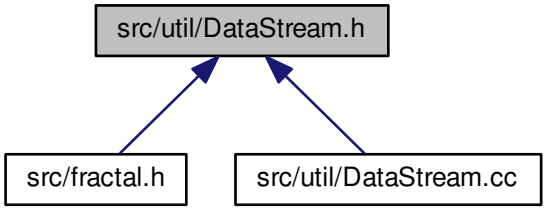
8.29 src/util/DataStream.h File Reference

```
#include <vector>
#include <random>
#include "Stream.h"
#include "../core/FractalCommon.h"
```

Include dependency graph for DataStream.h:



This graph shows which files directly or indirectly include this file:



Classes

- class fractal::DataStream

Namespaces

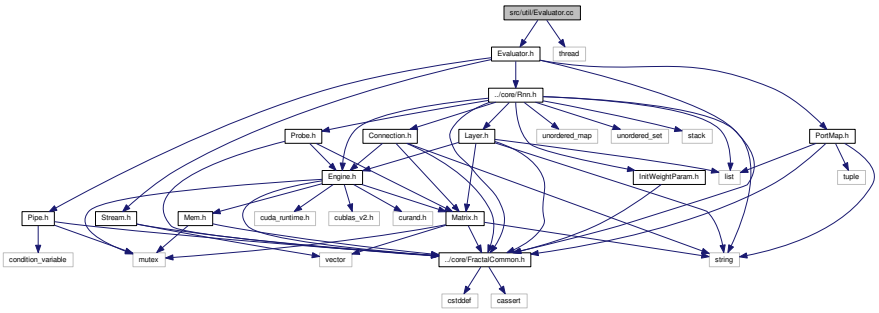
- fractal

  The topmost namespace of libfractal.

8.30 src/util/Evaluator.cc File Reference

```
#include "Evaluator.h"
```

```
#include <thread>
```
Include dependency graph for Evaluator.cc:



Namespaces

- fractal

  The topmost namespace of libfractal.

Macros

- #define LOC 1

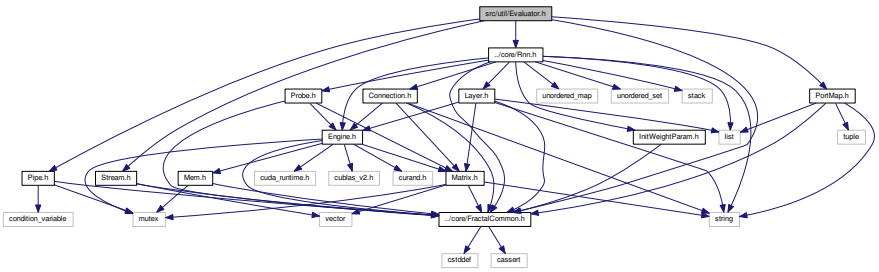
8.30.1 Macro Definition Documentation

8.30.1.1 #define LOC 1

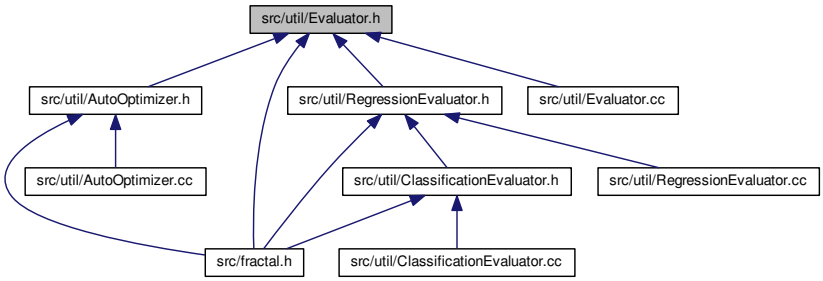
Definition at line 27 of file Evaluator.cc.

8.31 src/util/Evaluator.h File Reference

```
#include "Pipe.h"
#include "PortMap.h"
#include "Stream.h"
#include "../core/Rnn.h"
#include "../core/FractalCommon.h"
```
Include dependency graph for Evaluator.h:



This graph shows which files directly or indirectly include this file:



Classes

- class fractal::EvaluateArgs
- class fractal::Evaluator

Namespaces

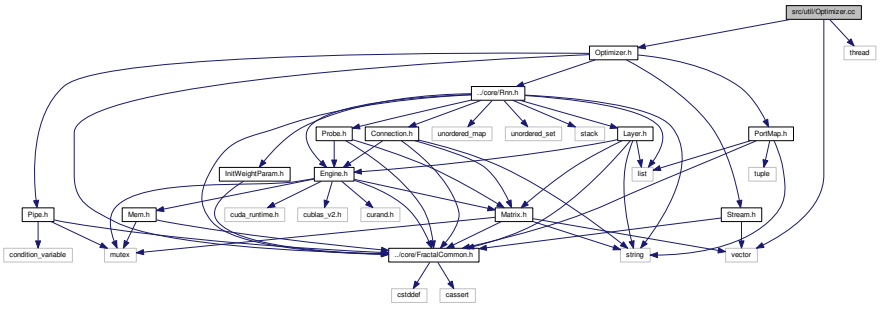
- fractal

  The topmost namespace of libfractal.

