

qpp  
0.1

Generated by Doxygen 1.8.5

Sun Apr 6 2014 23:07:20



# Contents

<b>1</b>	<b>Namespace Index</b>	<b>1</b>
1.1	Namespace List . . . . .	1
<b>2</b>	<b>Hierarchical Index</b>	<b>3</b>
2.1	Class Hierarchy . . . . .	3
<b>3</b>	<b>Class Index</b>	<b>5</b>
3.1	Class List . . . . .	5
<b>4</b>	<b>File Index</b>	<b>7</b>
4.1	File List . . . . .	7
<b>5</b>	<b>Namespace Documentation</b>	<b>9</b>
5.1	qpp Namespace Reference . . . . .	9
5.1.1	Function Documentation . . . . .	12
5.1.1.1	_init . . . . .	12
5.1.1.2	absm . . . . .	12
5.1.1.3	adjoint . . . . .	13
5.1.1.4	anticomm . . . . .	13
5.1.1.5	channel . . . . .	13
5.1.1.6	choi . . . . .	14
5.1.1.7	choi2kraus . . . . .	14
5.1.1.8	comm . . . . .	15
5.1.1.9	conjugate . . . . .	15
5.1.1.10	cosm . . . . .	15
5.1.1.11	det . . . . .	16
5.1.1.12	disp . . . . .	16
5.1.1.13	disp . . . . .	16
5.1.1.14	disp . . . . .	16
5.1.1.15	disp . . . . .	16
5.1.1.16	displn . . . . .	16
5.1.1.17	displn . . . . .	17
5.1.1.18	displn . . . . .	17

5.1.1.19	<a href="#">displn</a>	17
5.1.1.20	<a href="#">evals</a>	18
5.1.1.21	<a href="#">evecs</a>	18
5.1.1.22	<a href="#">expandout</a>	19
5.1.1.23	<a href="#">expm</a>	19
5.1.1.24	<a href="#">fun</a>	20
5.1.1.25	<a href="#">funm</a>	20
5.1.1.26	<a href="#">grams</a>	21
5.1.1.27	<a href="#">grams</a>	21
5.1.1.28	<a href="#">hevals</a>	22
5.1.1.29	<a href="#">hevecs</a>	22
5.1.1.30	<a href="#">kron</a>	22
5.1.1.31	<a href="#">kronlist</a>	23
5.1.1.32	<a href="#">kronpow</a>	23
5.1.1.33	<a href="#">load</a>	23
5.1.1.34	<a href="#">loadMATLABmatrix</a>	23
5.1.1.35	<a href="#">loadMATLABmatrix</a>	23
5.1.1.36	<a href="#">loadMATLABmatrix</a>	23
5.1.1.37	<a href="#">logm</a>	24
5.1.1.38	<a href="#">norm</a>	24
5.1.1.39	<a href="#">powm</a>	24
5.1.1.40	<a href="#">proj</a>	25
5.1.1.41	<a href="#">ptrace</a>	25
5.1.1.42	<a href="#">ptrace2</a>	26
5.1.1.43	<a href="#">ptranspose</a>	26
5.1.1.44	<a href="#">rand</a>	27
5.1.1.45	<a href="#">rand</a>	27
5.1.1.46	<a href="#">rand</a>	27
5.1.1.47	<a href="#">rand</a>	27
5.1.1.48	<a href="#">randH</a>	27
5.1.1.49	<a href="#">randket</a>	27
5.1.1.50	<a href="#">randKraus</a>	28
5.1.1.51	<a href="#">randn</a>	28
5.1.1.52	<a href="#">randn</a>	28
5.1.1.53	<a href="#">randn</a>	28
5.1.1.54	<a href="#">randn</a>	29
5.1.1.55	<a href="#">randrho</a>	29
5.1.1.56	<a href="#">randU</a>	29
5.1.1.57	<a href="#">randV</a>	29
5.1.1.58	<a href="#">renyi</a>	30

5.1.1.59	renyi_inf	30
5.1.1.60	reshape	30
5.1.1.61	save	31
5.1.1.62	saveMATLABmatrix	31
5.1.1.63	saveMATLABmatrix	31
5.1.1.64	saveMATLABmatrix	31
5.1.1.65	shannon	32
5.1.1.66	sinm	32
5.1.1.67	spectralpowm	32
5.1.1.68	sqrtn	33
5.1.1.69	sum	33
5.1.1.70	super	33
5.1.1.71	syspermute	34
5.1.1.72	trace	34
5.1.1.73	transpose	35
5.2	qpp::ct Namespace Reference	35
5.2.1	Function Documentation	35
5.2.1.1	omega	35
5.2.2	Variable Documentation	35
5.2.2.1	chop	35
5.2.2.2	ee	35
5.2.2.3	eps	35
5.2.2.4	ii	35
5.2.2.5	pi	35
5.3	qpp::gt Namespace Reference	35
5.3.1	Function Documentation	36
5.3.1.1	_init_gates	36
5.3.1.2	CTRL	37
5.3.1.3	Fd	37
5.3.1.4	Id	37
5.3.1.5	Rtheta	37
5.3.1.6	Xd	38
5.3.1.7	Zd	38
5.3.2	Variable Documentation	38
5.3.2.1	b00	38
5.3.2.2	b01	38
5.3.2.3	b10	38
5.3.2.4	b11	38
5.3.2.5	CNOTab	38
5.3.2.6	CNOTba	38

5.3.2.7	CS	38
5.3.2.8	CZ	38
5.3.2.9	FRED	38
5.3.2.10	H	38
5.3.2.11	Id2	38
5.3.2.12	S	39
5.3.2.13	SWAP	39
5.3.2.14	T	39
5.3.2.15	TOF	39
5.3.2.16	X	39
5.3.2.17	x0	39
5.3.2.18	x1	39
5.3.2.19	Y	39
5.3.2.20	y0	39
5.3.2.21	y1	39
5.3.2.22	Z	39
5.3.2.23	z0	39
5.3.2.24	z1	39
5.4	qpp::internal Namespace Reference	39
5.4.1	Function Documentation	40
5.4.1.1	_check_col_vector	40
5.4.1.2	_check_dims	40
5.4.1.3	_check_dims_match_mat	40
5.4.1.4	_check_eq_dims	40
5.4.1.5	_check_nonzero_size	40
5.4.1.6	_check_perm	40
5.4.1.7	_check_row_vector	40
5.4.1.8	_check_square_mat	40
5.4.1.9	_check_subsys	40
5.4.1.10	_check_vector	40
5.4.1.11	_multiidx2n	40
5.4.1.12	_n2multiidx	40
5.4.1.13	_ptranspose_worker	40
5.4.1.14	_syspermute_worker	41
5.5	qpp::stat Namespace Reference	41
5.5.1	Variable Documentation	41
5.5.1.1	_rd	41
5.5.1.2	_rng	41
5.6	qpp::types Namespace Reference	41
5.6.1	Typedef Documentation	42

5.6.1.1	<a href="#">cmat</a>	42
5.6.1.2	<a href="#">cplx</a>	42
5.6.1.3	<a href="#">dmat</a>	42
5.6.1.4	<a href="#">DynMat</a>	42
5.6.1.5	<a href="#">Expression2DynMat</a>	42
5.6.1.6	<a href="#">fmat</a>	42
5.6.1.7	<a href="#">imat</a>	42
5.6.2	<a href="#">Function Documentation</a>	42
5.6.2.1	<a href="#">myfunc</a>	42
<b>6</b>	<b><a href="#">Class Documentation</a></b>	<b>43</b>
6.1	<a href="#">qpp::stat::DiscreteDistribution Class Reference</a>	43
6.1.1	<a href="#">Constructor &amp; Destructor Documentation</a>	43
6.1.1.1	<a href="#">DiscreteDistribution</a>	43
6.1.1.2	<a href="#">DiscreteDistribution</a>	43
6.1.1.3	<a href="#">DiscreteDistribution</a>	43
6.1.2	<a href="#">Member Function Documentation</a>	43
6.1.2.1	<a href="#">probabilities</a>	43
6.1.2.2	<a href="#">sample</a>	43
6.1.3	<a href="#">Member Data Documentation</a>	43
6.1.3.1	<a href="#">_d</a>	43
6.2	<a href="#">qpp::stat::DiscreteDistributionFromComplex Class Reference</a>	44
6.2.1	<a href="#">Constructor &amp; Destructor Documentation</a>	44
6.2.1.1	<a href="#">DiscreteDistributionFromComplex</a>	44
6.2.1.2	<a href="#">DiscreteDistributionFromComplex</a>	45
6.2.1.3	<a href="#">DiscreteDistributionFromComplex</a>	45
6.2.1.4	<a href="#">DiscreteDistributionFromComplex</a>	45
6.2.2	<a href="#">Member Function Documentation</a>	45
6.2.2.1	<a href="#">cplx2amplitudes</a>	46
6.2.2.2	<a href="#">probabilities</a>	46
6.2.2.3	<a href="#">sample</a>	46
6.2.3	<a href="#">Member Data Documentation</a>	46
6.2.3.1	<a href="#">_d</a>	46
6.3	<a href="#">qpp::Exception Class Reference</a>	46
6.3.1	<a href="#">Member Enumeration Documentation</a>	47
6.3.1.1	<a href="#">Type</a>	47
6.3.2	<a href="#">Constructor &amp; Destructor Documentation</a>	48
6.3.2.1	<a href="#">Exception</a>	48
6.3.2.2	<a href="#">Exception</a>	48
6.3.2.3	<a href="#">~Exception</a>	48

6.3.3	Member Function Documentation	48
6.3.3.1	_construct_exception_msg	48
6.3.3.2	what	48
6.3.4	Member Data Documentation	48
6.3.4.1	_custom	48
6.3.4.2	_msg	48
6.3.4.3	_type	48
6.3.4.4	_where	48
6.4	qpp::stat::NormalDistribution Class Reference	49
6.4.1	Constructor & Destructor Documentation	49
6.4.1.1	NormalDistribution	49
6.4.2	Member Function Documentation	49
6.4.2.1	sample	49
6.4.3	Member Data Documentation	49
6.4.3.1	_d	49
6.5	qpp::Timer Class Reference	49
6.5.1	Constructor & Destructor Documentation	50
6.5.1.1	Timer	50
6.5.1.2	~Timer	50
6.5.2	Member Function Documentation	50
6.5.2.1	seconds	50
6.5.2.2	tic	50
6.5.2.3	toc	50
6.5.3	Friends And Related Function Documentation	50
6.5.3.1	operator<<	50
6.5.4	Member Data Documentation	50
6.5.4.1	_end	50
6.5.4.2	_start	50
6.6	qpp::stat::UniformRealDistribution Class Reference	50
6.6.1	Constructor & Destructor Documentation	50
6.6.1.1	UniformRealDistribution	50
6.6.2	Member Function Documentation	50
6.6.2.1	sample	50
6.6.3	Member Data Documentation	50
6.6.3.1	_d	51
<b>7</b>	<b>File Documentation</b>	<b>53</b>
7.1	include/channels.h File Reference	53
7.2	include/constants.h File Reference	54
7.3	include/entropies.h File Reference	55



7.4	<a href="#">include/exception.h File Reference</a>	57
7.5	<a href="#">include/functions.h File Reference</a>	58
7.6	<a href="#">include/gates.h File Reference</a>	60
7.7	<a href="#">include/internal.h File Reference</a>	62
7.8	<a href="#">include/io.h File Reference</a>	64
7.9	<a href="#">include/matlab.h File Reference</a>	65
7.10	<a href="#">include/qpp.h File Reference</a>	66
7.11	<a href="#">include/random.h File Reference</a>	67
7.12	<a href="#">include/stat.h File Reference</a>	69
7.13	<a href="#">include/timer.h File Reference</a>	70
7.14	<a href="#">include/types.h File Reference</a>	71
7.15	<a href="#">src/main.cpp File Reference</a>	73
7.15.1	<a href="#">Function Documentation</a>	73
7.15.1.1	<a href="#">main</a>	74



# Chapter 1

## Namespace Index

### 1.1 Namespace List

Here is a list of all namespaces with brief descriptions:

<a href="#">qpp</a>	9
<a href="#">qpp::ct</a>	35
<a href="#">qpp::gt</a>	35
<a href="#">qpp::internal</a>	39
<a href="#">qpp::stat</a>	41
<a href="#">qpp::types</a>	41



## Chapter 2

# Hierarchical Index

### 2.1 Class Hierarchy

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

qpp::stat::DiscreteDistribution . . . . .	43
qpp::stat::DiscreteDistributionFromComplex . . . . .	44
exception	
qpp::Exception . . . . .	46
qpp::stat::NormalDistribution . . . . .	49
qpp::Timer . . . . .	49
qpp::stat::UniformRealDistribution . . . . .	50



## Chapter 3

# Class Index

### 3.1 Class List

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

<a href="#">qpp::stat::DiscreteDistribution</a>	43
<a href="#">qpp::stat::DiscreteDistributionFromComplex</a>	44
<a href="#">qpp::Exception</a>	46
<a href="#">qpp::stat::NormalDistribution</a>	49
<a href="#">qpp::Timer</a>	49
<a href="#">qpp::stat::UniformRealDistribution</a>	50





# Chapter 4

## File Index

### 4.1 File List

Here is a list of all files with brief descriptions:

include/channels.h	53
include/constants.h	54
include/entropies.h	55
include/exception.h	57
include/functions.h	58
include/gates.h	60
include/internal.h	62
include/io.h	64
include/matlab.h	65
include/qpp.h	66
include/random.h	67
include/stat.h	69
include/timer.h	70
include/types.h	71
src/main.cpp	73



## Chapter 5

# Namespace Documentation

### 5.1 qpp Namespace Reference

#### Namespaces

- [ct](#)
- [gt](#)
- [internal](#)
- [stat](#)
- [types](#)

#### Classes

- class [Exception](#)
- class [Timer](#)

#### Functions

- [types::cmat channel](#) (const [types::cmat](#) &rho, const std::vector< [types::cmat](#) > &Ks)
- [types::cmat super](#) (const std::vector< [types::cmat](#) > &Ks)
- [types::cmat choi](#) (const std::vector< [types::cmat](#) > &Ks)
- std::vector< [types::cmat](#) > [choi2kraus](#) (const [types::cmat](#) &A)
- template<typename Scalar >  
double [shannon](#) (const [types::DynMat](#)< Scalar > &A)
- template<typename Scalar >  
double [renyi](#) (const double alpha, const [types::DynMat](#)< Scalar > &A)
- template<typename Scalar >  
double [renyi\\_inf](#) (const [types::DynMat](#)< Scalar > &A)
- template<typename Scalar >  
[types::DynMat](#)< Scalar > [transpose](#) (const [types::DynMat](#)< Scalar > &A)
- template<typename Scalar >  
[types::DynMat](#)< Scalar > [conjugate](#) (const [types::DynMat](#)< Scalar > &A)
- template<typename Scalar >  
[types::DynMat](#)< Scalar > [adjoint](#) (const [types::DynMat](#)< Scalar > &A)
- template<typename Scalar >  
Scalar [trace](#) (const [types::DynMat](#)< Scalar > &A)
- template<typename Scalar >  
Scalar [det](#) (const [types::DynMat](#)< Scalar > &A)
- template<typename Scalar >  
Scalar [sum](#) (const [types::DynMat](#)< Scalar > &A)