8.32 src/util/Optimizer.cc File Reference

```
#include "Optimizer.h"
#include <thread>
#include <vector>
```

Include dependency graph for Optimizer.cc:



Namespaces

- fractal

  The topmost namespace of libfractal.

Macros

- #define LOC 1

8.32.1 Macro Definition Documentation

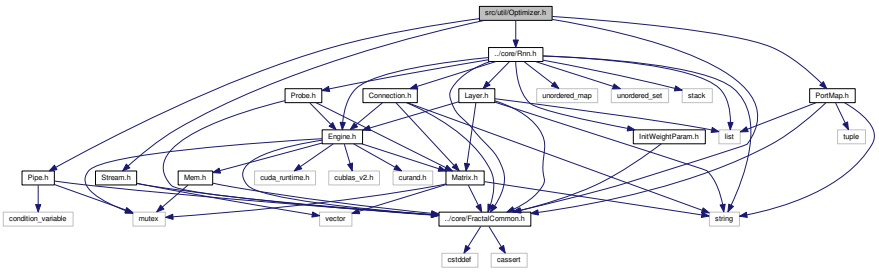
8.32.1.1 #define LOC 1

Definition at line 29 of file Optimizer.cc.

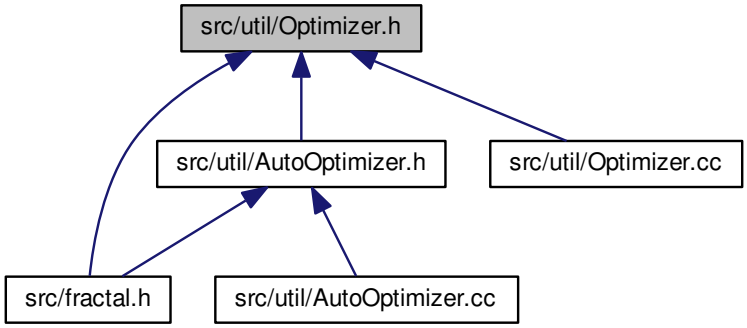
8.33 src/util/Optimizer.h File Reference

```
#include "Pipe.h"
#include "PortMap.h"
#include "Stream.h"
#include "../core/Rnn.h"
#include "../core/FractalCommon.h"
```

Include dependency graph for Optimizer.h:



This graph shows which files directly or indirectly include this file:



Classes

- class fractal::BackpropArgs
- class fractal::Optimizer

Namespaces

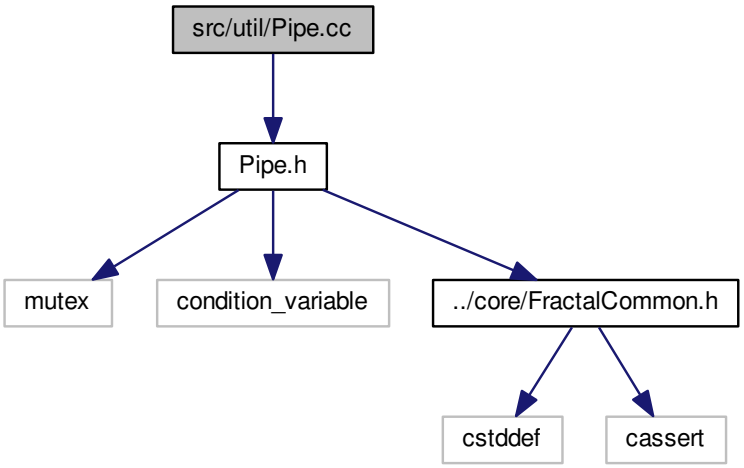
- fractal

  The topmost namespace of libfractal.