- `template<typename Scalar >`  
`double norm (const types::DynMat< Scalar > &A)`
- `template<typename Scalar >`  
`types::cmat evals (const types::DynMat< Scalar > &A)`
- `template<typename Scalar >`  
`types::cmat evects (const types::DynMat< Scalar > &A)`
- `template<typename Scalar >`  
`types::cmat hevals (const types::DynMat< Scalar > &A)`
- `template<typename Scalar >`  
`types::cmat hevects (const types::DynMat< Scalar > &A)`
- `template<typename Scalar >`  
`types::cmat funm (const types::DynMat< Scalar > &A, types::cplx(*f)(const types::cplx &))`
- `template<typename Scalar >`  
`types::cmat absm (const types::DynMat< Scalar > &A)`
- `template<typename Scalar >`  
`types::cmat expm (const types::DynMat< Scalar > &A)`
- `template<typename Scalar >`  
`types::cmat logm (const types::DynMat< Scalar > &A)`
- `template<typename Scalar >`  
`types::cmat sqrtm (const types::DynMat< Scalar > &A)`
- `template<typename Scalar >`  
`types::cmat sinm (const types::DynMat< Scalar > &A)`
- `template<typename Scalar >`  
`types::cmat cosm (const types::DynMat< Scalar > &A)`
- `template<typename Scalar >`  
`types::cmat spectralpowm (const types::DynMat< Scalar > &A, const types::cplx z)`
- `template<typename Scalar >`  
`types::DynMat< Scalar > powm (const types::DynMat< Scalar > &A, size_t n)`
- `template<typename InputScalar , typename OutputScalar >`  
`types::DynMat< OutputScalar > fun (const types::DynMat< InputScalar > &A, OutputScalar(*f)(const InputScalar &))`
- `template<typename Scalar >`  
`types::DynMat< Scalar > kron (const types::DynMat< Scalar > &A, const types::DynMat< Scalar > &B)`
- `template<typename Scalar >`  
`types::DynMat< Scalar > kronlist (const std::vector< types::DynMat< Scalar > > &list)`
- `template<typename Scalar >`  
`types::DynMat< Scalar > kronpow (const types::DynMat< Scalar > &A, size_t n)`
- `template<typename Scalar >`  
`types::DynMat< Scalar > reshape (const types::DynMat< Scalar > &A, size_t rows, size_t cols)`
- `template<typename Scalar >`  
`types::DynMat< Scalar > syspermute (const types::DynMat< Scalar > &A, const std::vector< size_t > perm, const std::vector< size_t > &dims)`
- `template<typename Scalar >`  
`types::DynMat< Scalar > ptrace2 (const types::DynMat< Scalar > &A, const std::vector< size_t > dims)`
- `template<typename Scalar >`  
`types::DynMat< Scalar > ptrace (const types::DynMat< Scalar > &A, const std::vector< size_t > &subsys, const std::vector< size_t > &dims)`
- `template<typename Scalar >`  
`types::DynMat< Scalar > ptranspose (const types::DynMat< Scalar > &A, const std::vector< size_t > &subsys, const std::vector< size_t > &dims)`
- `template<typename Scalar >`  
`types::DynMat< Scalar > comm (const types::DynMat< Scalar > &A, const types::DynMat< Scalar > &B)`
- `template<typename Scalar >`  
`types::DynMat< Scalar > anticomm (const types::DynMat< Scalar > &A, const types::DynMat< Scalar > &B)`
- `template<typename Scalar >`  
`types::DynMat< Scalar > proj (const types::DynMat< Scalar > &V)`

- `template<typename Scalar >`  
`types::DynMat< Scalar > expandout` (const `types::DynMat< Scalar > &A`, `size_t pos`, const `std::vector< size_t > &dims`)
- `template<typename Scalar >`  
`types::DynMat< Scalar > grams` (const `std::vector< types::DynMat< Scalar > > &vecs`)
- `template<typename Scalar >`  
`types::DynMat< Scalar > grams` (const `types::DynMat< Scalar > &A`)
- `template<typename T >`  
`void disp` (const `T &x`, const `std::string &separator=" "`, const `std::string &start="["`, const `std::string &end="]"`, `std::ostream &os=std::cout`)
- `template<typename T >`  
`void displn` (const `T &x`, const `std::string &separator=" "`, const `std::string &start="["`, const `std::string &end="]"`, `std::ostream &os=std::cout`)
- `template<typename T >`  
`void disp` (const `T *x`, const `size_t n`, const `std::string &separator=" "`, const `std::string &start="["`, const `std::string &end="]"`, `std::ostream &os=std::cout`)
- `template<typename T >`  
`void displn` (const `T *x`, const `size_t n`, const `std::string &separator=" "`, const `std::string &start="["`, const `std::string &end="]"`, `std::ostream &os=std::cout`)
- `template<typename Scalar >`  
`void disp` (const `types::DynMat< Scalar > &A`, double `chop=ct::chop`, `std::ostream &os=std::cout`)
- `template<typename Scalar >`  
`void displn` (const `types::DynMat< Scalar > &A`, double `chop=ct::chop`, `std::ostream &os=std::cout`)
- `void disp` (const `types::cplx c`, double `chop=ct::chop`, `std::ostream &os=std::cout`)
- `void displn` (const `types::cplx c`, double `chop=ct::chop`, `std::ostream &os=std::cout`)
- `template<typename Scalar >`  
`void save` (const `types::DynMat< Scalar > &A`, const `std::string &fname`)
- `template<typename Scalar >`  
`types::DynMat< Scalar > load` (const `std::string &fname`)
- `template<typename Scalar >`  
`types::DynMat< Scalar > loadMATLABmatrix` (const `std::string &mat_file`, const `std::string &var_name`)
- `template<>`  
`types::DynMat< double > loadMATLABmatrix` (const `std::string &mat_file`, const `std::string &var_name`)
- `template<>`  
`types::DynMat< types::cplx > loadMATLABmatrix` (const `std::string &mat_file`, const `std::string &var_name`)
- `template<typename Scalar >`  
`void saveMATLABmatrix` (const `types::DynMat< Scalar > &A`, const `std::string &mat_file`, const `std::string &var_name`, const `std::string &mode`)
- `template<>`  
`void saveMATLABmatrix` (const `types::DynMat< double > &A`, const `std::string &mat_file`, const `std::string &var_name`, const `std::string &mode`)
- `template<>`  
`void saveMATLABmatrix` (const `types::DynMat< types::cplx > &A`, const `std::string &mat_file`, const `std::string &var_name`, const `std::string &mode`)
- `int _init ()`
- `template<typename Scalar >`  
`types::DynMat< Scalar > rand` (`size_t rows`, `size_t cols`, double `a=0`, double `b=1`)
- `template<>`  
`types::DynMat< double > rand` (`size_t rows`, `size_t cols`, double `a`, double `b`)
- `template<>`  
`types::DynMat< types::cplx > rand` (`size_t rows`, `size_t cols`, double `a`, double `b`)
- double `rand` (double `a=0`, double `b=1`)
- `template<typename Scalar >`  
`types::DynMat< Scalar > randn` (`size_t rows`, `size_t cols`, double `mean=0`, double `sigma=1`)
- `template<>`  
`types::DynMat< double > randn` (`size_t rows`, `size_t cols`, double `mean`, double `sigma`)

- `template<>`  
`types::DynMat< types::cplx > randn` (size\_t rows, size\_t cols, double mean, double sigma)
- `double randn` (double mean=0, double sigma=1)
- `types::cmat randU` (size\_t D)
- `types::cmat randV` (size\_t Din, size\_t Dout)
- `std::vector< types::cmat > randKraus` (size\_t n, size\_t D)
- `types::cmat randH` (size\_t D)
- `types::cmat randket` (size\_t D)
- `types::cmat randrho` (size\_t D)

### 5.1.1 Function Documentation

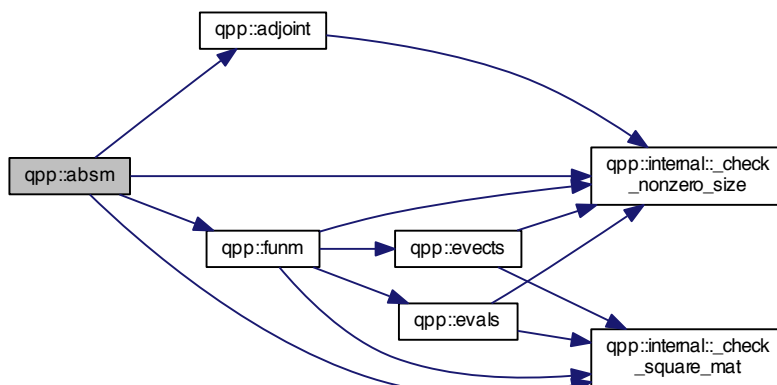
#### 5.1.1.1 `int qpp::_init ( )`

Here is the call graph for this function:



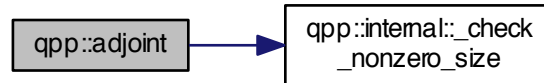
#### 5.1.1.2 `template<typename Scalar > types::cmat qpp::absm ( const types::DynMat< Scalar > & A )`

Here is the call graph for this function:



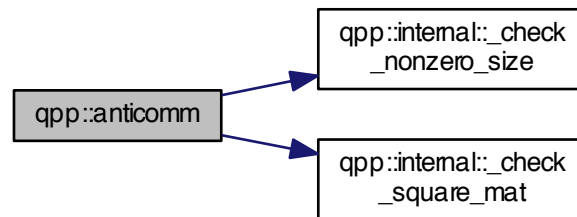
5.1.1.3 `template<typename Scalar > types::DynMat<Scalar> qpp::adjoint ( const types::DynMat< Scalar > & A )`

Here is the call graph for this function:



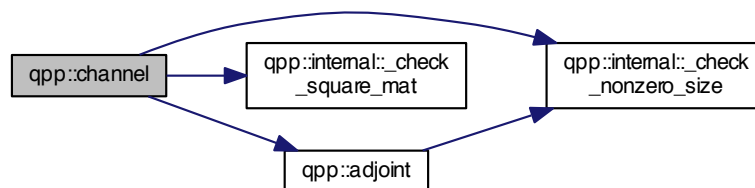
5.1.1.4 `template<typename Scalar > types::DynMat<Scalar> qpp::anticomm ( const types::DynMat< Scalar > & A, const types::DynMat< Scalar > & B )`

Here is the call graph for this function:



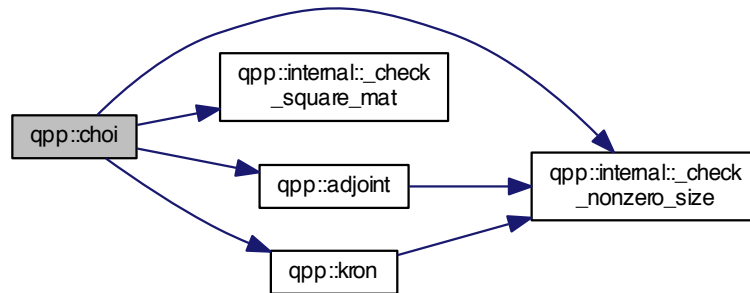
5.1.1.5 `types::cmat qpp::channel ( const types::cmat & rho, const std::vector< types::cmat > & Ks )`

Here is the call graph for this function:



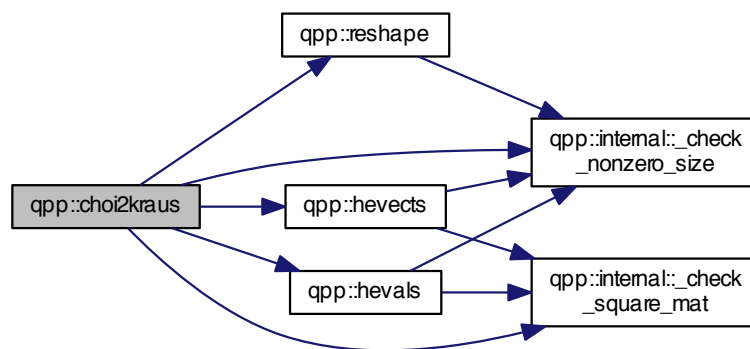
#### 5.1.1.6 `types::cmat qpp::choi ( const std::vector< types::cmat > & Ks )`

Here is the call graph for this function:



#### 5.1.1.7 `std::vector<types::cmat> qpp::choi2kraus ( const types::cmat & A )`

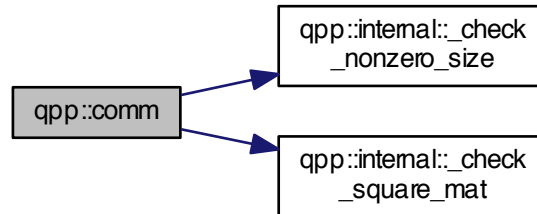
Here is the call graph for this function:





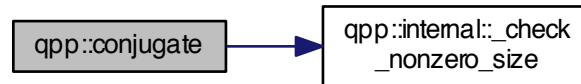
5.1.1.8 `template<typename Scalar > types::DynMat<Scalar> qpp::comm ( const types::DynMat< Scalar > & A, const types::DynMat< Scalar > & B )`

Here is the call graph for this function:



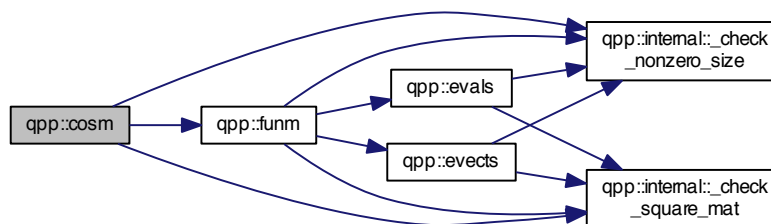
5.1.1.9 `template<typename Scalar > types::DynMat<Scalar> qpp::conjugate ( const types::DynMat< Scalar > & A )`

Here is the call graph for this function:



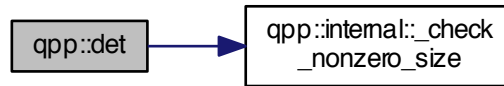
5.1.1.10 `template<typename Scalar > types::cmat qpp::cosm ( const types::DynMat< Scalar > & A )`

Here is the call graph for this function:



5.1.1.11 `template<typename Scalar > Scalar qpp::det ( const types::DynMat< Scalar > & A )`

Here is the call graph for this function:



5.1.1.12 `template<typename T > void qpp::disp ( const T & x, const std::string & separator = " ", const std::string & start = "[", const std::string & end = "]" , std::ostream & os = std::cout )`

5.1.1.13 `template<typename T > void qpp::disp ( const T * x, const size_t n, const std::string & separator = " ", const std::string & start = "[", const std::string & end = "]" , std::ostream & os = std::cout )`

5.1.1.14 `template<typename Scalar > void qpp::disp ( const types::DynMat< Scalar > & A, double chop = ct::chop, std::ostream & os = std::cout )`

5.1.1.15 `void qpp::disp ( const types::cplx c, double chop = ct::chop, std::ostream & os = std::cout )`

Here is the call graph for this function:



5.1.1.16 `template<typename T > void qpp::displn ( const T & x, const std::string & separator = " ", const std::string & start = "[", const std::string & end = "]" , std::ostream & os = std::cout )`

Here is the call graph for this function:



5.1.1.17 `template<typename T> void qpp::displn ( const T * x, const size_t n, const std::string & separator = " ", const std::string & start = " [", const std::string & end = " ] ", std::ostream & os = std::cout )`

Here is the call graph for this function:



5.1.1.18 `template<typename Scalar> void qpp::displn ( const types::DynMat< Scalar > & A, double chop = ct::chop, std::ostream & os = std::cout )`

Here is the call graph for this function:



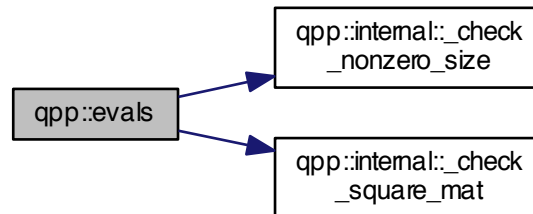
5.1.1.19 `void qpp::displn ( const types::cplx c, double chop = ct::chop, std::ostream & os = std::cout )`

Here is the call graph for this function:



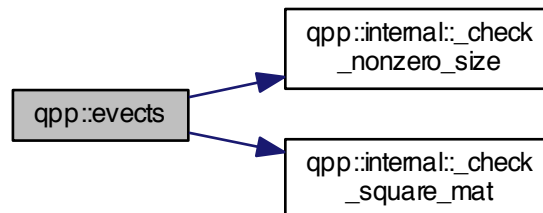
5.1.1.20 `template<typename Scalar > types::cmat qpp::evals ( const types::DynMat< Scalar > & A )`

Here is the call graph for this function:



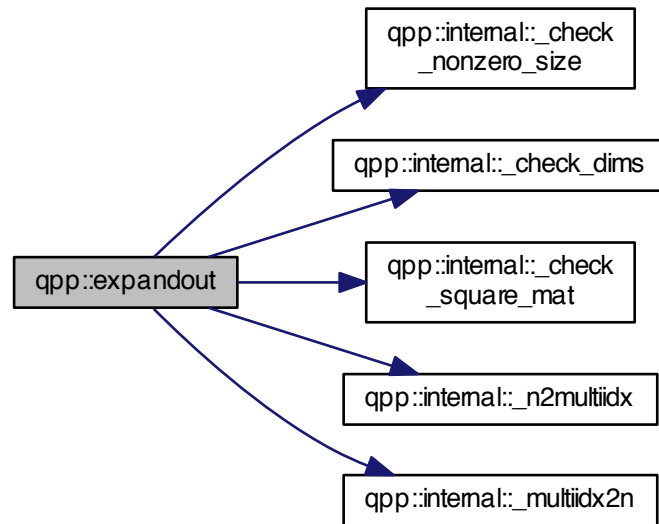
5.1.1.21 `template<typename Scalar > types::cmat qpp::evecs ( const types::DynMat< Scalar > & A )`

Here is the call graph for this function:



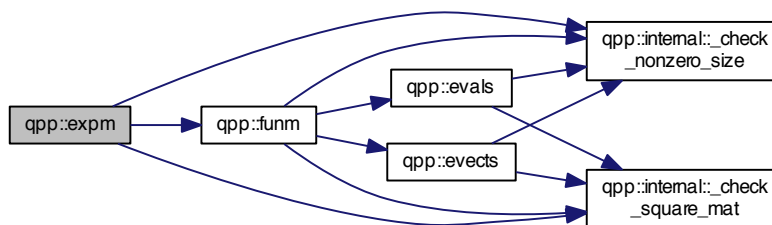
5.1.1.22 `template<typename Scalar > types::DynMat<Scalar> qpp::expandout ( const types::DynMat< Scalar > & A, size_t pos, const std::vector< size_t > & dims )`

Here is the call graph for this function:



5.1.1.23 `template<typename Scalar > types::cmat qpp::expm ( const types::DynMat< Scalar > & A )`

Here is the call graph for this function:



5.1.1.24 `template<typename InputScalar , typename OutputScalar > types::DynMat<OutputScalar> qpp::fun ( const types::DynMat< InputScalar > & A, OutputScalar(*) (const InputScalar &) f )`

Here is the call graph for this function:



5.1.1.25 `template<typename Scalar > types::cmat qpp::funm ( const types::DynMat< Scalar > & A, types::cplx(*) (const types::cplx &) f )`

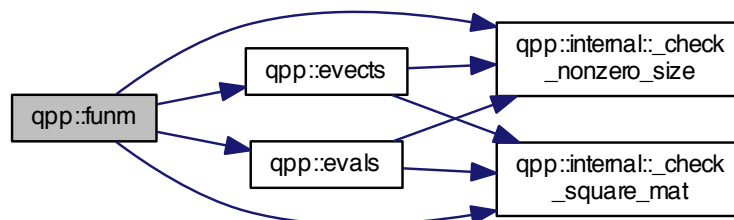
Parameters

<i>A</i>	input matrix
<i>f</i>	function pointer

Returns

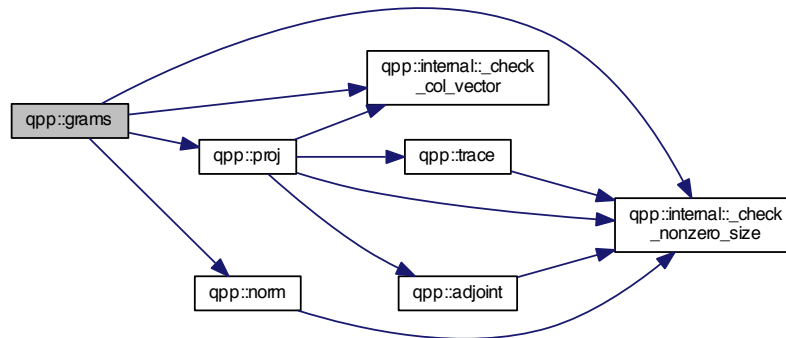
`types::cmat`

Here is the call graph for this function:



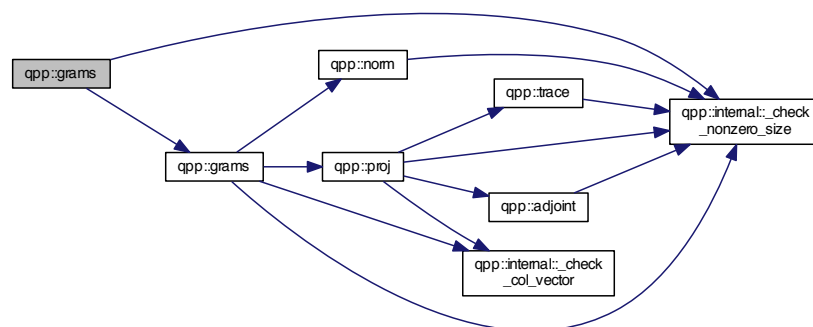
5.1.1.26 `template<typename Scalar> types::DynMat<Scalar> qpp::grams ( const std::vector< types::DynMat< Scalar >> & vecs )`

Here is the call graph for this function:



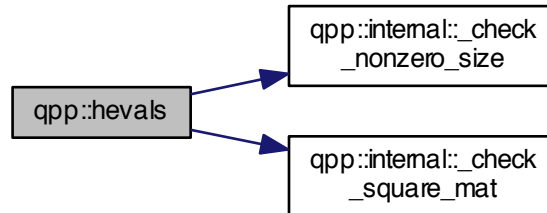
5.1.1.27 `template<typename Scalar> types::DynMat<Scalar> qpp::grams ( const types::DynMat< Scalar > & A )`

Here is the call graph for this function:



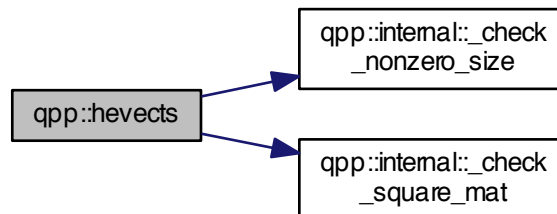
5.1.1.28 `template<typename Scalar > types::cmat qpp::hevals ( const types::DynMat< Scalar > & A )`

Here is the call graph for this function:



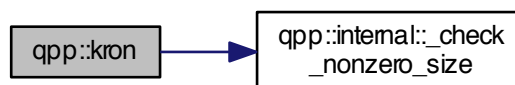
5.1.1.29 `template<typename Scalar > types::cmat qpp::hevects ( const types::DynMat< Scalar > & A )`

Here is the call graph for this function:



5.1.1.30 `template<typename Scalar > types::DynMat<Scalar> qpp::kron ( const types::DynMat< Scalar > & A, const types::DynMat< Scalar > & B )`

Here is the call graph for this function:





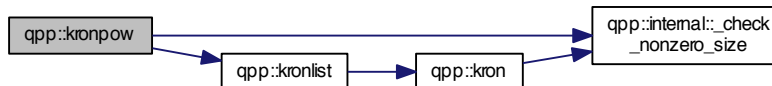
5.1.1.31 `template<typename Scalar > types::DynMat<Scalar> qpp::kronlist ( const std::vector< types::DynMat< Scalar >> & list )`

Here is the call graph for this function:



5.1.1.32 `template<typename Scalar > types::DynMat<Scalar> qpp::kronpow ( const types::DynMat< Scalar > & A, size_t n )`

Here is the call graph for this function:



5.1.1.33 `template<typename Scalar > types::DynMat<Scalar> qpp::load ( const std::string & fname )`

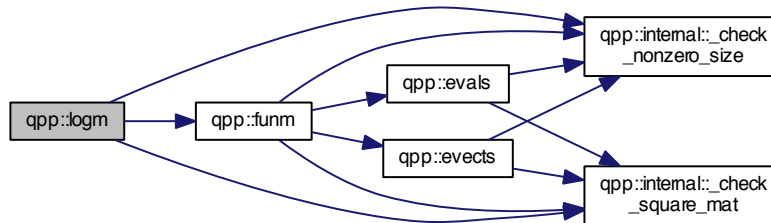
5.1.1.34 `template<typename Scalar > types::DynMat<Scalar> qpp::loadMATLABmatrix ( const std::string & mat_file, const std::string & var_name )`

5.1.1.35 `template<> types::DynMat<double> qpp::loadMATLABmatrix ( const std::string & mat_file, const std::string & var_name )`

5.1.1.36 `template<> types::DynMat<types::cplx> qpp::loadMATLABmatrix ( const std::string & mat_file, const std::string & var_name )`

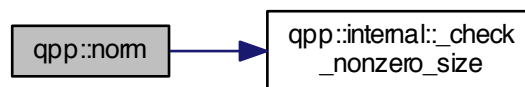
5.1.1.37 `template<typename Scalar > types::cmat qpp::logm ( const types::DynMat< Scalar > & A )`

Here is the call graph for this function:



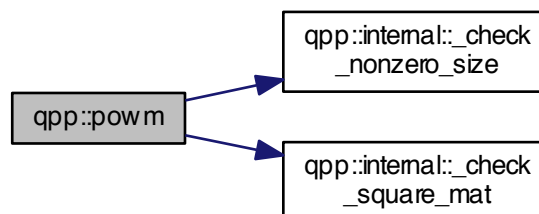
5.1.1.38 `template<typename Scalar > double qpp::norm ( const types::DynMat< Scalar > & A )`

Here is the call graph for this function:



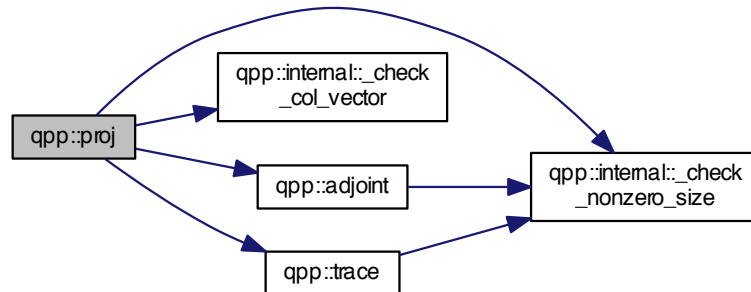
5.1.1.39 `template<typename Scalar > types::DynMat<Scalar> qpp::powm ( const types::DynMat< Scalar > & A, size_t n )`

Here is the call graph for this function:



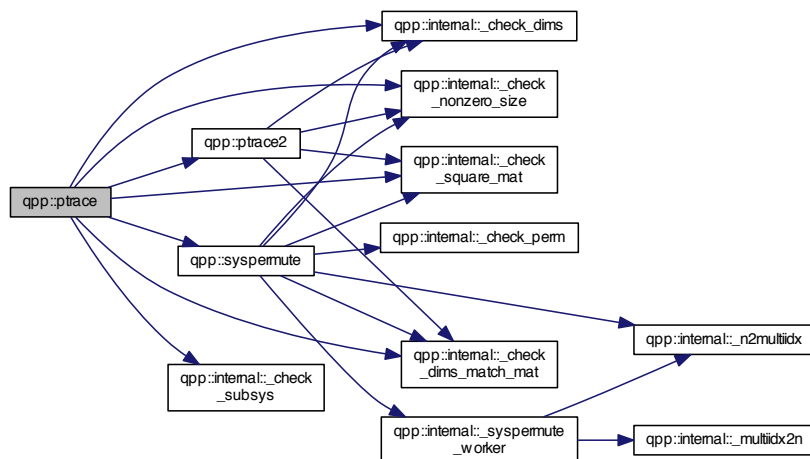
5.1.1.40 `template<typename Scalar> types::DynMat<Scalar> qpp::proj ( const types::DynMat< Scalar> & V )`

Here is the call graph for this function:



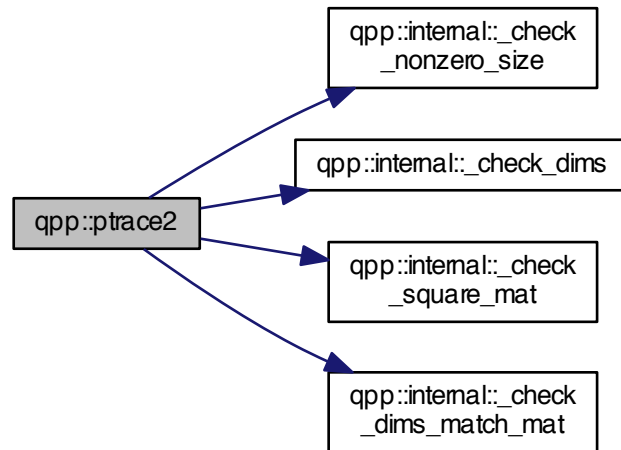
5.1.1.41 `template<typename Scalar> types::DynMat<Scalar> qpp::ptrace ( const types::DynMat< Scalar> & A, const std::vector< size_t> & subsys, const std::vector< size_t> & dims )`

Here is the call graph for this function:



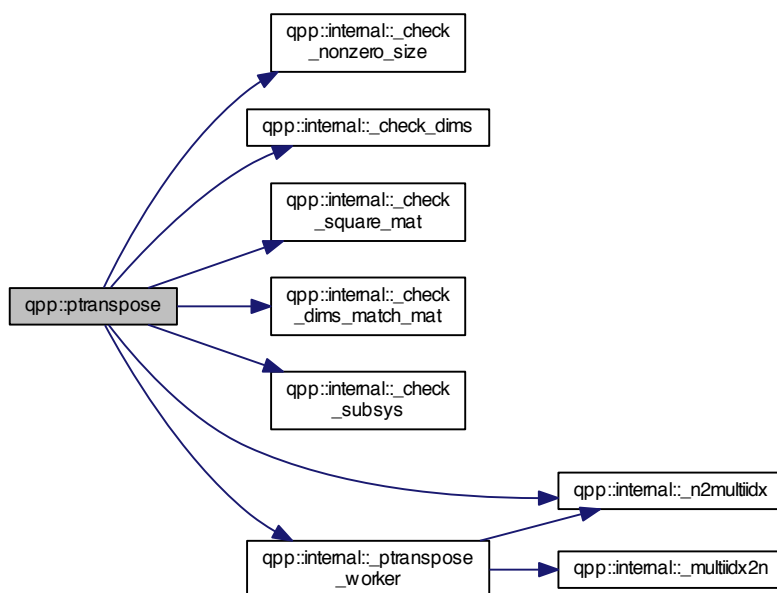
5.1.1.42 `template<typename Scalar > types::DynMat<Scalar> qpp::ptrace2 ( const types::DynMat< Scalar > & A, const std::vector< size_t > dims )`

Here is the call graph for this function:



5.1.1.43 `template<typename Scalar > types::DynMat<Scalar> qpp::ptranspose ( const types::DynMat< Scalar > & A, const std::vector< size_t > & subsys, const std::vector< size_t > & dims )`

Here is the call graph for this function:



5.1.1.44 `template<typename Scalar > types::DynMat<Scalar> qpp::rand ( size_t rows, size_t cols, double a = 0, double b = 1 )`

5.1.1.45 `template<> types::DynMat<double> qpp::rand ( size_t rows, size_t cols, double a, double b )`

5.1.1.46 `template<> types::DynMat<types::cplx> qpp::rand ( size_t rows, size_t cols, double a, double b )`

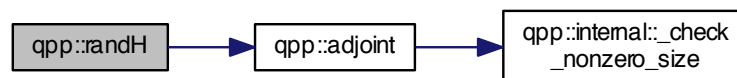
5.1.1.47 `double qpp::rand ( double a = 0, double b = 1 )`

Here is the call graph for this function:



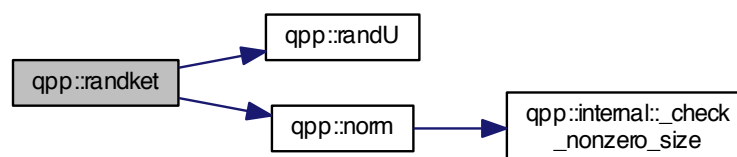
5.1.1.48 `types::cmat qpp::randH ( size_t D )`

Here is the call graph for this function:



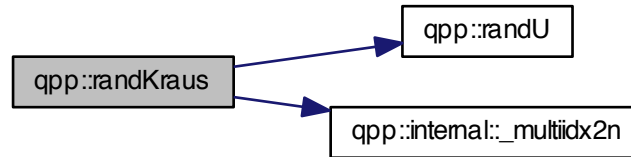
5.1.1.49 `types::cmat qpp::randket ( size_t D )`

Here is the call graph for this function:



5.1.1.50 `std::vector<types::cmat> qpp::randKraus ( size_t n, size_t D )`

Here is the call graph for this function:



5.1.1.51 `template<typename Scalar > types::DynMat<Scalar> qpp::randn ( size_t rows, size_t cols, double mean = 0, double sigma = 1 )`

5.1.1.52 `template<> types::DynMat<double> qpp::randn ( size_t rows, size_t cols, double mean, double sigma )`

Here is the call graph for this function:



5.1.1.53 `template<> types::DynMat<types::cplx> qpp::randn ( size_t rows, size_t cols, double mean, double sigma )`

Here is the call graph for this function:



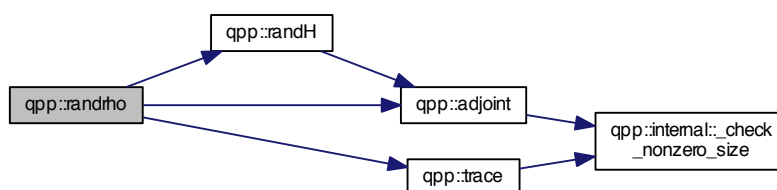
5.1.1.54 `double qpp::randn ( double mean = 0, double sigma = 1 )`

Here is the call graph for this function:



5.1.1.55 `types::cmat qpp::randrho ( size_t D )`

Here is the call graph for this function:



5.1.1.56 `types::cmat qpp::randU ( size_t D )`

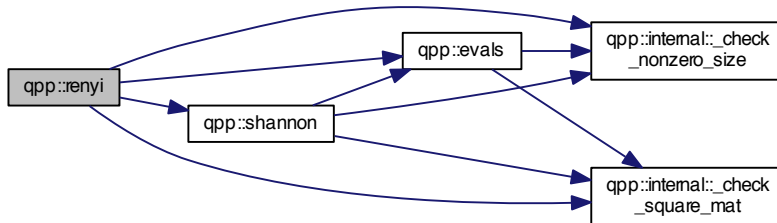
5.1.1.57 `types::cmat qpp::randV ( size_t Din, size_t Dout )`

Here is the call graph for this function:



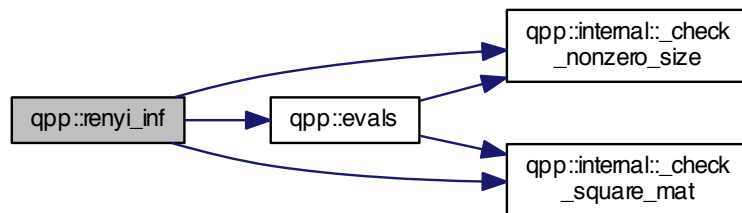
5.1.1.58 `template<typename Scalar > double qpp::renyi ( const double alpha, const types::DynMat< Scalar > & A )`

Here is the call graph for this function:



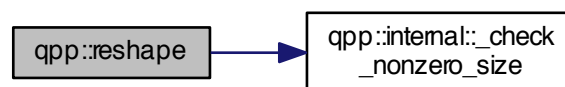
5.1.1.59 `template<typename Scalar > double qpp::renyi_inf ( const types::DynMat< Scalar > & A )`

Here is the call graph for this function:



5.1.1.60 `template<typename Scalar > types::DynMat<Scalar> qpp::reshape ( const types::DynMat< Scalar > & A, size_t rows, size_t cols )`

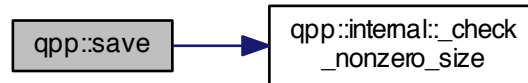
Here is the call graph for this function:





5.1.1.61 `template<typename Scalar > void qpp::save ( const types::DynMat< Scalar > & A, const std::string & fname )`

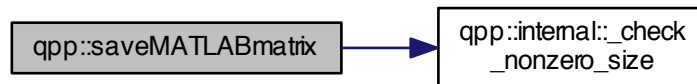
Here is the call graph for this function:



5.1.1.62 `template<typename Scalar > void qpp::saveMATLABmatrix ( const types::DynMat< Scalar > & A, const std::string & mat_file, const std::string & var_name, const std::string & mode )`

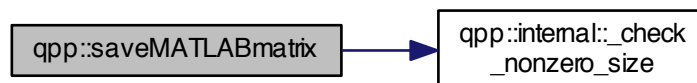
5.1.1.63 `template<> void qpp::saveMATLABmatrix ( const types::DynMat< double > & A, const std::string & mat_file, const std::string & var_name, const std::string & mode )`

Here is the call graph for this function:



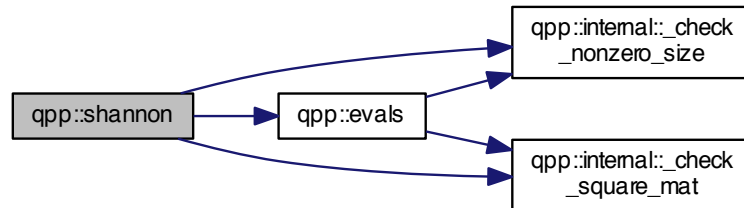
5.1.1.64 `template<> void qpp::saveMATLABmatrix ( const types::DynMat< types::cplx > & A, const std::string & mat_file, const std::string & var_name, const std::string & mode )`

Here is the call graph for this function:



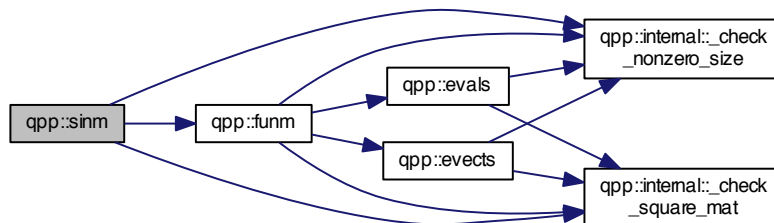
5.1.1.65 `template<typename Scalar > double qpp::shannon ( const types::DynMat< Scalar > & A )`

Here is the call graph for this function:



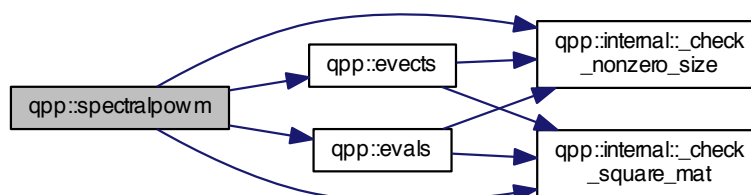
5.1.1.66 `template<typename Scalar > types::cmat qpp::sinm ( const types::DynMat< Scalar > & A )`

Here is the call graph for this function:



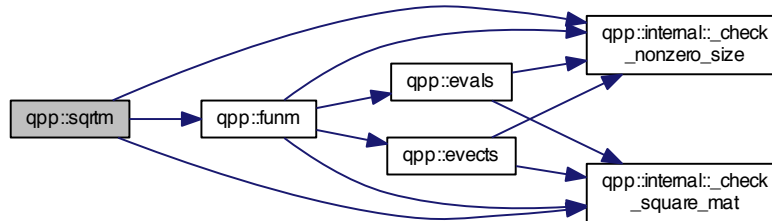
5.1.1.67 `template<typename Scalar > types::cmat qpp::spectralpowm ( const types::DynMat< Scalar > & A, const types::cplx z )`

Here is the call graph for this function:



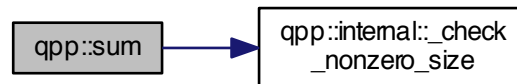
5.1.1.68 `template<typename Scalar > types::cmat qpp::sqrtm ( const types::DynMat< Scalar > & A )`

Here is the call graph for this function:



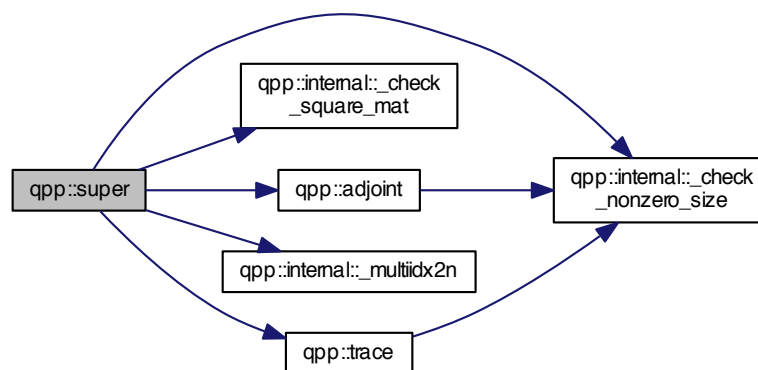
5.1.1.69 `template<typename Scalar > Scalar qpp::sum ( const types::DynMat< Scalar > & A )`

Here is the call graph for this function:



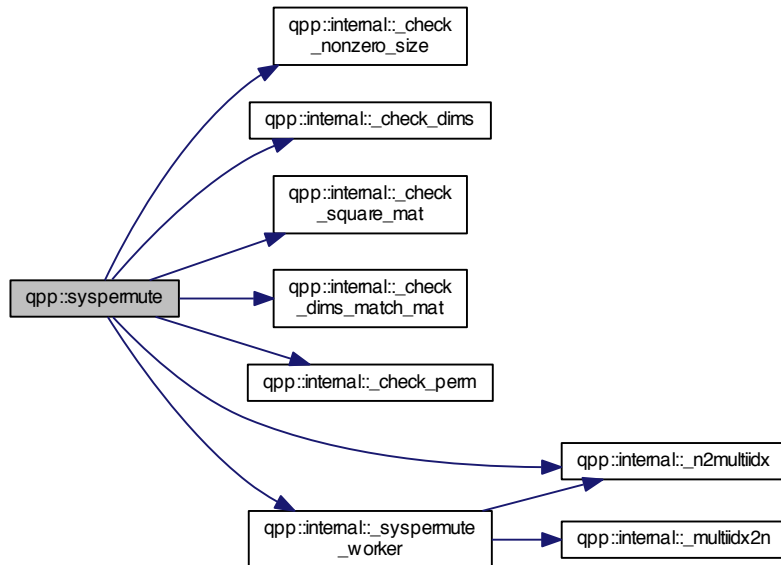
5.1.1.70 `types::cmat qpp::super ( const std::vector< types::cmat > & Ks )`

Here is the call graph for this function:



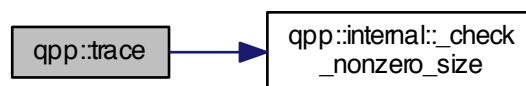
5.1.1.71 `template<typename Scalar > types::DynMat<Scalar> qpp::syspermute ( const types::DynMat< Scalar > & A,  
const std::vector< size_t > perm, const std::vector< size_t > & dims )`

Here is the call graph for this function:



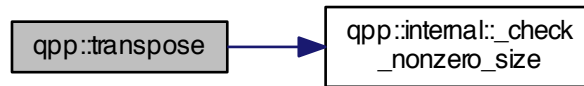
5.1.1.72 `template<typename Scalar > Scalar qpp::trace ( const types::DynMat< Scalar > & A )`

Here is the call graph for this function:



5.1.1.73 `template<typename Scalar > types::DynMat<Scalar> qpp::transpose ( const types::DynMat< Scalar > & A )`

Here is the call graph for this function:



## 5.2 qpp::ct Namespace Reference

### Functions

- `std::complex< double > omega (size_t D)`

### Variables

- `const double chop = 1e-10`
- `const double eps = 1e-14`
- `const std::complex< double > ii = { 0, 1 }`
- `const double pi = 3.141592653589793238462643383279502884`
- `const double ee = 2.718281828459045235360287471352662497`

### 5.2.1 Function Documentation

5.2.1.1 `std::complex<double> qpp::ct::omega ( size_t D )`

### 5.2.2 Variable Documentation

5.2.2.1 `const double qpp::ct::chop = 1e-10`

5.2.2.2 `const double qpp::ct::ee = 2.718281828459045235360287471352662497`

5.2.2.3 `const double qpp::ct::eps = 1e-14`

5.2.2.4 `const std::complex<double> qpp::ct::ii = { 0, 1 }`

5.2.2.5 `const double qpp::ct::pi = 3.141592653589793238462643383279502884`

## 5.3 qpp::gt Namespace Reference

### Functions

- `void _init_gates ()`
- `types::cmat Rtheta (double theta)`
- `types::cmat Id (size_t D)`

- [types::cmat Zd](#) (size\_t D)
- [types::cmat Fd](#) (size\_t D)
- [types::cmat Xd](#) (size\_t D)
- [types::cmat CTRL](#) (const [types::cmat](#) &A, const std::vector< size\_t > &ctrl, const std::vector< size\_t > &gate, size\_t n, size\_t D=2)

## Variables

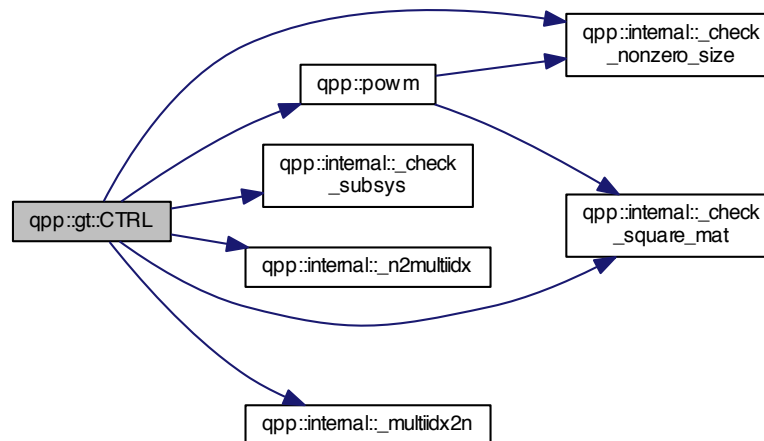
- [types::cmat Id2](#)
- [types::cmat H](#)
- [types::cmat X](#)
- [types::cmat Y](#)
- [types::cmat Z](#)
- [types::cmat S](#)
- [types::cmat T](#)
- [types::cmat CNOTab](#)
- [types::cmat CNOTba](#)
- [types::cmat CZ](#)
- [types::cmat CS](#)
- [types::cmat SWAP](#)
- [types::cmat TOF](#)
- [types::cmat FRED](#)
- [types::cmat x0](#)
- [types::cmat x1](#)
- [types::cmat y0](#)
- [types::cmat y1](#)
- [types::cmat z0](#)
- [types::cmat z1](#)
- [types::cmat b00](#)
- [types::cmat b01](#)
- [types::cmat b10](#)
- [types::cmat b11](#)