8.34 src/util/Pipe.cc File Reference

```
#include "Pipe.h"
```

Include dependency graph for Pipe.cc:



Namespaces

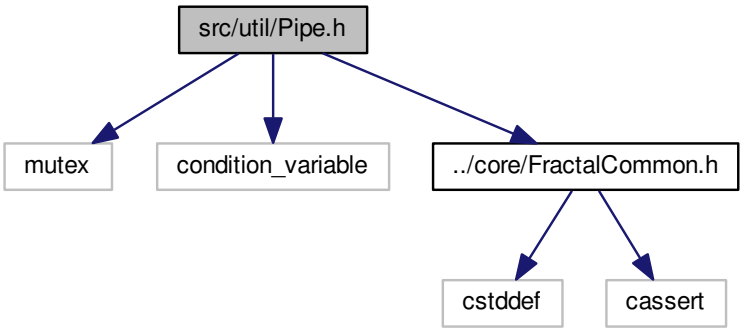
- fractal

  The topmost namespace of libfractal.

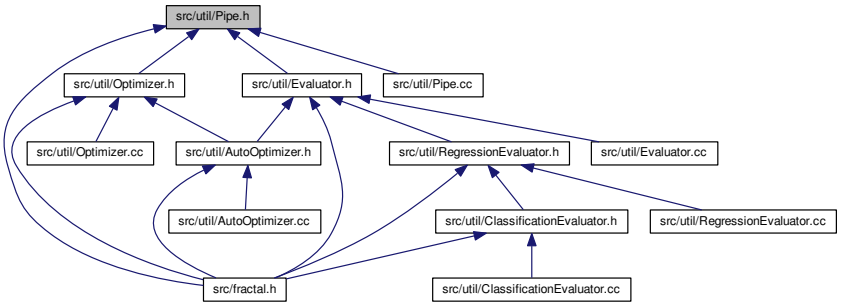
8.35 src/util/Pipe.h File Reference

```
#include <mutex>
#include <condition_variable>
#include "../core/FractalCommon.h"
```

Include dependency graph for Pipe.h:



This graph shows which files directly or indirectly include this file:



Classes

- class fractal::Pipe

Namespaces

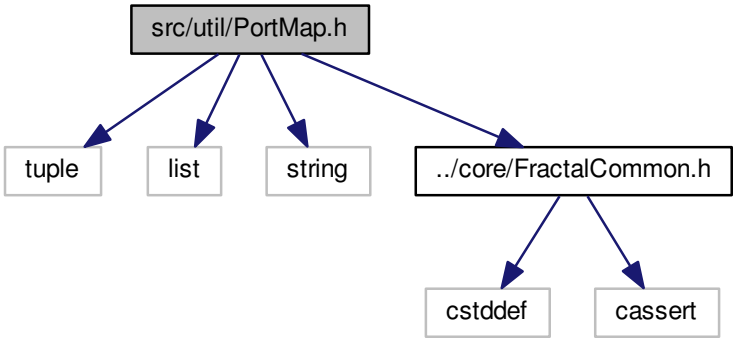
- fractal

  The topmost namespace of libfractal.

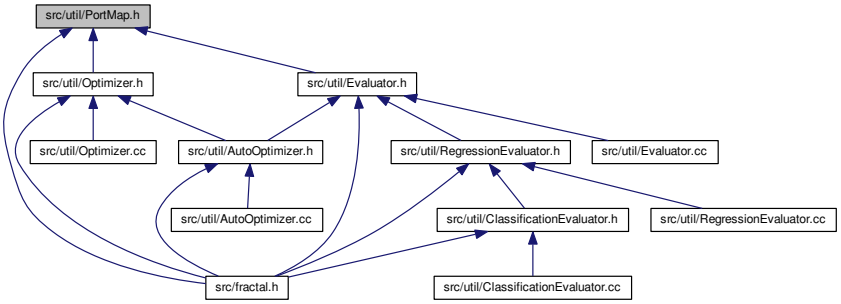
8.36 src/util/PortMap.h File Reference

```
#include <tuple>
#include <list>
#include <string>
#include "../core/FractalCommon.h"
```

Include dependency graph for PortMap.h:



This graph shows which files directly or indirectly include this file:



Namespaces

- fractal

  The topmost namespace of libfractal.