## 5.3.1 Function Documentation

### 5.3.1.1 void qpp::gt::\_init\_gates ( ) `[inline]`

5.3.1.2 `types::cmat qpp::gt::CTRL ( const types::cmat & A, const std::vector< size_t > & ctrl, const std::vector< size_t > & gate, size_t n, size_t D = 2 ) [inline]`

Here is the call graph for this function:



5.3.1.3 `types::cmat qpp::gt::Fd ( size_t D ) [inline]`

Here is the call graph for this function:



5.3.1.4 `types::cmat qpp::gt::ld ( size_t D ) [inline]`

5.3.1.5 `types::cmat qpp::gt::Rtheta ( double theta ) [inline]`

#### 5.3.1.6 `types::cmat qpp::gt::Xd ( size_t D ) [inline]`

Here is the call graph for this function:



#### 5.3.1.7 `types::cmat qpp::gt::Zd ( size_t D ) [inline]`

Here is the call graph for this function:



### 5.3.2 Variable Documentation

#### 5.3.2.1 `types::cmat qpp::gt::b00`

#### 5.3.2.2 `types::cmat qpp::gt::b01`

#### 5.3.2.3 `types::cmat qpp::gt::b10`

#### 5.3.2.4 `types::cmat qpp::gt::b11`

#### 5.3.2.5 `types::cmat qpp::gt::CNOTab`

#### 5.3.2.6 `types::cmat qpp::gt::CNOTba`

#### 5.3.2.7 `types::cmat qpp::gt::CS`

#### 5.3.2.8 `types::cmat qpp::gt::CZ`

#### 5.3.2.9 `types::cmat qpp::gt::FRED`

#### 5.3.2.10 `types::cmat qpp::gt::H`

#### 5.3.2.11 `types::cmat qpp::gt::Id2`



5.3.2.12 `types::cmat qpp::gt::S`

5.3.2.13 `types::cmat qpp::gt::SWAP`

5.3.2.14 `types::cmat qpp::gt::T`

5.3.2.15 `types::cmat qpp::gt::TOF`

5.3.2.16 `types::cmat qpp::gt::X`

5.3.2.17 `types::cmat qpp::gt::x0`

5.3.2.18 `types::cmat qpp::gt::x1`

5.3.2.19 `types::cmat qpp::gt::Y`

5.3.2.20 `types::cmat qpp::gt::y0`

5.3.2.21 `types::cmat qpp::gt::y1`

5.3.2.22 `types::cmat qpp::gt::Z`

5.3.2.23 `types::cmat qpp::gt::z0`

5.3.2.24 `types::cmat qpp::gt::z1`

## 5.4 qpp::internal Namespace Reference

### Functions

- `void _n2multiidx (size_t n, size_t numdims, const size_t *dims, size_t *result)`
- `size_t _multiidx2n (const size_t *midx, size_t numdims, const size_t *dims)`
- `template<typename Scalar >`  
`bool _check_square_mat (const types::DynMat< Scalar > &A)`
- `template<typename Scalar >`  
`bool _check_vector (const types::DynMat< Scalar > &A)`
- `template<typename Scalar >`  
`bool _check_row_vector (const types::DynMat< Scalar > &A)`
- `template<typename Scalar >`  
`bool _check_col_vector (const types::DynMat< Scalar > &A)`
- `template<typename T >`  
`bool _check_nonzero_size (const T &x)`
- `bool _check_dims (const std::vector< size_t > &dims)`
- `template<typename Scalar >`  
`bool _check_dims_match_mat (const std::vector< size_t > &dims, const types::DynMat< Scalar > &A)`
- `bool _check_eq_dims (const std::vector< size_t > &dims, size_t dim)`
- `bool _check_subsys (const std::vector< size_t > &subsys, const std::vector< size_t > &dims)`
- `bool _check_perm (const std::vector< size_t > &perm, const std::vector< size_t > &dims)`
- `template<typename Scalar >`  
`void _syspermute_worker (const size_t *midxcol, size_t numdims, const size_t *cdims, const size_t *cperm, size_t i, size_t j, size_t &iperm, size_t &jperm, const types::DynMat< Scalar > &A, types::DynMat< Scalar > &result)`
- `template<typename Scalar >`  
`void _ptranspose_worker (const size_t *midxcol, size_t numdims, size_t numsubsys, const size_t *cdims, const size_t *csubsys, size_t i, size_t j, size_t &iperm, size_t &jperm, const types::DynMat< Scalar > &A, types::DynMat< Scalar > &result)`

### 5.4.1 Function Documentation

5.4.1.1 `template<typename Scalar > bool qpp::internal::_check_col_vector ( const types::DynMat< Scalar > & A )`

5.4.1.2 `bool qpp::internal::_check_dims ( const std::vector< size_t > & dims )`

5.4.1.3 `template<typename Scalar > bool qpp::internal::_check_dims_match_mat ( const std::vector< size_t > & dims, const types::DynMat< Scalar > & A )`

5.4.1.4 `bool qpp::internal::_check_eq_dims ( const std::vector< size_t > & dims, size_t dim )`

5.4.1.5 `template<typename T > bool qpp::internal::_check_nonzero_size ( const T & x )`

5.4.1.6 `bool qpp::internal::_check_perm ( const std::vector< size_t > & perm, const std::vector< size_t > & dims )`

5.4.1.7 `template<typename Scalar > bool qpp::internal::_check_row_vector ( const types::DynMat< Scalar > & A )`

5.4.1.8 `template<typename Scalar > bool qpp::internal::_check_square_mat ( const types::DynMat< Scalar > & A )`

5.4.1.9 `bool qpp::internal::_check_subsys ( const std::vector< size_t > & subsys, const std::vector< size_t > & dims )`

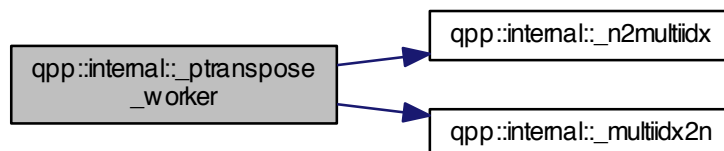
5.4.1.10 `template<typename Scalar > bool qpp::internal::_check_vector ( const types::DynMat< Scalar > & A )`

5.4.1.11 `size_t qpp::internal::_multiidx2n ( const size_t * midx, size_t numdims, const size_t * dims )`

5.4.1.12 `void qpp::internal::_n2multiidx ( size_t n, size_t numdims, const size_t * dims, size_t * result )`

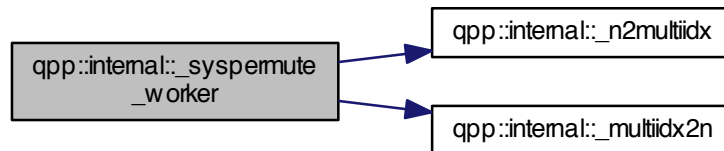
5.4.1.13 `template<typename Scalar > void qpp::internal::_ptranspose_worker ( const size_t * midxcol, size_t numdims, size_t numsubsys, const size_t * cdims, const size_t * csubsys, size_t i, size_t j, size_t iperm, size_t jperm, const types::DynMat< Scalar > & A, types::DynMat< Scalar > & result )`

Here is the call graph for this function:



5.4.1.14 `template<typename Scalar > void qpp::internal::_syspermute_worker ( const size_t * midxcol, size_t numdims, const size_t * cdims, const size_t * cperm, size_t i, size_t j, size_t & iperm, size_t & jperm, const types::DynMat< Scalar > & A, types::DynMat< Scalar > & result )`

Here is the call graph for this function:



## 5.5 qpp::stat Namespace Reference

### Classes

- class [NormalDistribution](#)
- class [UniformRealDistribution](#)
- class [DiscreteDistribution](#)
- class [DiscreteDistributionFromComplex](#)

### Variables

- `std::random_device _rd`
- `std::mt19937 _rng`

### 5.5.1 Variable Documentation

5.5.1.1 `std::random_device qpp::stat::_rd`

5.5.1.2 `std::mt19937 qpp::stat::_rng`

## 5.6 qpp::types Namespace Reference

### Typedefs

- `typedef std::complex< double > cplx`
- `typedef Eigen::MatrixXcd cmat`
- `typedef Eigen::MatrixXd dmat`
- `typedef Eigen::MatrixXf fmat`
- `typedef Eigen::MatrixXi imat`
- `template<typename Expression > using Expression2DynMat = Eigen::Matrix< typename Expression::Scalar, Eigen::Dynamic, Eigen::Dynamic >`
- `template<typename Scalar > using DynMat = Eigen::Matrix< Scalar, Eigen::Dynamic, Eigen::Dynamic >`

## Functions

- `int myfunc (int a, int b)`

### 5.6.1 Typedef Documentation

5.6.1.1 `typedef Eigen::MatrixXcd qpp::types::cmat`

5.6.1.2 `typedef std::complex<double> qpp::types::cplx`

5.6.1.3 `typedef Eigen::MatrixXd qpp::types::dmat`

5.6.1.4 `template<typename Scalar > using qpp::types::DynMat = typedef Eigen::Matrix<Scalar, Eigen::Dynamic, Eigen::Dynamic>`

5.6.1.5 `template<typename Expression > using qpp::types::Expression2DynMat = typedef Eigen::Matrix<typename Expression::Scalar, Eigen::Dynamic, Eigen::Dynamic>`

5.6.1.6 `typedef Eigen::MatrixXf qpp::types::fmat`

5.6.1.7 `typedef Eigen::MatrixXi qpp::types::imat`

### 5.6.2 Function Documentation

5.6.2.1 `int qpp::types::myfunc ( int a, int b )`

## Chapter 6

# Class Documentation

### 6.1 qpp::stat::DiscreteDistribution Class Reference

```
#include <stat.h>
```

#### Public Member Functions

- `template<typename InputIterator >`  
`DiscreteDistribution` (`InputIterator first`, `InputIterator last`)
- `DiscreteDistribution` (`std::initializer_list< double > weights`)
- `DiscreteDistribution` (`std::vector< double > weights`)
- `size_t sample` ()
- `std::vector< double > probabilities` ()

#### Protected Attributes

- `std::discrete_distribution`  
`< size_t > _d`

#### 6.1.1 Constructor & Destructor Documentation

6.1.1.1 `template<typename InputIterator > qpp::stat::DiscreteDistribution::DiscreteDistribution ( InputIterator first, InputIterator last )` `[inline]`

6.1.1.2 `qpp::stat::DiscreteDistribution::DiscreteDistribution ( std::initializer_list< double > weights )` `[inline]`

6.1.1.3 `qpp::stat::DiscreteDistribution::DiscreteDistribution ( std::vector< double > weights )` `[inline]`

#### 6.1.2 Member Function Documentation

6.1.2.1 `std::vector<double> qpp::stat::DiscreteDistribution::probabilities ( )` `[inline]`

6.1.2.2 `size_t qpp::stat::DiscreteDistribution::sample ( )` `[inline]`

#### 6.1.3 Member Data Documentation

6.1.3.1 `std::discrete_distribution<size_t> qpp::stat::DiscreteDistribution::_d` `[protected]`

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

- [include/stat.h](#)

## 6.2 qpp::stat::DiscreteDistributionFromComplex Class Reference

```
#include <stat.h>
```

### Public Member Functions

- `template<typename InputIterator >`  
[DiscreteDistributionFromComplex](#) (InputIterator first, InputIterator last)
- [DiscreteDistributionFromComplex](#) (std::initializer\_list< [types::cplx](#) > amplitudes)
- [DiscreteDistributionFromComplex](#) (std::vector< [types::cplx](#) > amplitudes)
- [DiscreteDistributionFromComplex](#) (const [types::cmat](#) &V)
- `size_t` [sample](#) ()
- `std::vector< double >` [probabilities](#) ()

### Protected Member Functions

- `template<typename InputIterator >`  
`std::vector< double >` [cplx2amplitudes](#) (InputIterator first, InputIterator last)

### Protected Attributes

- `std::discrete_distribution`  
`< size_t >` [\\_d](#)

### 6.2.1 Constructor & Destructor Documentation

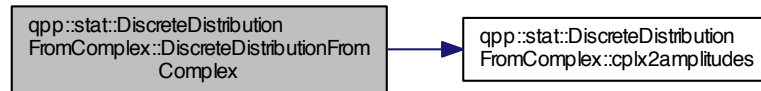
- 6.2.1.1 `template<typename InputIterator > qpp::stat::DiscreteDistributionFromComplex::DiscreteDistributionFromComplex (`  
`InputIterator first, InputIterator last )` `[inline]`

Here is the call graph for this function:



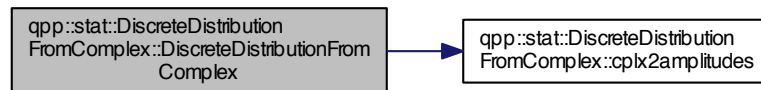
6.2.1.2 `qpp::stat::DiscreteDistributionFromComplex::DiscreteDistributionFromComplex ( std::initializer_list< types::cplx > amplitudes ) [inline]`

Here is the call graph for this function:



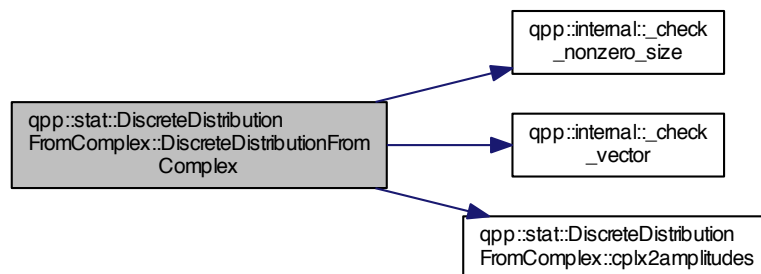
6.2.1.3 `qpp::stat::DiscreteDistributionFromComplex::DiscreteDistributionFromComplex ( std::vector< types::cplx > amplitudes ) [inline]`

Here is the call graph for this function:



6.2.1.4 `qpp::stat::DiscreteDistributionFromComplex::DiscreteDistributionFromComplex ( const types::cmat & V ) [inline]`

Here is the call graph for this function:



## 6.2.2 Member Function Documentation

6.2.2.1 `template<typename InputIterator > std::vector<double> qpp::stat::DiscreteDistributionFromComplex::cplx2amplitudes ( InputIterator first, InputIterator last ) [inline], [protected]`

6.2.2.2 `std::vector<double> qpp::stat::DiscreteDistributionFromComplex::probabilities ( ) [inline]`

6.2.2.3 `size_t qpp::stat::DiscreteDistributionFromComplex::sample ( ) [inline]`

### 6.2.3 Member Data Documentation

6.2.3.1 `std::discrete_distribution<size_t> qpp::stat::DiscreteDistributionFromComplex::_d [protected]`

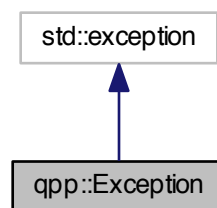
The documentation for this class was generated from the following file:

- [include/stat.h](#)

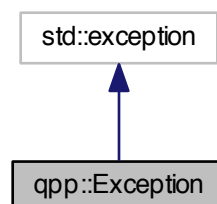
## 6.3 qpp::Exception Class Reference

```
#include <exception.h>
```

Inheritance diagram for qpp::Exception:



Collaboration diagram for qpp::Exception:





## Public Types

- enum [Type](#) {  
[Type::UNKNOWN\\_EXCEPTION](#) = 0, [Type::ZERO\\_SIZE](#), [Type::MATRIX\\_NOT\\_SQUARE](#), [Type::MATRIX\\_NOT\\_CVECTOR](#),  
[Type::MATRIX\\_NOT\\_RVECTOR](#), [Type::MATRIX\\_NOT\\_VECTOR](#), [Type::DIMS\\_INVALID](#), [Type::DIMS\\_NOT\\_EQUAL](#),  
[Type::DIMS\\_MISMATCH\\_MATRIX](#), [Type::SUBSYS\\_MISMATCH\\_DIMS](#), [Type::PERM\\_MISMATCH\\_DIMS](#),  
[Type::NOT\\_QUBIT\\_GATE](#),  
[Type::NOT\\_QUBIT\\_SUBSYS](#), [Type::OUT\\_OF\\_RANGE](#), [Type::UNDEFINED\\_TYPE](#), [Type::CUSTOM\\_EXCEPTION](#) }

## Public Member Functions

- [Exception](#) (const std::string &where, const [Type](#) &type)
- [Exception](#) (const std::string &where, const std::string &custom)
- virtual const char \* [what](#) () const noexcept override
- virtual [~Exception](#) () noexcept

## Private Member Functions

- std::string [\\_construct\\_exception\\_msg](#) ()

## Private Attributes

- std::string [\\_where](#)
- std::string [\\_msg](#)
- [Type](#) [\\_type](#)
- std::string [\\_custom](#)

### 6.3.1 Member Enumeration Documentation

#### 6.3.1.1 enum qpp::Exception::Type [strong]

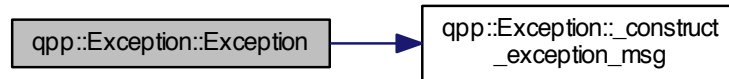
Enumerator

***UNKNOWN\_EXCEPTION***  
***ZERO\_SIZE***  
***MATRIX\_NOT\_SQUARE***  
***MATRIX\_NOT\_CVECTOR***  
***MATRIX\_NOT\_RVECTOR***  
***MATRIX\_NOT\_VECTOR***  
***DIMS\_INVALID***  
***DIMS\_NOT\_EQUAL***  
***DIMS\_MISMATCH\_MATRIX***  
***SUBSYS\_MISMATCH\_DIMS***  
***PERM\_MISMATCH\_DIMS***  
***NOT\_QUBIT\_GATE***  
***NOT\_QUBIT\_SUBSYS***  
***OUT\_OF\_RANGE***  
***UNDEFINED\_TYPE***  
***CUSTOM\_EXCEPTION***

### 6.3.2 Constructor & Destructor Documentation

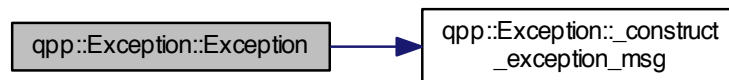
6.3.2.1 `qpp::Exception::Exception ( const std::string & where, const Type & type )` `[inline]`

Here is the call graph for this function:



6.3.2.2 `qpp::Exception::Exception ( const std::string & where, const std::string & custom )` `[inline]`

Here is the call graph for this function:



6.3.2.3 `virtual qpp::Exception::~~Exception ( )` `[inline]`, `[virtual]`, `[noexcept]`

### 6.3.3 Member Function Documentation

6.3.3.1 `std::string qpp::Exception::_construct_exception_msg ( )` `[inline]`, `[private]`

6.3.3.2 `virtual const char* qpp::Exception::what ( ) const` `[inline]`, `[override]`, `[virtual]`, `[noexcept]`

### 6.3.4 Member Data Documentation

6.3.4.1 `std::string qpp::Exception::_custom` `[private]`

6.3.4.2 `std::string qpp::Exception::_msg` `[private]`

6.3.4.3 `Type qpp::Exception::_type` `[private]`

6.3.4.4 `std::string qpp::Exception::_where` `[private]`

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

- [include/exception.h](#)

## 6.4 qpp::stat::NormalDistribution Class Reference

```
#include <stat.h>
```

### Public Member Functions

- [NormalDistribution](#) (double mean=0, double sigma=1)
- double [sample](#) ()

### Protected Attributes

- std::normal\_distribution [\\_d](#)

#### 6.4.1 Constructor & Destructor Documentation

6.4.1.1 `qpp::stat::NormalDistribution::NormalDistribution ( double mean = 0, double sigma = 1 )` [inline]

#### 6.4.2 Member Function Documentation

6.4.2.1 `double qpp::stat::NormalDistribution::sample ( )` [inline]

#### 6.4.3 Member Data Documentation

6.4.3.1 `std::normal_distribution qpp::stat::NormalDistribution::_d` [protected]

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

- include/[stat.h](#)

## 6.5 qpp::Timer Class Reference

```
#include <timer.h>
```

### Public Member Functions

- [Timer](#) ()
- void [tic](#) ()
- void [toc](#) ()
- double [seconds](#) () const
- virtual [~Timer](#) ()=default

### Protected Attributes

- std::chrono::high\_resolution\_clock::time\_point [\\_start](#)
- std::chrono::high\_resolution\_clock::time\_point [\\_end](#)

### Friends

- std::ostream & [operator<<](#) (std::ostream &os, const [Timer](#) &rhs)

### 6.5.1 Constructor & Destructor Documentation

6.5.1.1 `qpp::Timer::Timer ( )` `[inline]`

6.5.1.2 `virtual qpp::Timer::~~Timer ( )` `[virtual],[default]`

### 6.5.2 Member Function Documentation

6.5.2.1 `double qpp::Timer::seconds ( ) const` `[inline]`

6.5.2.2 `void qpp::Timer::tic ( )` `[inline]`

6.5.2.3 `void qpp::Timer::toc ( )` `[inline]`

### 6.5.3 Friends And Related Function Documentation

6.5.3.1 `std::ostream& operator<< ( std::ostream & os, const Timer & rhs )` `[friend]`

### 6.5.4 Member Data Documentation

6.5.4.1 `std::chrono::high_resolution_clock::time_point qpp::Timer::_end` `[protected]`

6.5.4.2 `std::chrono::high_resolution_clock::time_point qpp::Timer::_start` `[protected]`

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

- `include/timer.h`

## 6.6 qpp::stat::UniformRealDistribution Class Reference

```
#include <stat.h>
```

### Public Member Functions

- `UniformRealDistribution` (double *a*=0, double *b*=1)
- double `sample` ()

### Protected Attributes

- `std::uniform_real_distribution _d`

### 6.6.1 Constructor & Destructor Documentation

6.6.1.1 `qpp::stat::UniformRealDistribution::UniformRealDistribution ( double a = 0, double b = 1 )` `[inline]`

### 6.6.2 Member Function Documentation

6.6.2.1 `double qpp::stat::UniformRealDistribution::sample ( )` `[inline]`

### 6.6.3 Member Data Documentation

### 6.6.3.1 std::uniform\_real\_distribution qpp::stat::UniformRealDistribution::\_d [protected]

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

- include/[stat.h](#)



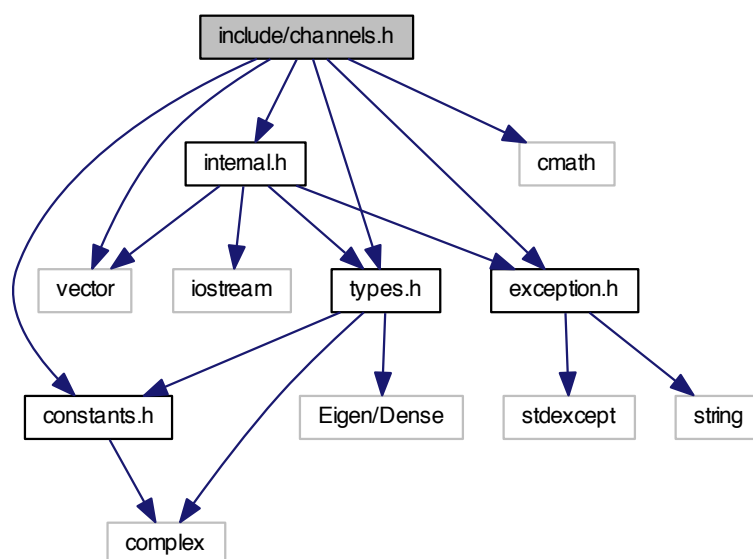
## Chapter 7

# File Documentation

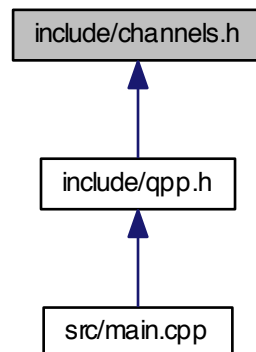
### 7.1 include/channels.h File Reference

```
#include <vector>
#include <cmath>
#include "types.h"
#include "internal.h"
#include "exception.h"
#include "constants.h"
```

Include dependency graph for channels.h:



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



## Namespaces

- [qpp](#)

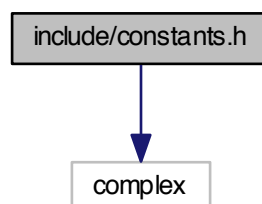
## Functions

- `types::cmat` [qpp::channel](#) (`const types::cmat &rho`, `const std::vector< types::cmat > &Ks`)
- `types::cmat` [qpp::super](#) (`const std::vector< types::cmat > &Ks`)
- `types::cmat` [qpp::choi](#) (`const std::vector< types::cmat > &Ks`)
- `std::vector< types::cmat >` [qpp::choi2kraus](#) (`const types::cmat &A`)

## 7.2 include/constants.h File Reference

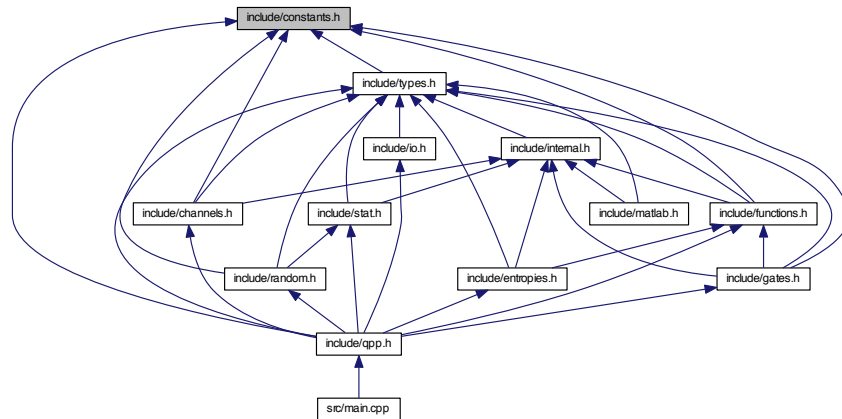
```
#include <complex>
```

Include dependency graph for constants.h:





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



## Namespaces

- [qpp](#)
- [qpp::ct](#)

## Functions

- `std::complex< double > qpp::ct::omega (size_t D)`

## Variables

- `const double qpp::ct::chop = 1e-10`
- `const double qpp::ct::eps = 1e-14`
- `const std::complex< double > qpp::ct::ii = { 0, 1 }`
- `const double qpp::ct::pi = 3.141592653589793238462643383279502884`
- `const double qpp::ct::ee = 2.718281828459045235360287471352662497`

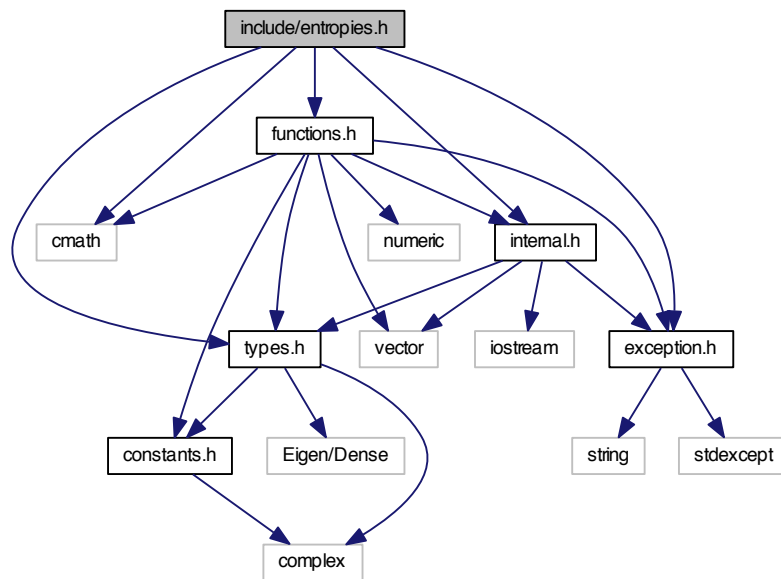
## 7.3 include/entropies.h File Reference

```

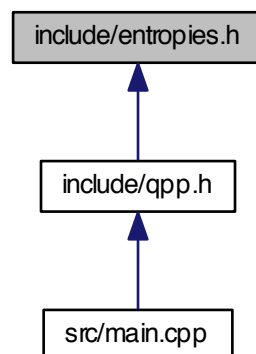
#include <cmath>
#include "types.h"
#include "functions.h"
#include "internal.h"
#include "exception.h"

```

Include dependency graph for entropies.h:



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



## Namespaces

- [qpp](#)

## Functions

- `template<typename Scalar >`  
`double qpp::shannon (const types::DynMat< Scalar > &A)`

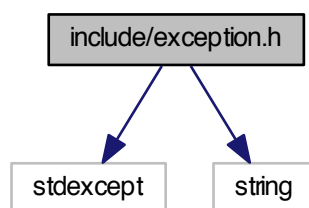
- `template<typename Scalar >`  
`double qpp::renyi (const double alpha, const types::DynMat< Scalar > &A)`
- `template<typename Scalar >`  
`double qpp::renyi\_inf (const types::DynMat< Scalar > &A)`

## 7.4 include/exception.h File Reference

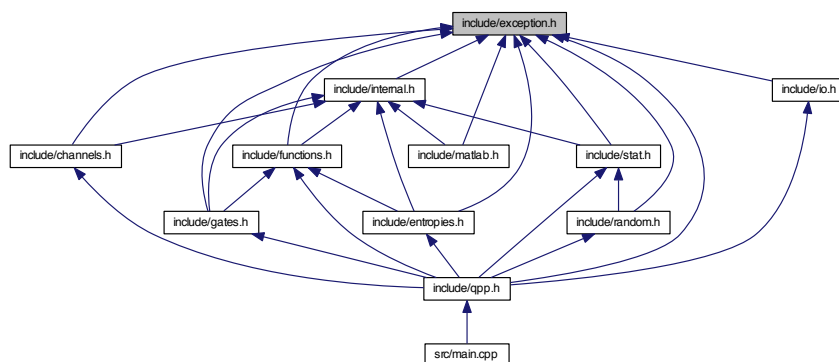
```
#include <stdexcept>
```

```
#include <string>
```

Include dependency graph for exception.h:



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



### Classes

- class [qpp::Exception](#)