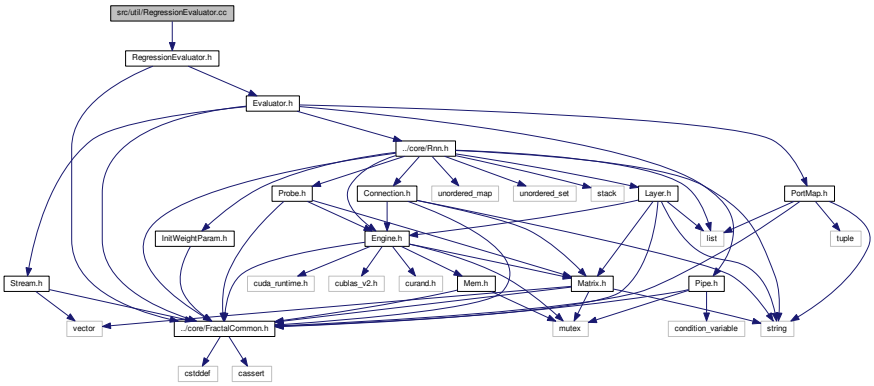
Typedefs

- typedef std::tuple
  < std::string, unsigned long > fractal::PortMap
- typedef std::list< PortMap > fractal::PortMapList

8.37 src/util/RegressionEvaluator.cc File Reference

```
#include "RegressionEvaluator.h"
```

Include dependency graph for RegressionEvaluator.cc:



Namespaces

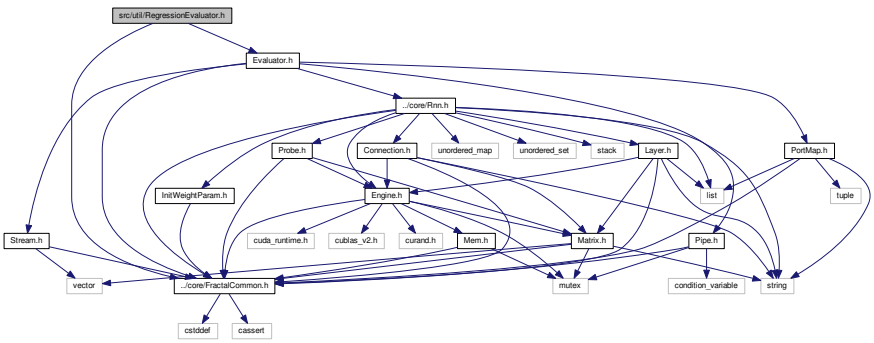
- fractal

  The topmost namespace of libfractal.

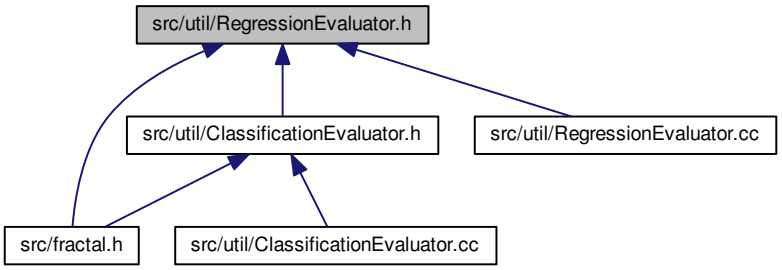
8.38 src/util/RegressionEvaluator.h File Reference

```
#include "Evaluator.h"
#include "../core/FractalCommon.h"
```

Include dependency graph for RegressionEvaluator.h:



This graph shows which files directly or indirectly include this file:



Classes

- class fractal::RegressionEvaluator

Namespaces

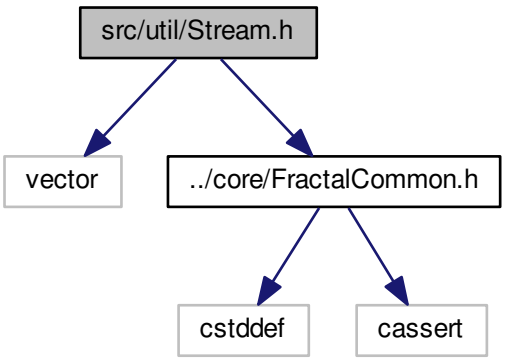
- fractal

  The topmost namespace of libfractal.

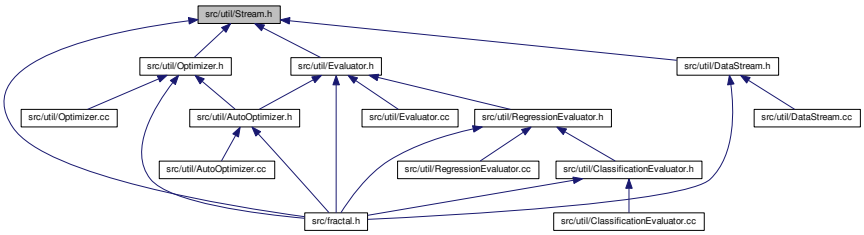
8.39 src/util/Stream.h File Reference

```
#include <vector>
#include "../core/FractalCommon.h"
```

Include dependency graph for Stream.h:



This graph shows which files directly or indirectly include this file:



Classes

- class fractal::Stream

Namespaces

- fractal

  The topmost namespace of libfractal.

Index