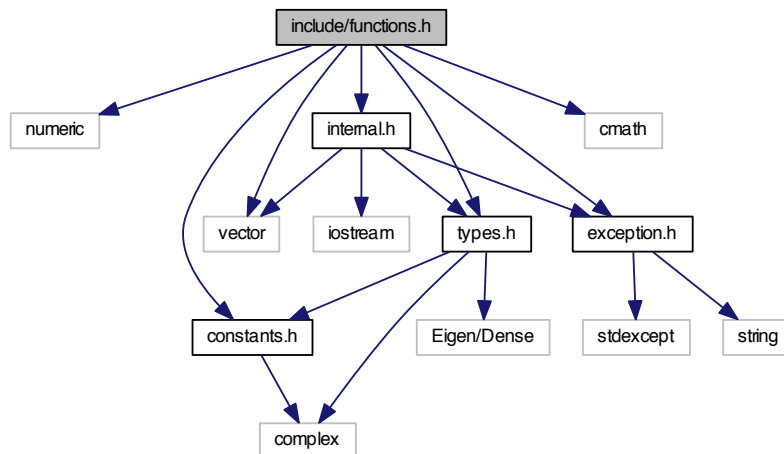
### Namespaces

- [qpp](#)

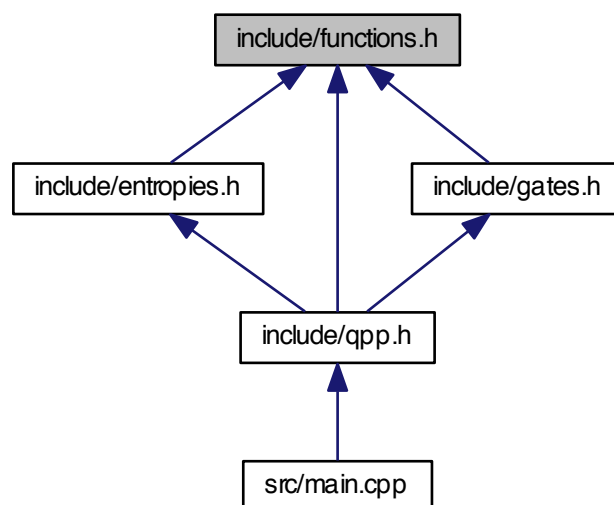
## 7.5 include/functions.h File Reference

```
#include <numeric>
#include <vector>
#include <cmath>
#include "types.h"
#include "internal.h"
#include "exception.h"
#include "constants.h"
```

Include dependency graph for functions.h:



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



## Namespaces

- [qpp](#)

## Functions

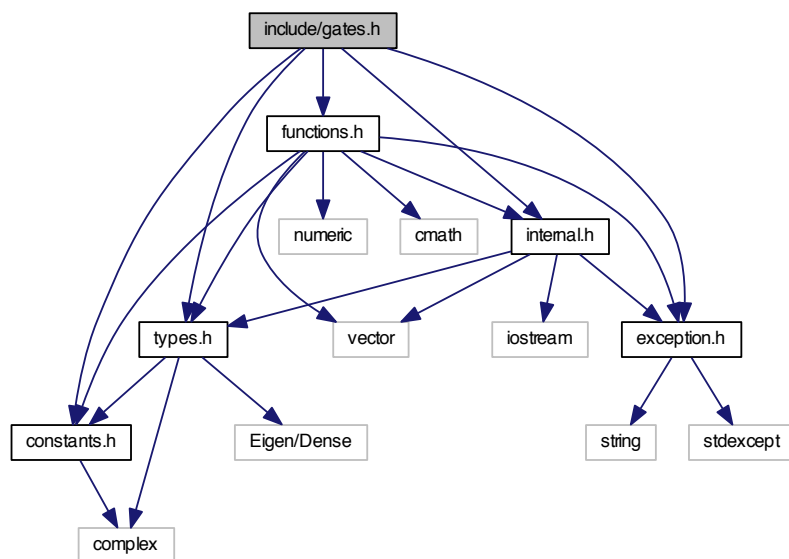
- `template<typename Scalar >`  
`types::DynMat< Scalar > qpp::transpose (const types::DynMat< Scalar > &A)`
- `template<typename Scalar >`  
`types::DynMat< Scalar > qpp::conjugate (const types::DynMat< Scalar > &A)`
- `template<typename Scalar >`  
`types::DynMat< Scalar > qpp::adjoint (const types::DynMat< Scalar > &A)`
- `template<typename Scalar >`  
`Scalar qpp::trace (const types::DynMat< Scalar > &A)`
- `template<typename Scalar >`  
`Scalar qpp::det (const types::DynMat< Scalar > &A)`
- `template<typename Scalar >`  
`Scalar qpp::sum (const types::DynMat< Scalar > &A)`
- `template<typename Scalar >`  
`double qpp::norm (const types::DynMat< Scalar > &A)`
- `template<typename Scalar >`  
`types::cmat qpp::evals (const types::DynMat< Scalar > &A)`
- `template<typename Scalar >`  
`types::cmat qpp::evecs (const types::DynMat< Scalar > &A)`
- `template<typename Scalar >`  
`types::cmat qpp::hevals (const types::DynMat< Scalar > &A)`
- `template<typename Scalar >`  
`types::cmat qpp::hevecs (const types::DynMat< Scalar > &A)`
- `template<typename Scalar >`  
`types::cmat qpp::funm (const types::DynMat< Scalar > &A, types::cplx(*f)(const types::cplx &))`
- `template<typename Scalar >`  
`types::cmat qpp::absm (const types::DynMat< Scalar > &A)`
- `template<typename Scalar >`  
`types::cmat qpp::expm (const types::DynMat< Scalar > &A)`
- `template<typename Scalar >`  
`types::cmat qpp::logm (const types::DynMat< Scalar > &A)`
- `template<typename Scalar >`  
`types::cmat qpp::sqrtm (const types::DynMat< Scalar > &A)`
- `template<typename Scalar >`  
`types::cmat qpp::sinm (const types::DynMat< Scalar > &A)`
- `template<typename Scalar >`  
`types::cmat qpp::cosm (const types::DynMat< Scalar > &A)`
- `template<typename Scalar >`  
`types::cmat qpp::spectralpowm (const types::DynMat< Scalar > &A, const types::cplx z)`
- `template<typename Scalar >`  
`types::DynMat< Scalar > qpp::powm (const types::DynMat< Scalar > &A, size_t n)`
- `template<typename InputScalar , typename OutputScalar >`  
`types::DynMat< OutputScalar > qpp::fun (const types::DynMat< InputScalar > &A, OutputScalar(*f)(const InputScalar &))`
- `template<typename Scalar >`  
`types::DynMat< Scalar > qpp::kron (const types::DynMat< Scalar > &A, const types::DynMat< Scalar > &B)`
- `template<typename Scalar >`  
`types::DynMat< Scalar > qpp::kronlist (const std::vector< types::DynMat< Scalar > > &list)`
- `template<typename Scalar >`  
`types::DynMat< Scalar > qpp::kronpow (const types::DynMat< Scalar > &A, size_t n)`

- `template<typename Scalar >`  
`types::DynMat< Scalar > qpp::reshape (const types::DynMat< Scalar > &A, size_t rows, size_t cols)`
- `template<typename Scalar >`  
`types::DynMat< Scalar > qpp::syspermute (const types::DynMat< Scalar > &A, const std::vector< size_t > perm, const std::vector< size_t > &dims)`
- `template<typename Scalar >`  
`types::DynMat< Scalar > qpp::ptrace2 (const types::DynMat< Scalar > &A, const std::vector< size_t > dims)`
- `template<typename Scalar >`  
`types::DynMat< Scalar > qpp::ptrace (const types::DynMat< Scalar > &A, const std::vector< size_t > &subsys, const std::vector< size_t > &dims)`
- `template<typename Scalar >`  
`types::DynMat< Scalar > qpp::ptranspose (const types::DynMat< Scalar > &A, const std::vector< size_t > &subsys, const std::vector< size_t > &dims)`
- `template<typename Scalar >`  
`types::DynMat< Scalar > qpp::comm (const types::DynMat< Scalar > &A, const types::DynMat< Scalar > &B)`
- `template<typename Scalar >`  
`types::DynMat< Scalar > qpp::anticomm (const types::DynMat< Scalar > &A, const types::DynMat< Scalar > &B)`
- `template<typename Scalar >`  
`types::DynMat< Scalar > qpp::proj (const types::DynMat< Scalar > &V)`
- `template<typename Scalar >`  
`types::DynMat< Scalar > qpp::expandout (const types::DynMat< Scalar > &A, size_t pos, const std::vector< size_t > &dims)`
- `template<typename Scalar >`  
`types::DynMat< Scalar > qpp::grams (const std::vector< types::DynMat< Scalar > > &vecs)`
- `template<typename Scalar >`  
`types::DynMat< Scalar > qpp::grams (const types::DynMat< Scalar > &A)`

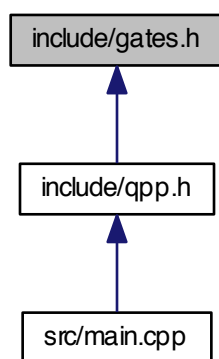
## 7.6 include/gates.h File Reference

```
#include "types.h"
#include "constants.h"
#include "functions.h"
#include "internal.h"
#include "exception.h"
```

Include dependency graph for gates.h:



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



## Namespaces

- `qpp`
- `qpp::gt`

## Functions

- void `qpp::gt::_init_gates()`
- types::cmat `qpp::gt::Rtheta` (double theta)

- `types::cmat qpp::gt::ld` (size\_t D)
- `types::cmat qpp::gt::Zd` (size\_t D)
- `types::cmat qpp::gt::Fd` (size\_t D)
- `types::cmat qpp::gt::Xd` (size\_t D)
- `types::cmat qpp::gt::CTRL` (const types::cmat &A, const std::vector< size\_t > &ctrl, const std::vector< size\_t > &gate, size\_t n, size\_t D=2)

## Variables

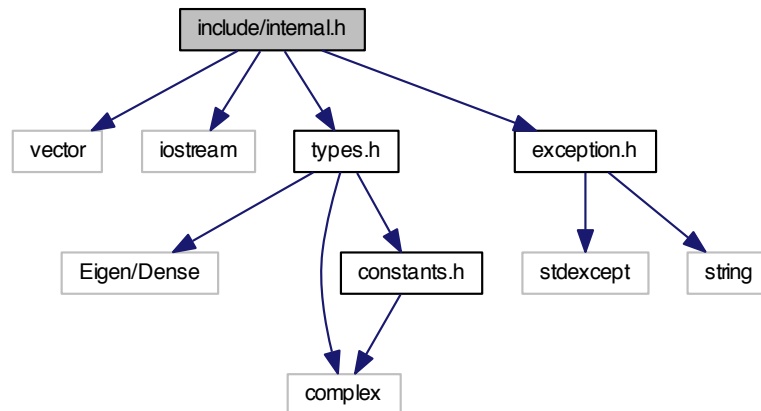
- `types::cmat qpp::gt::ld2`
- `types::cmat qpp::gt::H`
- `types::cmat qpp::gt::X`
- `types::cmat qpp::gt::Y`
- `types::cmat qpp::gt::Z`
- `types::cmat qpp::gt::S`
- `types::cmat qpp::gt::T`
- `types::cmat qpp::gt::CNOTab`
- `types::cmat qpp::gt::CNOTba`
- `types::cmat qpp::gt::CZ`
- `types::cmat qpp::gt::CS`
- `types::cmat qpp::gt::SWAP`
- `types::cmat qpp::gt::TOF`
- `types::cmat qpp::gt::FRED`
- `types::cmat qpp::gt::x0`
- `types::cmat qpp::gt::x1`
- `types::cmat qpp::gt::y0`
- `types::cmat qpp::gt::y1`
- `types::cmat qpp::gt::z0`
- `types::cmat qpp::gt::z1`
- `types::cmat qpp::gt::b00`
- `types::cmat qpp::gt::b01`
- `types::cmat qpp::gt::b10`
- `types::cmat qpp::gt::b11`

## 7.7 include/internal.h File Reference

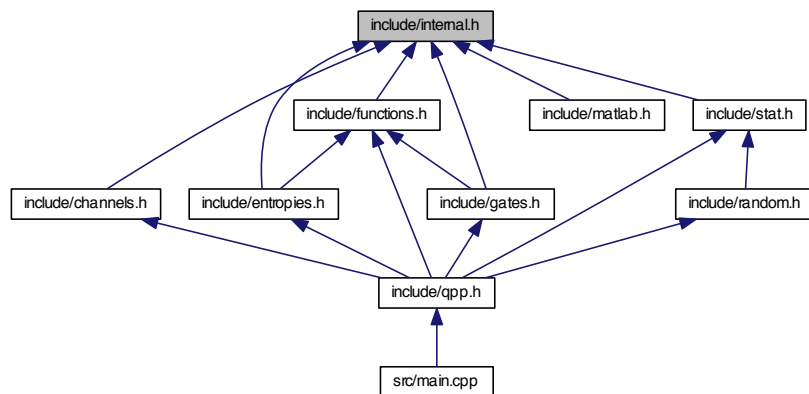
```
#include <vector>
#include <iostream>
#include "types.h"
#include "exception.h"
```



Include dependency graph for internal.h:



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



## Namespaces

- [qpp](#)
- [qpp::internal](#)

## Functions

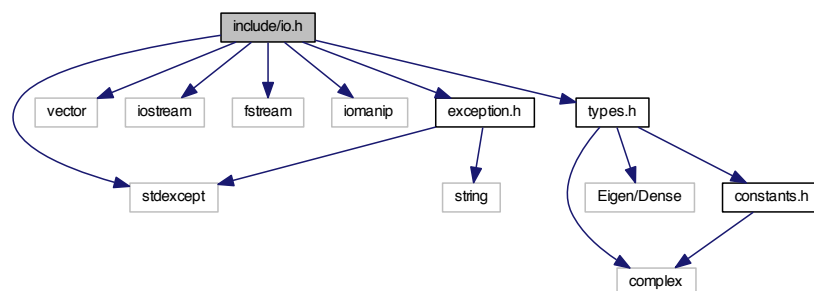
- void [qpp::internal::\\_n2multiidx](#) (size\_t n, size\_t numdims, const size\_t \*dims, size\_t \*result)
- size\_t [qpp::internal::\\_multiidx2n](#) (const size\_t \*midx, size\_t numdims, const size\_t \*dims)
- template<typename Scalar >  
bool [qpp::internal::\\_check\\_square\\_mat](#) (const types::DynMat< Scalar > &A)
- template<typename Scalar >  
bool [qpp::internal::\\_check\\_vector](#) (const types::DynMat< Scalar > &A)

- `template<typename Scalar >`  
`bool qpp::internal::_check_row_vector (const types::DynMat< Scalar > &A)`
- `template<typename Scalar >`  
`bool qpp::internal::_check_col_vector (const types::DynMat< Scalar > &A)`
- `template<typename T >`  
`bool qpp::internal::_check_nonzero_size (const T &x)`
- `bool qpp::internal::_check_dims (const std::vector< size_t > &dims)`
- `template<typename Scalar >`  
`bool qpp::internal::_check_dims_match_mat (const std::vector< size_t > &dims, const types::DynMat< Scalar > &A)`
- `bool qpp::internal::_check_eq_dims (const std::vector< size_t > &dims, size_t dim)`
- `bool qpp::internal::_check_subsys (const std::vector< size_t > &subsys, const std::vector< size_t > &dims)`
- `bool qpp::internal::_check_perm (const std::vector< size_t > &perm, const std::vector< size_t > &dims)`
- `template<typename Scalar >`  
`void qpp::internal::_syspermute_worker (const size_t *midxcol, size_t numdims, const size_t *cdims, const size_t *cperm, size_t i, size_t j, size_t &iperm, size_t &jperm, const types::DynMat< Scalar > &A, types::DynMat< Scalar > &result)`
- `template<typename Scalar >`  
`void qpp::internal::_ptranspose_worker (const size_t *midxcol, size_t numdims, size_t numsubsys, const size_t *cdims, const size_t *csubsys, size_t i, size_t j, size_t &iperm, size_t &jperm, const types::DynMat< Scalar > &A, types::DynMat< Scalar > &result)`

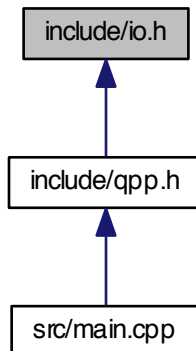
## 7.8 include/io.h File Reference

```
#include <stdexcept>
#include <vector>
#include <iostream>
#include <fstream>
#include <iomanip>
#include "types.h"
#include "exception.h"
```

Include dependency graph for io.h:



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



## Namespaces

- [qpp](#)

## Functions

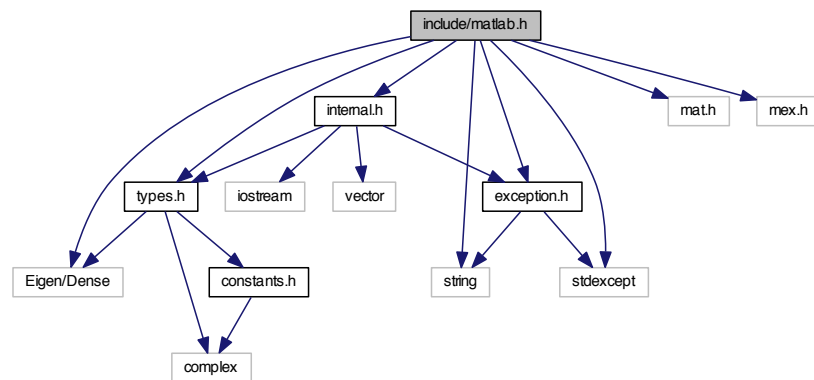
- `template<typename T >`  
`void qpp::disp (const T &x, const std::string &separator=" ", const std::string &start="[" , const std::string &end="]", std::ostream &os=std::cout)`
- `template<typename T >`  
`void qpp::displn (const T &x, const std::string &separator=" ", const std::string &start="[" , const std::string &end="]", std::ostream &os=std::cout)`
- `template<typename T >`  
`void qpp::disp (const T *x, const size_t n, const std::string &separator=" ", const std::string &start="[" , const std::string &end="]", std::ostream &os=std::cout)`
- `template<typename T >`  
`void qpp::displn (const T *x, const size_t n, const std::string &separator=" ", const std::string &start="[" , const std::string &end="]", std::ostream &os=std::cout)`
- `template<typename Scalar >`  
`void qpp::disp (const types::DynMat< Scalar > &A, double chop=ct::chop, std::ostream &os=std::cout)`
- `template<typename Scalar >`  
`void qpp::displn (const types::DynMat< Scalar > &A, double chop=ct::chop, std::ostream &os=std::cout)`
- `void qpp::disp (const types::cplx c, double chop=ct::chop, std::ostream &os=std::cout)`
- `void qpp::displn (const types::cplx c, double chop=ct::chop, std::ostream &os=std::cout)`
- `template<typename Scalar >`  
`void qpp::save (const types::DynMat< Scalar > &A, const std::string &fname)`
- `template<typename Scalar >`  
`types::DynMat< Scalar > qpp::load (const std::string &fname)`

## 7.9 include/matlab.h File Reference

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

```
#include <string>
#include <stdexcept>
#include "types.h"
#include "internal.h"
#include "exception.h"
#include "mat.h"
#include "mex.h"
```

Include dependency graph for matlab.h:



## Namespaces

- [qpp](#)

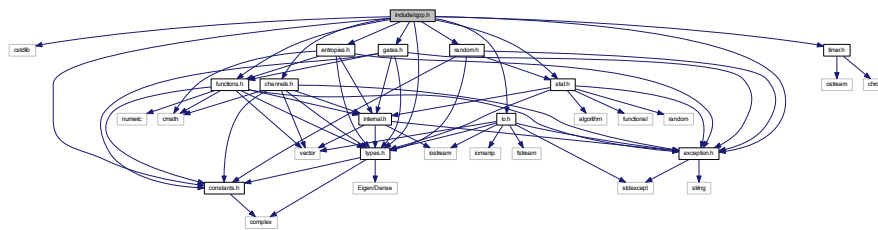
## Functions

- `template<typename Scalar >`  
`types::DynMat< Scalar > qpp::loadMATLABmatrix (const std::string &mat_file, const std::string &var_name)`
- `template<>`  
`types::DynMat< double > qpp::loadMATLABmatrix (const std::string &mat_file, const std::string &var_name)`
- `template<>`  
`types::DynMat< types::cplx > qpp::loadMATLABmatrix (const std::string &mat_file, const std::string &var_name)`
- `template<typename Scalar >`  
`void qpp::saveMATLABmatrix (const types::DynMat< Scalar > &A, const std::string &mat_file, const std::string &var_name, const std::string &mode)`
- `template<>`  
`void qpp::saveMATLABmatrix (const types::DynMat< double > &A, const std::string &mat_file, const std::string &var_name, const std::string &mode)`
- `template<>`  
`void qpp::saveMATLABmatrix (const types::DynMat< types::cplx > &A, const std::string &mat_file, const std::string &var_name, const std::string &mode)`

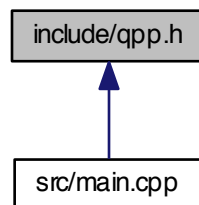
## 7.10 include/qpp.h File Reference

```
#include <cstdlib>
```

```
#include "types.h"
#include "constants.h"
#include "gates.h"
#include "stat.h"
#include "functions.h"
#include "random.h"
#include "entropies.h"
#include "io.h"
#include "timer.h"
#include "exception.h"
#include "channels.h"
Include dependency graph for qpp.h:
```



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



## Namespaces

- [qpp](#)
- [qpp::gt](#)

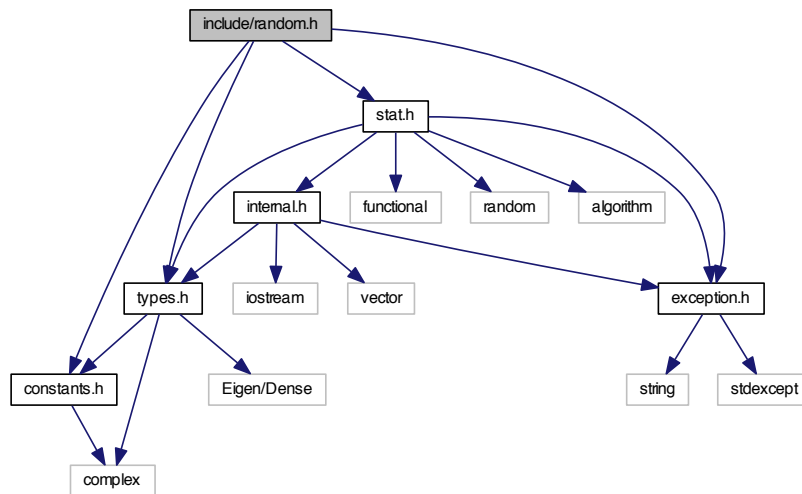
## Functions

- [int qpp::\\_init\(\)](#)

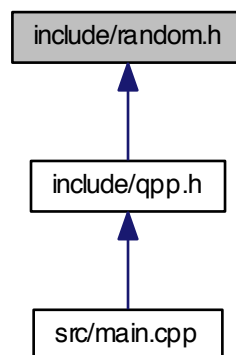
## 7.11 include/random.h File Reference

```
#include "types.h"
#include "stat.h"
#include "constants.h"
#include "exception.h"
```

Include dependency graph for random.h:



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



## Namespaces

- [qpp](#)

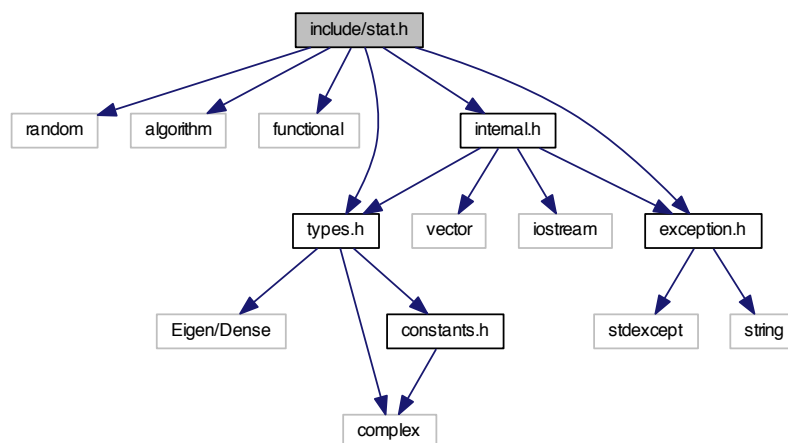
## Functions

- `template<typename Scalar >`  
`types::DynMat< Scalar > qpp::rand (size_t rows, size_t cols, double a=0, double b=1)`
- `template<>`  
`types::DynMat< double > qpp::rand (size_t rows, size_t cols, double a, double b)`

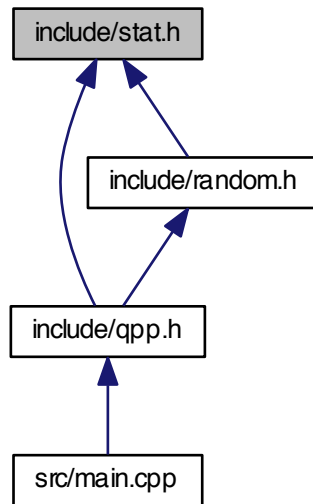
- `template<>`  
`types::DynMat< types::cplx > qpp::rand (size_t rows, size_t cols, double a, double b)`
- `double qpp::rand (double a=0, double b=1)`
- `template<typename Scalar >`  
`types::DynMat< Scalar > qpp::randn (size_t rows, size_t cols, double mean=0, double sigma=1)`
- `template<>`  
`types::DynMat< double > qpp::randn (size_t rows, size_t cols, double mean, double sigma)`
- `template<>`  
`types::DynMat< types::cplx > qpp::randn (size_t rows, size_t cols, double mean, double sigma)`
- `double qpp::randn (double mean=0, double sigma=1)`
- `types::cmat qpp::randU (size_t D)`
- `types::cmat qpp::randV (size_t Din, size_t Dout)`
- `std::vector< types::cmat > qpp::randKraus (size_t n, size_t D)`
- `types::cmat qpp::randH (size_t D)`
- `types::cmat qpp::randket (size_t D)`
- `types::cmat qpp::randrho (size_t D)`

## 7.12 include/stat.h File Reference

```
#include <random>
#include <algorithm>
#include <functional>
#include "types.h"
#include "internal.h"
#include "exception.h"
Include dependency graph for stat.h:
```



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



## Classes

- class [qpp::stat::NormalDistribution](#)
- class [qpp::stat::UniformRealDistribution](#)
- class [qpp::stat::DiscreteDistribution](#)
- class [qpp::stat::DiscreteDistributionFromComplex](#)

## Namespaces

- [qpp](#)
- [qpp::stat](#)

## Variables

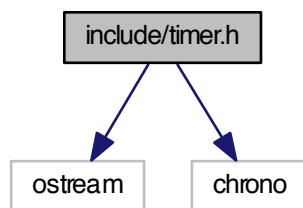
- `std::random_device` [qpp::stat::\\_rd](#)
- `std::mt19937` [qpp::stat::\\_rng](#)

## 7.13 include/timer.h File Reference

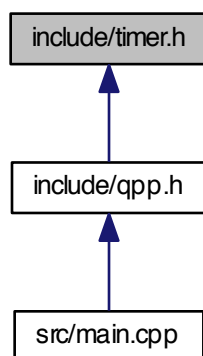
```
#include <ostream>
#include <chrono>
```



Include dependency graph for timer.h:



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



## Classes

- class [qpp::Timer](#)

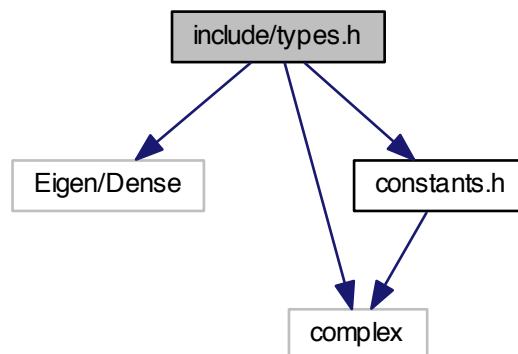
## Namespaces

- [qpp](#)

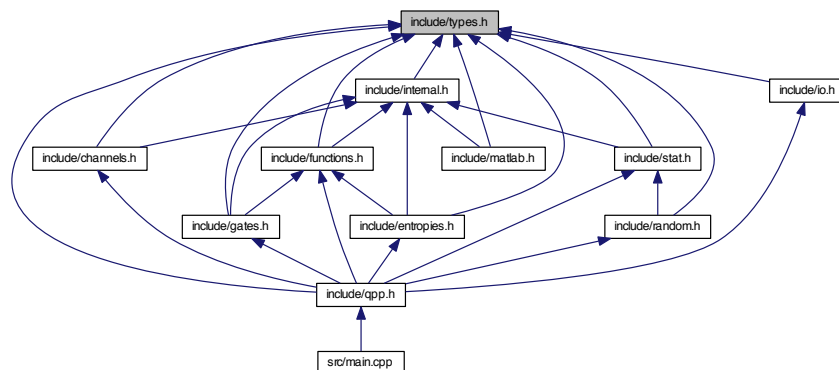
## 7.14 include/types.h File Reference

```
#include <Eigen/Dense>
#include <complex>
#include "constants.h"
```

Include dependency graph for types.h:



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



## Namespaces

- `qpp`
- `qpp::types`

## Typedefs

- `typedef std::complex< double > qpp::types::cplx`
- `typedef Eigen::MatrixXcd qpp::types::cmat`
- `typedef Eigen::MatrixXd qpp::types::dmat`
- `typedef Eigen::MatrixXf qpp::types::fmat`
- `typedef Eigen::MatrixXi qpp::types::imat`
- `template<typename Expression >`  
`using qpp::types::Expression2DynMat = Eigen::Matrix< typename Expression::Scalar, Eigen::Dynamic,`  
`Eigen::Dynamic >`

- `template<typename Scalar >`  
`using qpp::types::DynMat = Eigen::Matrix< Scalar, Eigen::Dynamic, Eigen::Dynamic >`

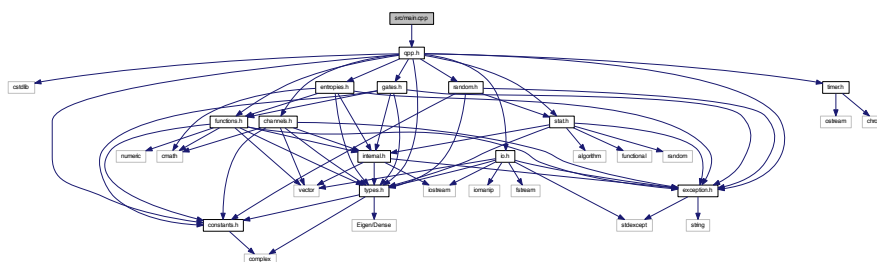
## Functions

- `int qpp::types::myfunc (int a, int b)`

## 7.15 src/main.cpp File Reference

```
#include "qpp.h"
```

Include dependency graph for main.cpp:



## Functions

- `int main ()`

### 7.15.1 Function Documentation

Here is the call graph for this function:

