

qpp  
0.1

Generated by Doxygen 1.8.5

Mon Apr 7 2014 23:58:41



# 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	absm . . . . .	12
5.1.1.2	adjoint . . . . .	12
5.1.1.3	anticomm . . . . .	13
5.1.1.4	channel . . . . .	13
5.1.1.5	choi . . . . .	14
5.1.1.6	choi2kraus . . . . .	14
5.1.1.7	comm . . . . .	15
5.1.1.8	conjugate . . . . .	15
5.1.1.9	cosm . . . . .	15
5.1.1.10	det . . . . .	16
5.1.1.11	disp . . . . .	16
5.1.1.12	disp . . . . .	16
5.1.1.13	disp . . . . .	16
5.1.1.14	disp . . . . .	16
5.1.1.15	displn . . . . .	16
5.1.1.16	displn . . . . .	17
5.1.1.17	displn . . . . .	17
5.1.1.18	displn . . . . .	17

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

5.1.1.59	<a href="#">reshape</a>	30
5.1.1.60	<a href="#">save</a>	31
5.1.1.61	<a href="#">saveMATLABmatrix</a>	31
5.1.1.62	<a href="#">saveMATLABmatrix</a>	31
5.1.1.63	<a href="#">saveMATLABmatrix</a>	31
5.1.1.64	<a href="#">shannon</a>	32
5.1.1.65	<a href="#">sinm</a>	32
5.1.1.66	<a href="#">spectralpwm</a>	32
5.1.1.67	<a href="#">sqrtm</a>	33
5.1.1.68	<a href="#">sum</a>	33
5.1.1.69	<a href="#">super</a>	33
5.1.1.70	<a href="#">syspermute</a>	34
5.1.1.71	<a href="#">trace</a>	34
5.1.1.72	<a href="#">transpose</a>	35
5.1.2	<a href="#">Variable Documentation</a>	35
5.1.2.1	<a href="#">gt</a>	35
5.1.2.2	<a href="#">rdevs</a>	35
5.2	<a href="#">qpp::ct Namespace Reference</a>	35
5.2.1	<a href="#">Function Documentation</a>	35
5.2.1.1	<a href="#">omega</a>	35
5.2.2	<a href="#">Variable Documentation</a>	35
5.2.2.1	<a href="#">chop</a>	35
5.2.2.2	<a href="#">ee</a>	35
5.2.2.3	<a href="#">eps</a>	35
5.2.2.4	<a href="#">ii</a>	35
5.2.2.5	<a href="#">pi</a>	35
5.3	<a href="#">qpp::internal Namespace Reference</a>	35
5.3.1	<a href="#">Function Documentation</a>	36
5.3.1.1	<a href="#">_check_col_vector</a>	36
5.3.1.2	<a href="#">_check_dims</a>	36
5.3.1.3	<a href="#">_check_dims_match_mat</a>	36
5.3.1.4	<a href="#">_check_eq_dims</a>	36
5.3.1.5	<a href="#">_check_nonzero_size</a>	36
5.3.1.6	<a href="#">_check_perm</a>	36
5.3.1.7	<a href="#">_check_row_vector</a>	36
5.3.1.8	<a href="#">_check_square_mat</a>	36
5.3.1.9	<a href="#">_check_subsys</a>	36
5.3.1.10	<a href="#">_check_vector</a>	36
5.3.1.11	<a href="#">_multiidx2n</a>	36
5.3.1.12	<a href="#">_n2multiidx</a>	36

5.3.1.13	<a href="#">_ptranspose_worker</a>	37
5.3.1.14	<a href="#">_syspermute_worker</a>	37
5.4	<a href="#">qpp::stat Namespace Reference</a>	37
5.5	<a href="#">qpp::types Namespace Reference</a>	37
5.5.1	<a href="#">Typedef Documentation</a>	38
5.5.1.1	<a href="#">cmat</a>	38
5.5.1.2	<a href="#">cplx</a>	38
5.5.1.3	<a href="#">dmat</a>	38
5.5.1.4	<a href="#">DynMat</a>	38
5.5.1.5	<a href="#">Expression2DynMat</a>	38
5.5.1.6	<a href="#">fmat</a>	38
5.5.1.7	<a href="#">imat</a>	38
<b>6</b>	<b><a href="#">Class Documentation</a></b>	<b>39</b>
6.1	<a href="#">qpp::stat::DiscreteDistribution Class Reference</a>	39
6.1.1	<a href="#">Constructor &amp; Destructor Documentation</a>	39
6.1.1.1	<a href="#">DiscreteDistribution</a>	39
6.1.1.2	<a href="#">DiscreteDistribution</a>	39
6.1.1.3	<a href="#">DiscreteDistribution</a>	39
6.1.2	<a href="#">Member Function Documentation</a>	39
6.1.2.1	<a href="#">probabilities</a>	39
6.1.2.2	<a href="#">sample</a>	40
6.1.3	<a href="#">Member Data Documentation</a>	40
6.1.3.1	<a href="#">_d</a>	40
6.2	<a href="#">qpp::stat::DiscreteDistributionFromComplex Class Reference</a>	40
6.2.1	<a href="#">Constructor &amp; Destructor Documentation</a>	40
6.2.1.1	<a href="#">DiscreteDistributionFromComplex</a>	41
6.2.1.2	<a href="#">DiscreteDistributionFromComplex</a>	41
6.2.1.3	<a href="#">DiscreteDistributionFromComplex</a>	41
6.2.1.4	<a href="#">DiscreteDistributionFromComplex</a>	42
6.2.2	<a href="#">Member Function Documentation</a>	42
6.2.2.1	<a href="#">cplx2amplitudes</a>	42
6.2.2.2	<a href="#">probabilities</a>	42
6.2.2.3	<a href="#">sample</a>	42
6.2.3	<a href="#">Member Data Documentation</a>	42
6.2.3.1	<a href="#">_d</a>	42
6.3	<a href="#">qpp::Exception Class Reference</a>	42
6.3.1	<a href="#">Member Enumeration Documentation</a>	44
6.3.1.1	<a href="#">Type</a>	44
6.3.2	<a href="#">Constructor &amp; Destructor Documentation</a>	44

6.3.2.1	Exception	44
6.3.2.2	Exception	45
6.3.2.3	~Exception	45
6.3.3	Member Function Documentation	45
6.3.3.1	_construct_exception_msg	45
6.3.3.2	what	45
6.3.4	Member Data Documentation	45
6.3.4.1	_custom	45
6.3.4.2	_msg	45
6.3.4.3	_type	45
6.3.4.4	_where	45
6.4	qpp::Gates Class Reference	45
6.4.1	Constructor & Destructor Documentation	46
6.4.1.1	Gates	46
6.4.1.2	Gates	47
6.4.2	Member Function Documentation	47
6.4.2.1	_init_gates	47
6.4.2.2	CTRL	47
6.4.2.3	Fd	47
6.4.2.4	getInstance	47
6.4.2.5	Id	47
6.4.2.6	operator=	47
6.4.2.7	Rtheta	47
6.4.2.8	Xd	48
6.4.2.9	Zd	48
6.4.3	Member Data Documentation	48
6.4.3.1	b00	48
6.4.3.2	b01	48
6.4.3.3	b10	48
6.4.3.4	b11	48
6.4.3.5	CNOTab	48
6.4.3.6	CNOTba	48
6.4.3.7	CS	48
6.4.3.8	CZ	48
6.4.3.9	FRED	48
6.4.3.10	H	48
6.4.3.11	Id2	48
6.4.3.12	S	49
6.4.3.13	SWAP	49
6.4.3.14	T	49

6.4.3.15	TOF	49
6.4.3.16	X	49
6.4.3.17	x0	49
6.4.3.18	x1	49
6.4.3.19	Y	49
6.4.3.20	y0	49
6.4.3.21	y1	49
6.4.3.22	Z	49
6.4.3.23	z0	49
6.4.3.24	z1	49
6.5	qpp::stat::NormalDistribution Class Reference	49
6.5.1	Constructor & Destructor Documentation	49
6.5.1.1	NormalDistribution	49
6.5.2	Member Function Documentation	49
6.5.2.1	sample	50
6.5.3	Member Data Documentation	50
6.5.3.1	_d	50
6.6	qpp::RandomDevices Class Reference	50
6.6.1	Constructor & Destructor Documentation	50
6.6.1.1	RandomDevices	50
6.6.1.2	RandomDevices	51
6.6.1.3	~RandomDevices	51
6.6.2	Member Function Documentation	51
6.6.2.1	getInstance	51
6.6.2.2	operator=	51
6.6.3	Member Data Documentation	51
6.6.3.1	_rd	51
6.6.3.2	_rng	51
6.7	qpp::Timer Class Reference	51
6.7.1	Constructor & Destructor Documentation	51
6.7.1.1	Timer	51
6.7.1.2	~Timer	51
6.7.2	Member Function Documentation	51
6.7.2.1	seconds	51
6.7.2.2	tic	52
6.7.2.3	toc	52
6.7.3	Friends And Related Function Documentation	52
6.7.3.1	operator<<	52
6.7.4	Member Data Documentation	52
6.7.4.1	_end	52



6.7.4.2	<a href="#">_start</a>	52
6.8	<a href="#">qpp::stat::UniformRealDistribution Class Reference</a>	52
6.8.1	<a href="#">Constructor &amp; Destructor Documentation</a>	52
6.8.1.1	<a href="#">UniformRealDistribution</a>	52
6.8.2	<a href="#">Member Function Documentation</a>	52
6.8.2.1	<a href="#">sample</a>	52
6.8.3	<a href="#">Member Data Documentation</a>	53
6.8.3.1	<a href="#">_d</a>	53
<b>7</b>	<b><a href="#">File Documentation</a></b>	<b>55</b>
7.1	<a href="#">include/channels.h File Reference</a>	55
7.2	<a href="#">include/constants.h File Reference</a>	56
7.3	<a href="#">include/entropies.h File Reference</a>	57
7.4	<a href="#">include/exception.h File Reference</a>	59
7.5	<a href="#">include/functions.h File Reference</a>	60
7.6	<a href="#">include/gates.h File Reference</a>	62
7.7	<a href="#">include/internal.h File Reference</a>	64
7.8	<a href="#">include/io.h File Reference</a>	65
7.9	<a href="#">include/matlab.h File Reference</a>	66
7.10	<a href="#">include/qpp.h File Reference</a>	67
7.11	<a href="#">include/randevs.h File Reference</a>	68
7.12	<a href="#">include/random.h File Reference</a>	70
7.13	<a href="#">include/stat.h File Reference</a>	71
7.14	<a href="#">include/timer.h File Reference</a>	72
7.15	<a href="#">include/types.h File Reference</a>	73
7.16	<a href="#">src/main.cpp File Reference</a>	75
7.16.1	<a href="#">Function Documentation</a>	75
7.16.1.1	<a href="#">main</a>	75
7.16.1.2	<a href="#">test</a>	75



# 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::internal</a>	35
<a href="#">qpp::stat</a>	37
<a href="#">qpp::types</a>	37



## Chapter 2

# Hierarchical Index

### 2.1 Class Hierarchy

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

qpp::stat::DiscreteDistribution . . . . .	39
qpp::stat::DiscreteDistributionFromComplex . . . . .	40
exception	
qpp::Exception . . . . .	42
qpp::Gates . . . . .	45
qpp::stat::NormalDistribution . . . . .	49
qpp::RandomDevices . . . . .	50
qpp::Timer . . . . .	51
qpp::stat::UniformRealDistribution . . . . .	52



## 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>	39
<a href="#">qpp::stat::DiscreteDistributionFromComplex</a>	40
<a href="#">qpp::Exception</a>	42
<a href="#">qpp::Gates</a>	45
<a href="#">qpp::stat::NormalDistribution</a>	49
<a href="#">qpp::RandomDevices</a>	50
<a href="#">qpp::Timer</a>	51
<a href="#">qpp::stat::UniformRealDistribution</a>	52





# Chapter 4

## File Index

### 4.1 File List

Here is a list of all files with brief descriptions:

include/channels.h	55
include/constants.h	56
include/entropies.h	57
include/exception.h	59
include/functions.h	60
include/gates.h	62
include/internal.h	64
include/io.h	65
include/matlab.h	66
include/qpp.h	67
include/randevs.h	68
include/random.h	70
include/stat.h	71
include/timer.h	72
include/types.h	73
src/main.cpp	75



## Chapter 5

# Namespace Documentation

### 5.1 qpp Namespace Reference

#### Namespaces

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

#### Classes

- class [Exception](#)
- class [Gates](#)
- class [RandomDevices](#)
- 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 evecs (const types::DynMat< Scalar > &A)`
- `template<typename Scalar >`  
`types::cmat hevals (const types::DynMat< Scalar > &A)`
- `template<typename Scalar >`  
`types::cmat hevecs (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)`
- `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)

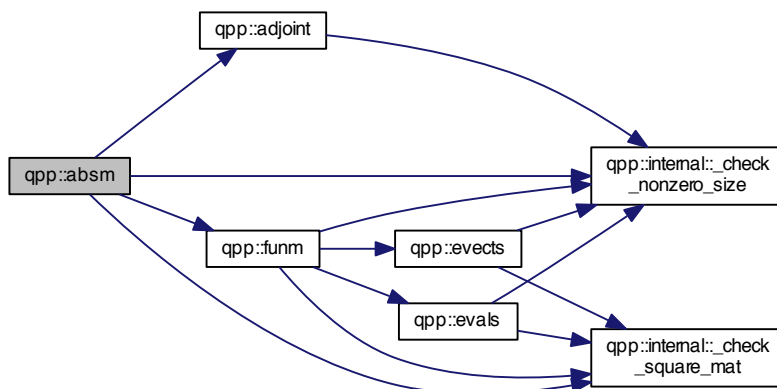
## Variables

- `RandomDevices & rdevs = RandomDevices::getInstance()`
- `const Gates & gt = Gates::getInstance()`

## 5.1.1 Function Documentation

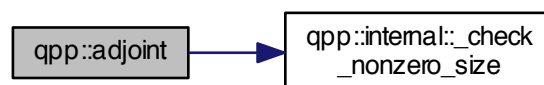
### 5.1.1.1 `template<typename Scalar > types::cmat qpp::absm ( const types::DynMat< Scalar > & A )`

Here is the call graph for this function:



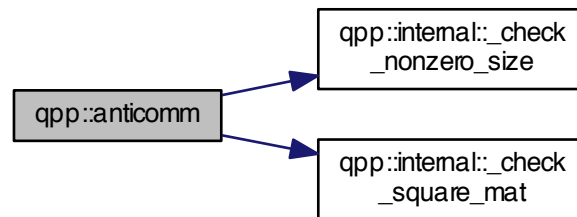
### 5.1.1.2 `template<typename Scalar > types::DynMat<Scalar> qpp::adjoint ( const types::DynMat< Scalar > & A )`

Here is the call graph for this function:



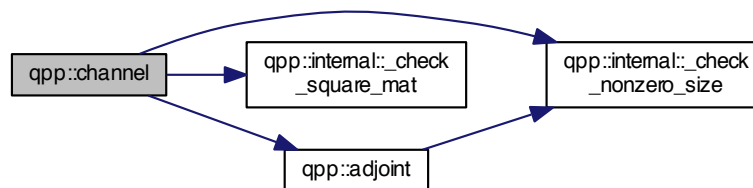
5.1.1.3 `template<typename Scalar > types::DynMat<Scalar> qpp::anticomm ( const types::DynMat< Scalar > & A,  
const types::DynMat< Scalar > & B )`

Here is the call graph for this function:



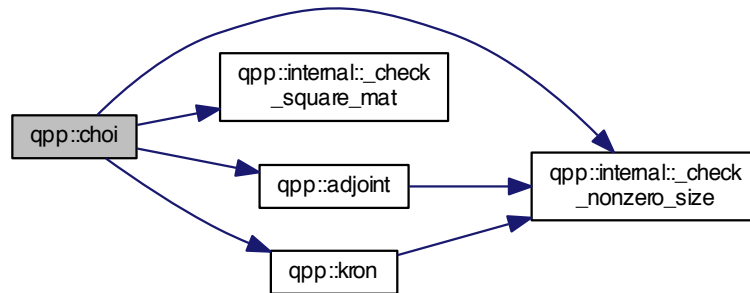
5.1.1.4 `types::cmat qpp::channel ( const types::cmat & rho, const std::vector< types::cmat > & Ks )`

Here is the call graph for this function:



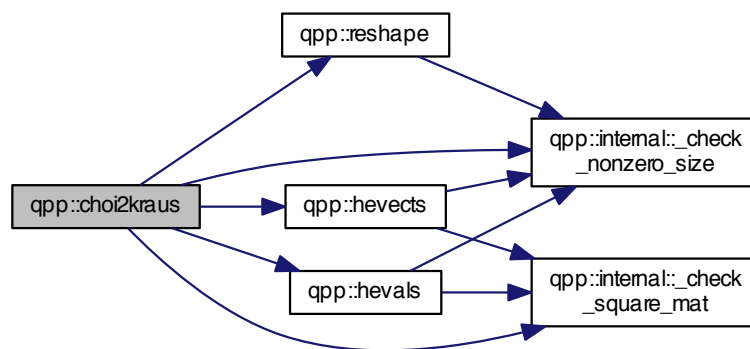
#### 5.1.1.5 `types::cmat qpp::choi ( const std::vector< types::cmat > & Ks )`

Here is the call graph for this function:



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

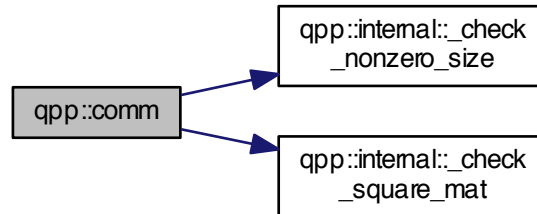
Here is the call graph for this function:





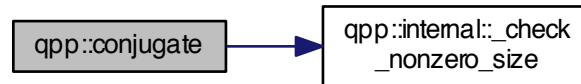
5.1.1.7 `template<typename Scalar > types::DynMat<Scalar> qpp::comm ( const types::DynMat< Scalar > & A, const types::DynMat< Scalar > & B )`

Here is the call graph for this function:



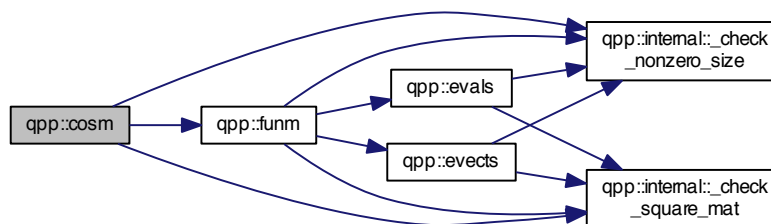
5.1.1.8 `template<typename Scalar > types::DynMat<Scalar> qpp::conjugate ( const types::DynMat< Scalar > & A )`

Here is the call graph for this function:



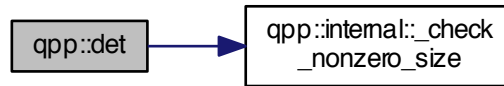
5.1.1.9 `template<typename Scalar > types::cmat qpp::cosm ( const types::DynMat< Scalar > & A )`

Here is the call graph for this function:



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

Here is the call graph for this function:



5.1.1.11 `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 )`

5.1.1.12 `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 )`

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

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

Here is the call graph for this function:



5.1.1.15 `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:



5.1.1.16 `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:



5.1.1.17 `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:



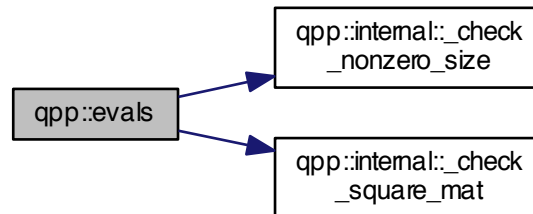
5.1.1.18 `void qpp::displn ( const types::cplx c, double chop = ct::chop, std::ostream & os = std::cout )`

Here is the call graph for this function:



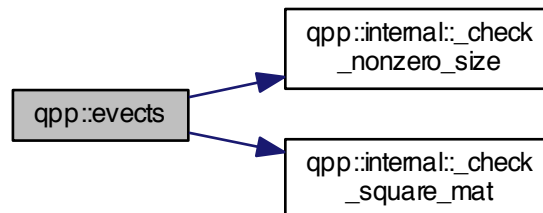
5.1.1.19 `template<typename Scalar > types::cmat qpp::evals ( const types::DynMat< Scalar > & A )`

Here is the call graph for this function:



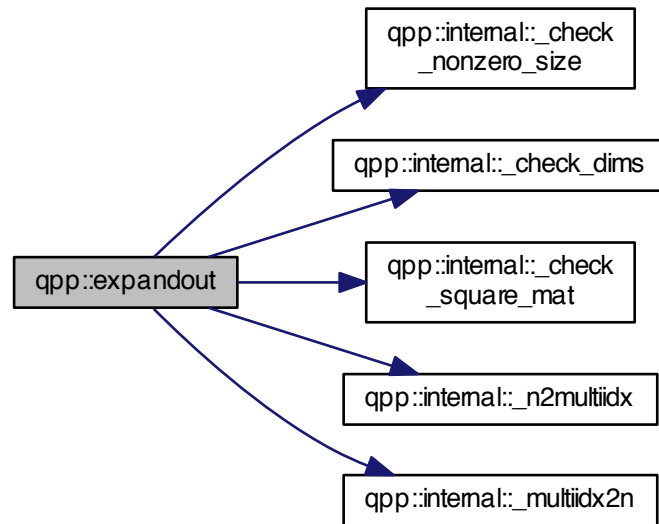
5.1.1.20 `template<typename Scalar > types::cmat qpp::evecs ( const types::DynMat< Scalar > & A )`

Here is the call graph for this function:



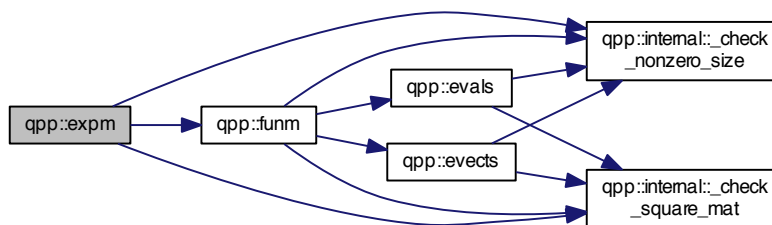
5.1.1.21 `template<typename Scalar > types::DynMat<Scalar> qpp::expandout ( const types::DynMat< Scalar > & A,  
size_t pos, const std::vector< size_t > & dims )`

Here is the call graph for this function:



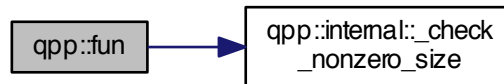
5.1.1.22 `template<typename Scalar > types::cmat qpp::expm ( const types::DynMat< Scalar > & A )`

Here is the call graph for this function:



5.1.1.23 `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:



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

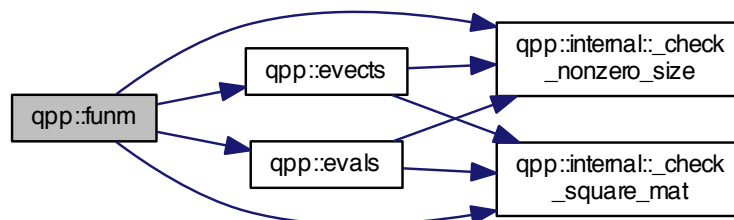
Parameters

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

Returns

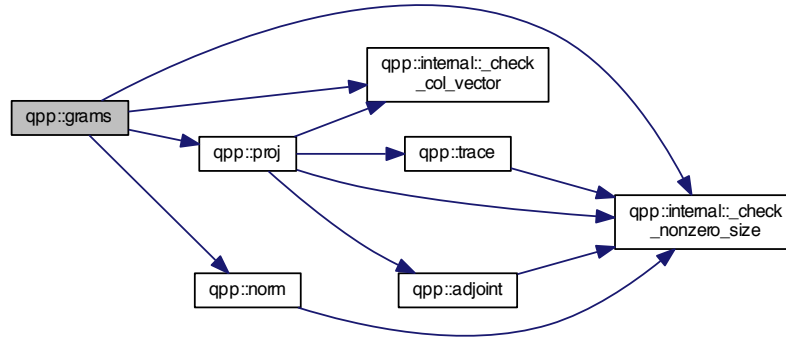
[types::cmat](#)

Here is the call graph for this function:



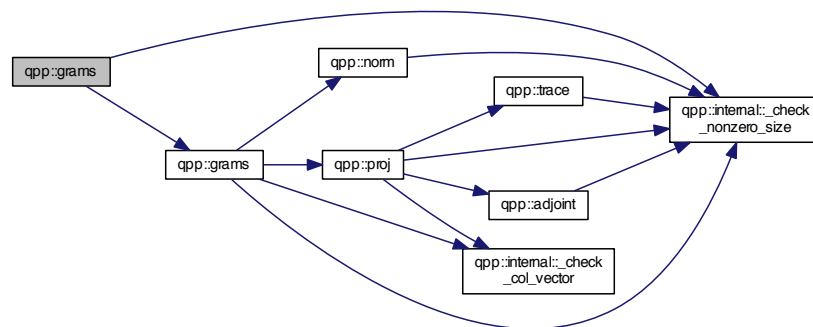
5.1.1.25 `template<typename Scalar > types::DynMat<Scalar> qpp::grams ( const std::vector< types::DynMat< Scalar >> & vecs )`

Here is the call graph for this function:



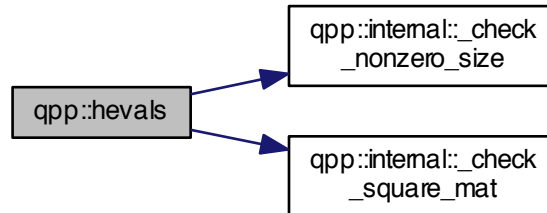
5.1.1.26 `template<typename Scalar > types::DynMat<Scalar> qpp::grams ( const types::DynMat< Scalar > & A )`

Here is the call graph for this function:



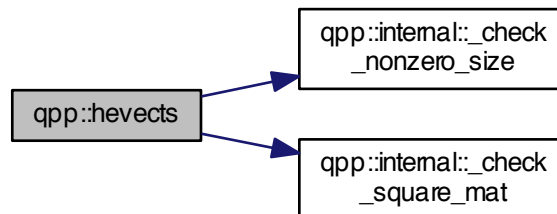
5.1.1.27 `template<typename Scalar > types::cmat qpp::hevals ( const types::DynMat< Scalar > & A )`

Here is the call graph for this function:



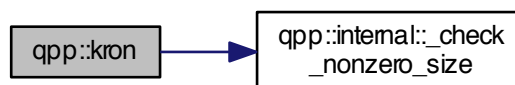
5.1.1.28 `template<typename Scalar > types::cmat qpp::hevects ( const types::DynMat< Scalar > & A )`

Here is the call graph for this function:



5.1.1.29 `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:





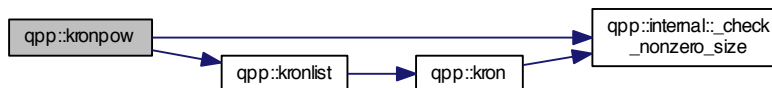
5.1.1.30 `template<typename Scalar > types::DynMat<Scalar> qpp::kronlist ( const std::vector< types::DynMat< Scalar >> & list )`

Here is the call graph for this function:



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

Here is the call graph for this function:



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

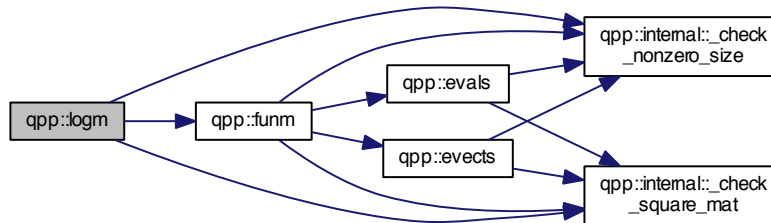
5.1.1.33 `template<typename Scalar > types::DynMat<Scalar> qpp::loadMATLABmatrix ( const std::string & mat_file, const std::string & var_name )`

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

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

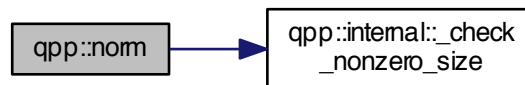
5.1.1.36 `template<typename Scalar > types::cmat qpp::logm ( const types::DynMat< Scalar > & A )`

Here is the call graph for this function:



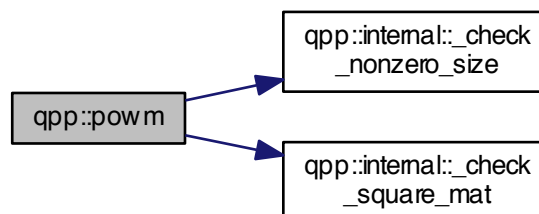
5.1.1.37 `template<typename Scalar > double qpp::norm ( const types::DynMat< Scalar > & A )`

Here is the call graph for this function:



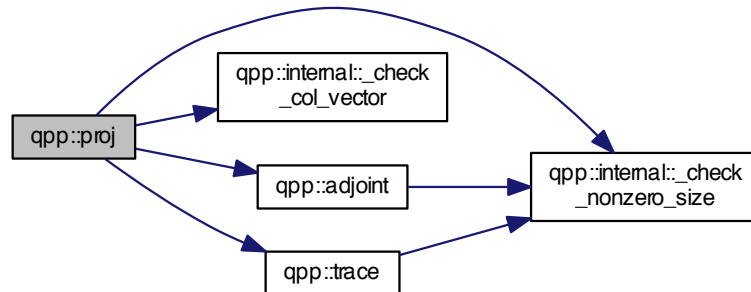
5.1.1.38 `template<typename Scalar > types::DynMat<Scalar> qpp::powm ( const types::DynMat< Scalar > & A, size_t n )`

Here is the call graph for this function:



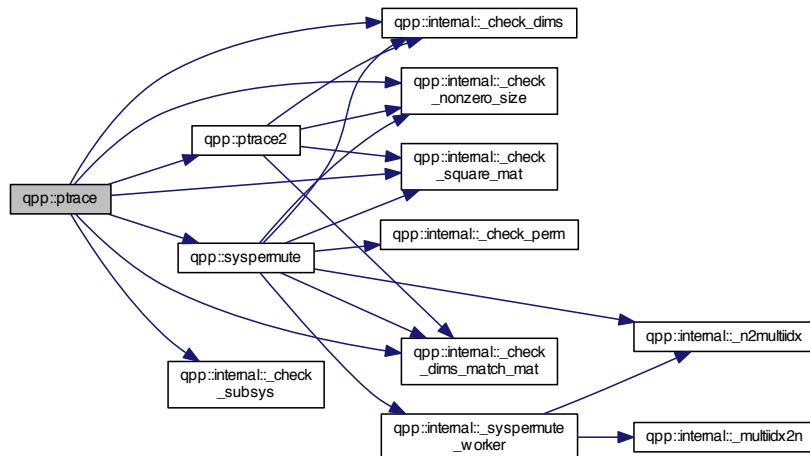
5.1.1.39 `template<typename Scalar > types::DynMat<Scalar> qpp::proj ( const types::DynMat< Scalar > & V )`

Here is the call graph for this function:



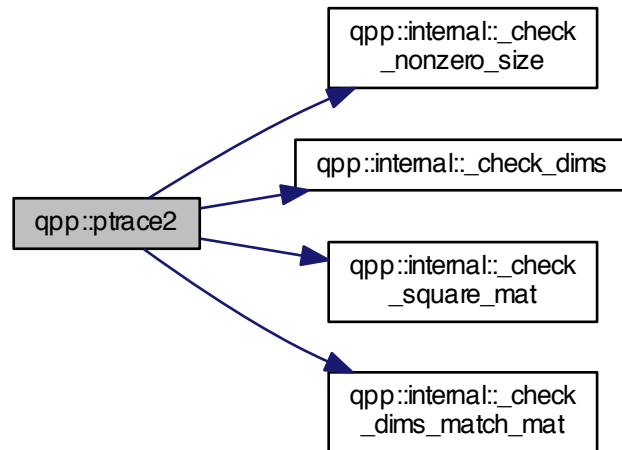
5.1.1.40 `template<typename Scalar > types::DynMat<Scalar> qpp::ptrace ( const types::DynMat< Scalar > & A, const std::vector< size_t > & subsys, const std::vector< size_t > & dims )`

Here is the call graph for this function:



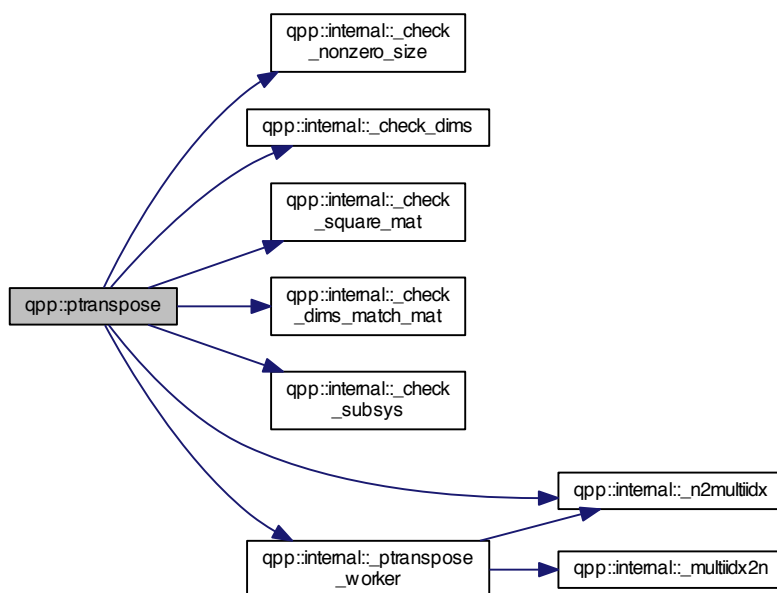
5.1.1.41 `template<typename Scalar > types::DynMat<Scalar> qpp::ptrace2 ( const types::DynMat< Scalar > & A, const std::vector< size_t > dims )`

Here is the call graph for this function:



5.1.1.42 `template<typename Scalar > types::DynMat<Scalar> qpp::ptrtranspose ( 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:



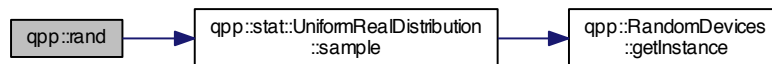
5.1.1.43 `template<typename Scalar > types::DynMat<Scalar> qpp::rand ( size_t rows, size_t cols, double a = 0, double b = 1 )`

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

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

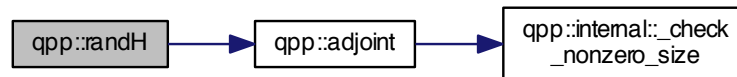
5.1.1.46 `double qpp::rand ( double a = 0, double b = 1 )`

Here is the call graph for this function:



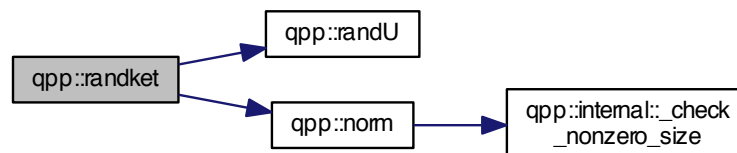
5.1.1.47 `types::cmat qpp::randH ( size_t D )`

Here is the call graph for this function:



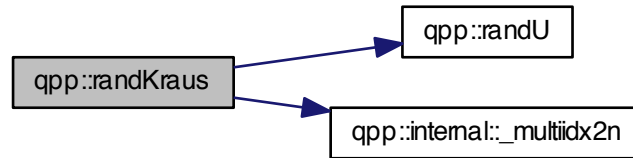
5.1.1.48 `types::cmat qpp::randket ( size_t D )`

Here is the call graph for this function:



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

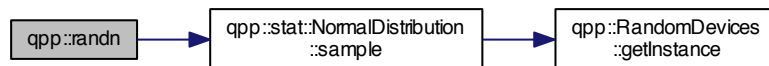
Here is the call graph for this function:



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

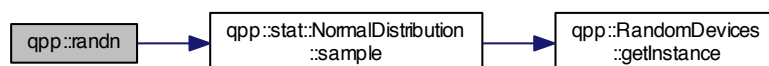
5.1.1.51 `template<> types::DynMat<double> qpp::randn ( size_t rows, size_t cols, double mean, double sigma )`

Here is the call graph for this function:



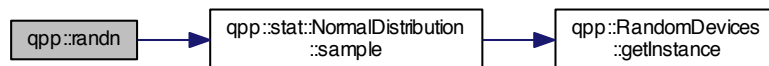
5.1.1.52 `template<> types::DynMat<types::cplx> qpp::randn ( size_t rows, size_t cols, double mean, double sigma )`

Here is the call graph for this function:



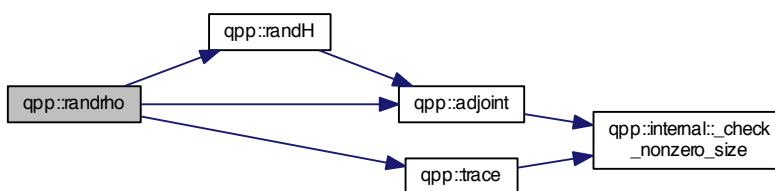
#### 5.1.1.53 `double qpp::randn ( double mean = 0, double sigma = 1 )`

Here is the call graph for this function:



#### 5.1.1.54 `types::cmat qpp::randrho ( size_t D )`

Here is the call graph for this function:



#### 5.1.1.55 `types::cmat qpp::randU ( size_t D )`

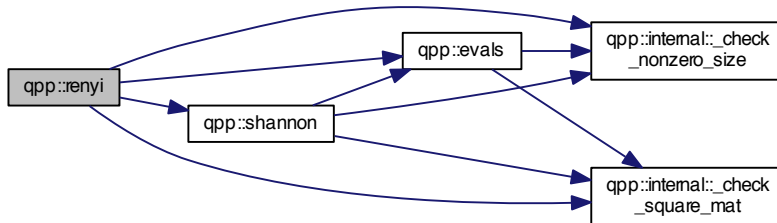
#### 5.1.1.56 `types::cmat qpp::randV ( size_t Din, size_t Dout )`

Here is the call graph for this function:



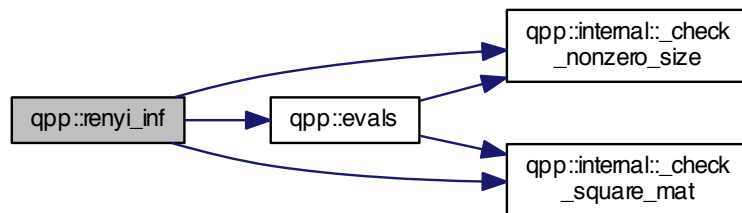
5.1.1.57 `template<typename Scalar > double qpp::renyi ( const double alpha, const types::DynMat< Scalar > & A )`

Here is the call graph for this function:



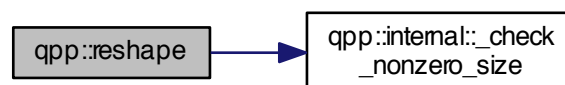
5.1.1.58 `template<typename Scalar > double qpp::renyi_inf ( const types::DynMat< Scalar > & A )`

Here is the call graph for this function:



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

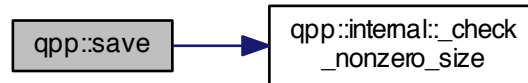
Here is the call graph for this function:





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

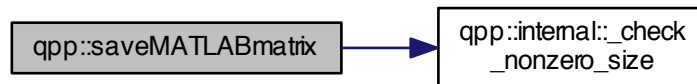
Here is the call graph for this function:



5.1.1.61 `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 )`

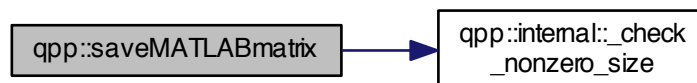
5.1.1.62 `template<> void qpp::saveMATLABmatrix ( const types::DynMat< double > & A, const std::string & mat_file, const std::string & var_name, const std::string & mode )`

Here is the call graph for this function:



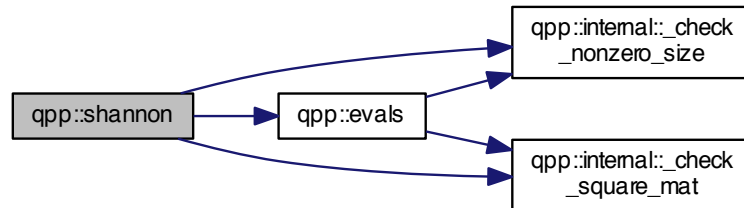
5.1.1.63 `template<> void qpp::saveMATLABmatrix ( const types::DynMat< types::cplx > & A, const std::string & mat_file, const std::string & var_name, const std::string & mode )`

Here is the call graph for this function:



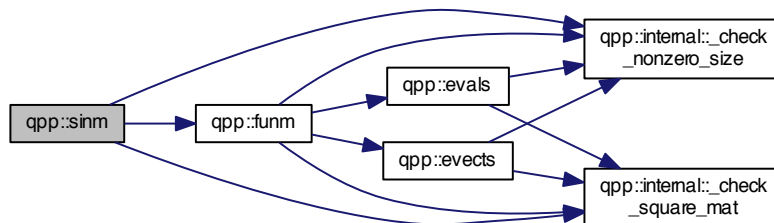
5.1.1.64 `template<typename Scalar > double qpp::shannon ( const types::DynMat< Scalar > & A )`

Here is the call graph for this function:



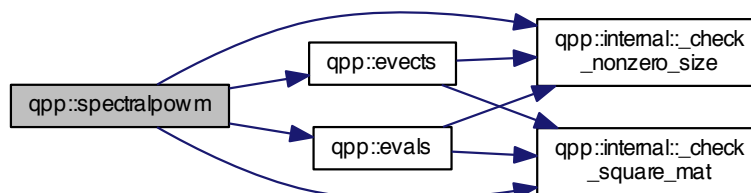
5.1.1.65 `template<typename Scalar > types::cmat qpp::sinm ( const types::DynMat< Scalar > & A )`

Here is the call graph for this function:



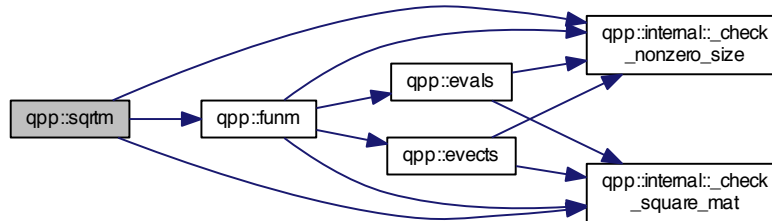
5.1.1.66 `template<typename Scalar > types::cmat qpp::spectralpowm ( const types::DynMat< Scalar > & A, const types::cplx z )`

Here is the call graph for this function:



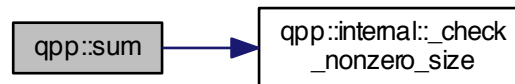
5.1.1.67 `template<typename Scalar > types::cmat qpp::sqrtm ( const types::DynMat< Scalar > & A )`

Here is the call graph for this function:



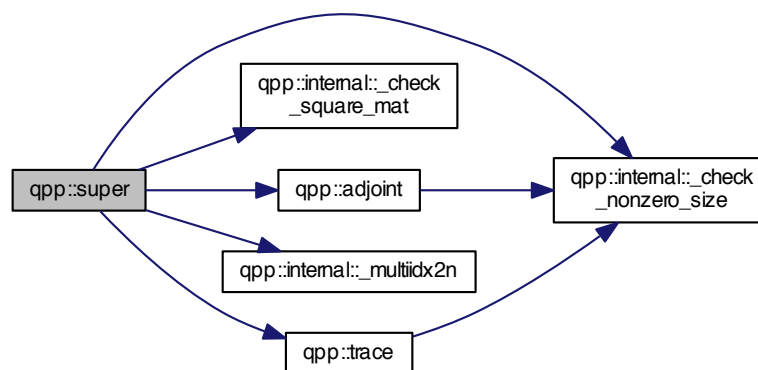
5.1.1.68 `template<typename Scalar > Scalar qpp::sum ( const types::DynMat< Scalar > & A )`

Here is the call graph for this function:



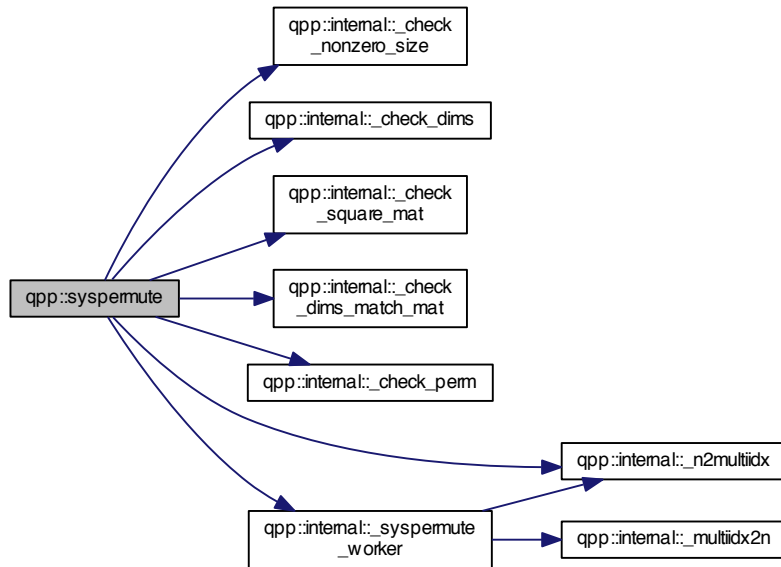
5.1.1.69 `types::cmat qpp::super ( const std::vector< types::cmat > & Ks )`

Here is the call graph for this function:



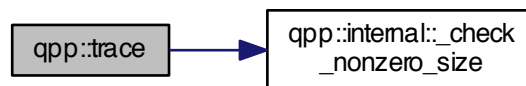
5.1.1.70 `template<typename Scalar > types::DynMat<Scalar> qpp::syspermute ( const types::DynMat< Scalar > & A,  
const std::vector< size_t > perm, const std::vector< size_t > & dims )`

Here is the call graph for this function:



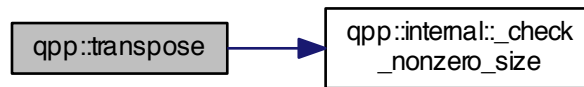
5.1.1.71 `template<typename Scalar > Scalar qpp::trace ( const types::DynMat< Scalar > & A )`

Here is the call graph for this function:



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

Here is the call graph for this function:



## 5.1.2 Variable Documentation

5.1.2.1 `const Gates& qpp::gt = Gates::getInstance()`

5.1.2.2 `RandomDevices& qpp::rdevs = RandomDevices::getInstance()`

## 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::internal Namespace Reference

## Functions

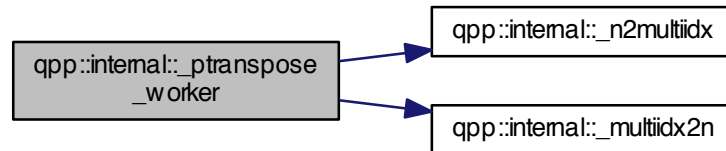
- void `_n2multiidx` (size\_t *n*, size\_t *numdims*, const size\_t \**dims*, size\_t \**result*)
- size\_t `_multiidx2n` (const size\_t \**midx*, size\_t *numdims*, const size\_t \**dims*)
- template<typename Scalar >  
bool `_check_square_mat` (const types::DynMat< Scalar > &*A*)
- template<typename 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 &*i*perm, size\_t &*j*perm, 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 &*i*perm, size\_t &*j*perm, const types::DynMat< Scalar > &*A*, types::DynMat< Scalar > &*result*)

### 5.3.1 Function Documentation

- 5.3.1.1 template<typename Scalar > bool qpp::internal::\_check\_col\_vector ( const types::DynMat< Scalar > & *A* )
- 5.3.1.2 bool qpp::internal::\_check\_dims ( const std::vector< size\_t > & *dims* )
- 5.3.1.3 template<typename Scalar > bool qpp::internal::\_check\_dims\_match\_mat ( const std::vector< size\_t > & *dims*, const types::DynMat< Scalar > & *A* )
- 5.3.1.4 bool qpp::internal::\_check\_eq\_dims ( const std::vector< size\_t > & *dims*, size\_t *dim* )
- 5.3.1.5 template<typename T > bool qpp::internal::\_check\_nonzero\_size ( const T & *x* )
- 5.3.1.6 bool qpp::internal::\_check\_perm ( const std::vector< size\_t > & *perm*, const std::vector< size\_t > & *dims* )
- 5.3.1.7 template<typename Scalar > bool qpp::internal::\_check\_row\_vector ( const types::DynMat< Scalar > & *A* )
- 5.3.1.8 template<typename Scalar > bool qpp::internal::\_check\_square\_mat ( const types::DynMat< Scalar > & *A* )
- 5.3.1.9 bool qpp::internal::\_check\_subsys ( const std::vector< size\_t > & *subsys*, const std::vector< size\_t > & *dims* )
- 5.3.1.10 template<typename Scalar > bool qpp::internal::\_check\_vector ( const types::DynMat< Scalar > & *A* )
- 5.3.1.11 size\_t qpp::internal::\_multiidx2n ( const size\_t \* *midx*, size\_t *numdims*, const size\_t \* *dims* )
- 5.3.1.12 void qpp::internal::\_n2multiidx ( size\_t *n*, size\_t *numdims*, const size\_t \* *dims*, size\_t \* *result* )

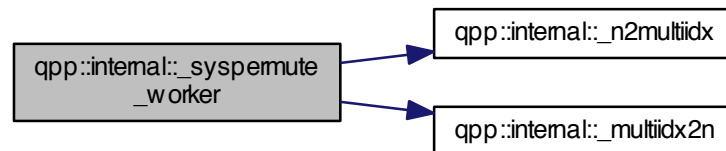
5.3.1.13 `template<typename Scalar > void qpp::internal::_pttranspose_worker ( const size_t * midxcol, size_t numdims, size_t numsubsys, const size_t * cdims, const size_t * csubsys, size_t i, size_t j, size_t & iperm, size_t & jperm, const types::DynMat< Scalar > & A, types::DynMat< Scalar > & result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

Here is the call graph for this function:



5.3.1.14 `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:



## 5.4 qpp::stat Namespace Reference

### Classes

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

## 5.5 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 >

## 5.5.1 Typedef Documentation

5.5.1.1 typedef Eigen::MatrixXcd [qpp::types::cmat](#)

5.5.1.2 typedef std::complex<double> [qpp::types::cplx](#)

5.5.1.3 typedef Eigen::MatrixXd [qpp::types::dmat](#)

5.5.1.4 template<typename Scalar > using [qpp::types::DynMat](#) = typedef Eigen::Matrix<Scalar, Eigen::Dynamic, Eigen::Dynamic>

5.5.1.5 template<typename Expression > using [qpp::types::Expression2DynMat](#) = typedef Eigen::Matrix<typename Expression::Scalar, Eigen::Dynamic, Eigen::Dynamic>

5.5.1.6 typedef Eigen::MatrixXf [qpp::types::fmat](#)

5.5.1.7 typedef Eigen::MatrixXi [qpp::types::imat](#)



## 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]

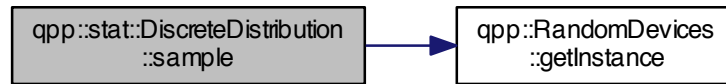
6.1.1.3 `qpp::stat::DiscreteDistribution::DiscreteDistribution ( std::vector< double > weights )` [inline]

#### 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]

Here is the call graph for this function:



## 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

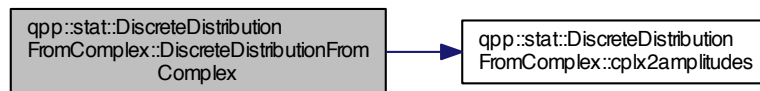
6.2.1.1 `template<typename InputIterator> qpp::stat::DiscreteDistributionFromComplex::DiscreteDistributionFromComplex ( InputIterator first, InputIterator last ) [inline]`

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:



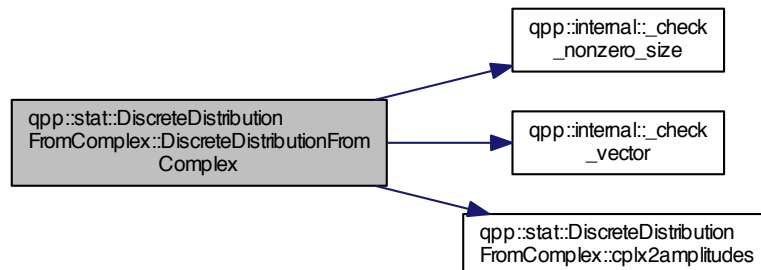
6.2.1.3 `qpp::stat::DiscreteDistributionFromComplex::DiscreteDistributionFromComplex ( std::vector< types::cplx > amplitudes ) [inline]`

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:



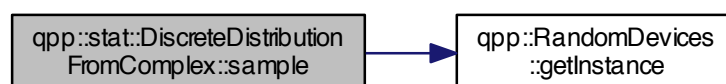
## 6.2.2 Member Function Documentation

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]`

Here is the call graph for this function:



## 6.2.3 Member Data Documentation

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

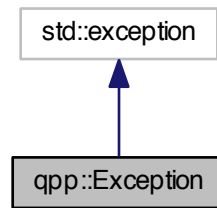
The documentation for this class was generated from the following file:

- [include/stat.h](#)

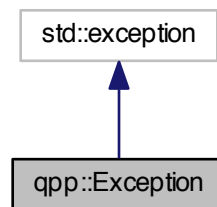
## 6.3 qpp::Exception Class Reference

```
#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]`

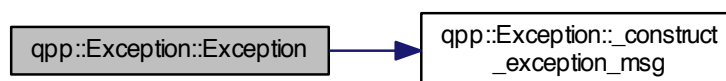
Enumerator

***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

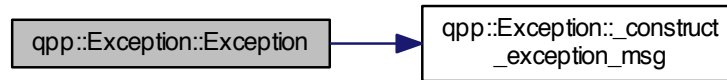
### 6.3.2.1 `qpp::Exception::Exception ( const std::string & where, const Type & type )` `[inline]`

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::Gates Class Reference

```
#include <gates.h>
```

### Public Member Functions

- [Gates](#) (const [Gates](#) &)=delete
- [Gates](#) & [operator=](#) (const [Gates](#) &)=delete
- [types::cmat Rtheta](#) (double theta) const
- [types::cmat Id](#) (size\_t D) const
- [types::cmat Zd](#) (size\_t D) const
- [types::cmat Fd](#) (size\_t D) const
- [types::cmat Xd](#) (size\_t D) const
- [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) const

## Static Public Member Functions

- static const [Gates](#) & [getInstance](#) ()

## Public Attributes

- [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](#) [CZ](#)
- [types::cmat](#) [CS](#)
- [types::cmat](#) [CNOTba](#)
- [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](#)

## Private Member Functions

- [Gates](#) ()
- void [\\_init\\_gates](#) ()

### 6.4.1 Constructor & Destructor Documentation

#### 6.4.1.1 `qpp::Gates::Gates ( )` `[inline]`, `[private]`

Here is the call graph for this function:





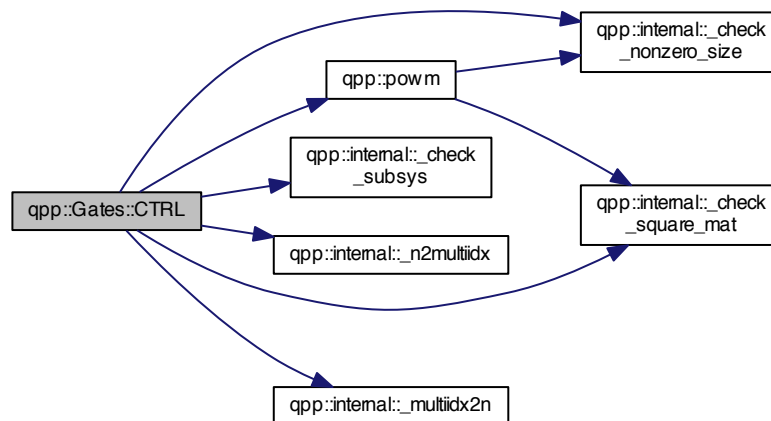
6.4.1.2 `qpp::Gates::Gates ( const Gates & )` `[delete]`

## 6.4.2 Member Function Documentation

6.4.2.1 `void qpp::Gates::_init_gates ( )` `[inline],[private]`

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

Here is the call graph for this function:



6.4.2.3 `types::cmat qpp::Gates::Fd ( size_t D ) const` `[inline]`

Here is the call graph for this function:



6.4.2.4 `static const Gates& qpp::Gates::getInstance ( )` `[inline],[static]`

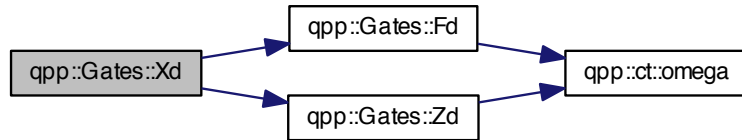
6.4.2.5 `types::cmat qpp::Gates::Id ( size_t D ) const` `[inline]`

6.4.2.6 `Gates& qpp::Gates::operator= ( const Gates & )` `[delete]`

6.4.2.7 `types::cmat qpp::Gates::Rtheta ( double theta ) const` `[inline]`

#### 6.4.2.8 `types::cmat qpp::Gates::Xd ( size_t D ) const [inline]`

Here is the call graph for this function:



#### 6.4.2.9 `types::cmat qpp::Gates::Zd ( size_t D ) const [inline]`

Here is the call graph for this function:



### 6.4.3 Member Data Documentation

6.4.3.1 `types::cmat qpp::Gates::b00`

6.4.3.2 `types::cmat qpp::Gates::b01`

6.4.3.3 `types::cmat qpp::Gates::b10`

6.4.3.4 `types::cmat qpp::Gates::b11`

6.4.3.5 `types::cmat qpp::Gates::CNOTab`

6.4.3.6 `types::cmat qpp::Gates::CNOTba`

6.4.3.7 `types::cmat qpp::Gates::CS`

6.4.3.8 `types::cmat qpp::Gates::CZ`

6.4.3.9 `types::cmat qpp::Gates::FRED`

6.4.3.10 `types::cmat qpp::Gates::H`

6.4.3.11 `types::cmat qpp::Gates::Id2`

- 6.4.3.12 `types::cmat qpp::Gates::S`
- 6.4.3.13 `types::cmat qpp::Gates::SWAP`
- 6.4.3.14 `types::cmat qpp::Gates::T`
- 6.4.3.15 `types::cmat qpp::Gates::TOF`
- 6.4.3.16 `types::cmat qpp::Gates::X`
- 6.4.3.17 `types::cmat qpp::Gates::x0`
- 6.4.3.18 `types::cmat qpp::Gates::x1`
- 6.4.3.19 `types::cmat qpp::Gates::Y`
- 6.4.3.20 `types::cmat qpp::Gates::y0`
- 6.4.3.21 `types::cmat qpp::Gates::y1`
- 6.4.3.22 `types::cmat qpp::Gates::Z`
- 6.4.3.23 `types::cmat qpp::Gates::z0`
- 6.4.3.24 `types::cmat qpp::Gates::z1`

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

- [include/gates.h](#)

## 6.5 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`

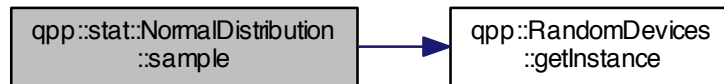
### 6.5.1 Constructor & Destructor Documentation

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

### 6.5.2 Member Function Documentation

### 6.5.2.1 double qpp::stat::NormalDistribution::sample ( ) [inline]

Here is the call graph for this function:



## 6.5.3 Member Data Documentation

### 6.5.3.1 std::normal\_distribution qpp::stat::NormalDistribution::\_d [protected]

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

- include/[stat.h](#)

## 6.6 qpp::RandomDevices Class Reference

```
#include <randevs.h>
```

### Public Member Functions

- [RandomDevices](#) (const [RandomDevices](#) &)=delete
- [RandomDevices](#) & operator= (const [RandomDevices](#) &)=delete
- virtual ~[RandomDevices](#) ()=default

### Static Public Member Functions

- static [RandomDevices](#) & [getInstance](#) ()

### Public Attributes

- std::random\_device [\\_rd](#)
- std::mt19937 [\\_rng](#)

### Private Member Functions

- [RandomDevices](#) ()

## 6.6.1 Constructor & Destructor Documentation

### 6.6.1.1 qpp::RandomDevices::RandomDevices ( ) [inline],[private]

6.6.1.2 `qpp::RandomDevices::RandomDevices ( const RandomDevices & ) [delete]`

6.6.1.3 `virtual qpp::RandomDevices::~~RandomDevices ( ) [virtual],[default]`

## 6.6.2 Member Function Documentation

6.6.2.1 `static RandomDevices& qpp::RandomDevices::getInstance ( ) [inline],[static]`

6.6.2.2 `RandomDevices& qpp::RandomDevices::operator= ( const RandomDevices & ) [delete]`

## 6.6.3 Member Data Documentation

6.6.3.1 `std::random_device qpp::RandomDevices::_rd`

6.6.3.2 `std::mt19937 qpp::RandomDevices::_rng`

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

- [include/randevs.h](#)

## 6.7 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)`

## 6.7.1 Constructor & Destructor Documentation

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

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

## 6.7.2 Member Function Documentation

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

6.7.2.2 void qpp::Timer::tic ( ) [inline]

6.7.2.3 void qpp::Timer::toc ( ) [inline]

### 6.7.3 Friends And Related Function Documentation

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

### 6.7.4 Member Data Documentation

6.7.4.1 std::chrono::high\_resolution\_clock::time\_point qpp::Timer::\_end [protected]

6.7.4.2 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

## 6.8 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

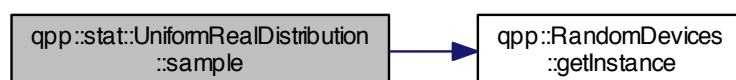
### 6.8.1 Constructor & Destructor Documentation

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

### 6.8.2 Member Function Documentation

6.8.2.1 double qpp::stat::UniformRealDistribution::sample ( ) [inline]

Here is the call graph for this function:



### 6.8.3 Member Data Documentation

#### 6.8.3.1 `std::uniform_real_distribution qpp::stat::UniformRealDistribution::_d` [protected]

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

- [include/stat.h](#)





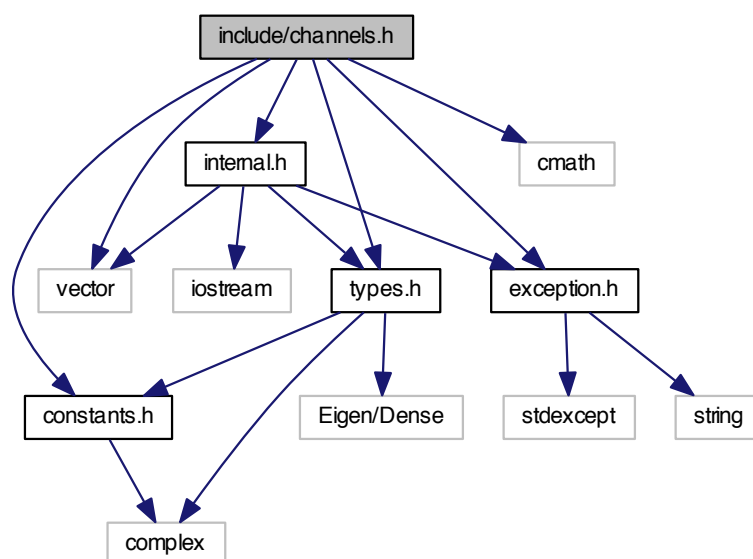
## Chapter 7

# File Documentation

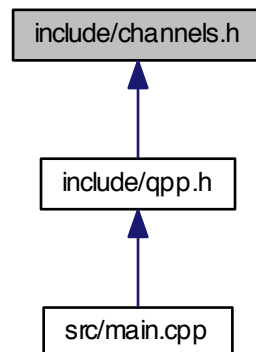
### 7.1 include/channels.h File Reference

```
#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](#)

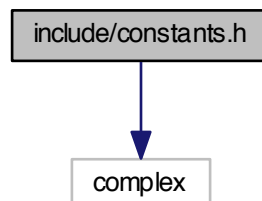
## Functions

- `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`)

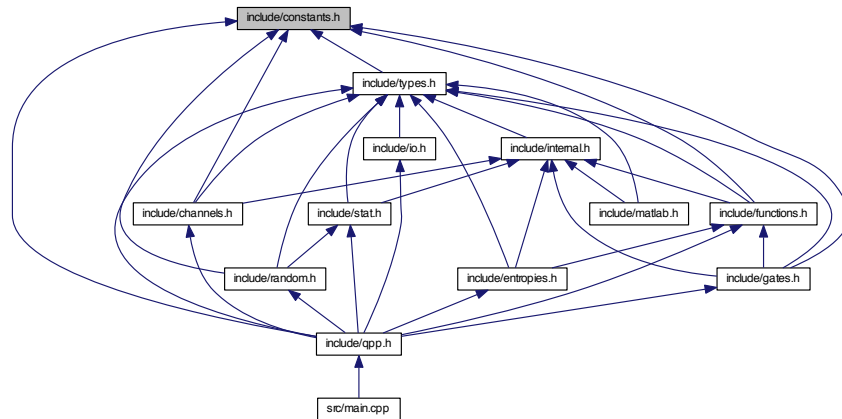
## 7.2 include/constants.h File Reference

```
#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`

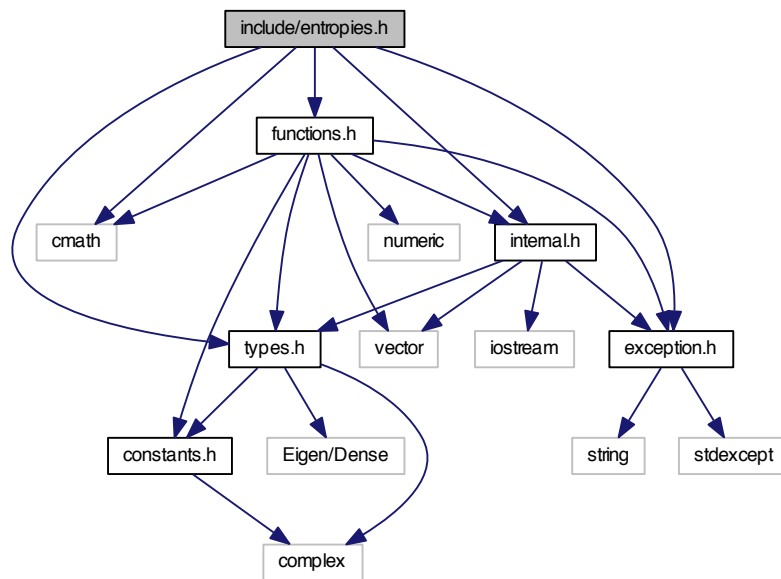
## 7.3 include/entropies.h File Reference

```

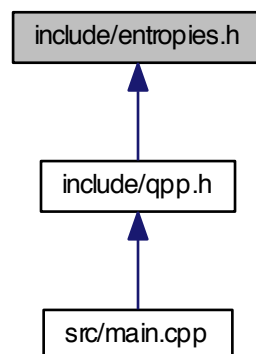
#include <cmath>
#include "types.h"
#include "functions.h"
#include "internal.h"
#include "exception.h"

```

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)`

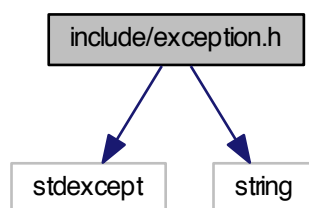
- `template<typename Scalar >`  
`double qpp::renyi (const double alpha, const types::DynMat< Scalar > &A)`
- `template<typename Scalar >`  
`double qpp::renyi\_inf (const types::DynMat< Scalar > &A)`

## 7.4 include/exception.h File Reference

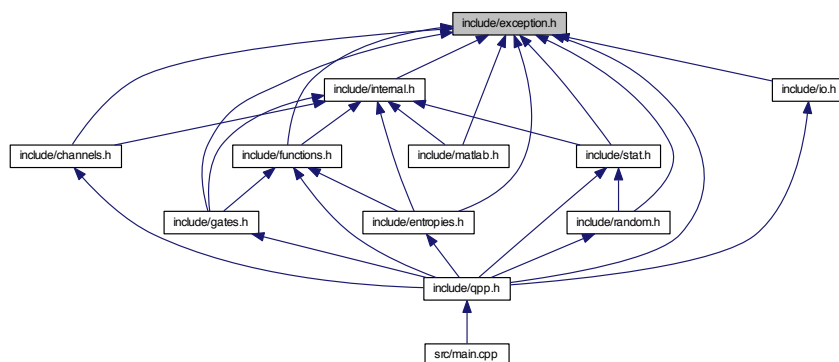
```
#include <stdexcept>
```

```
#include <string>
```

Include dependency graph for exception.h:



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



### Classes

- class [qpp::Exception](#)

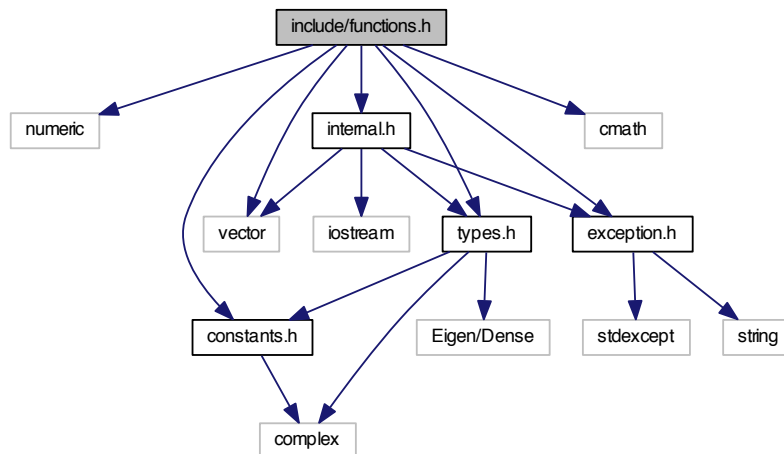
### Namespaces

- [qpp](#)

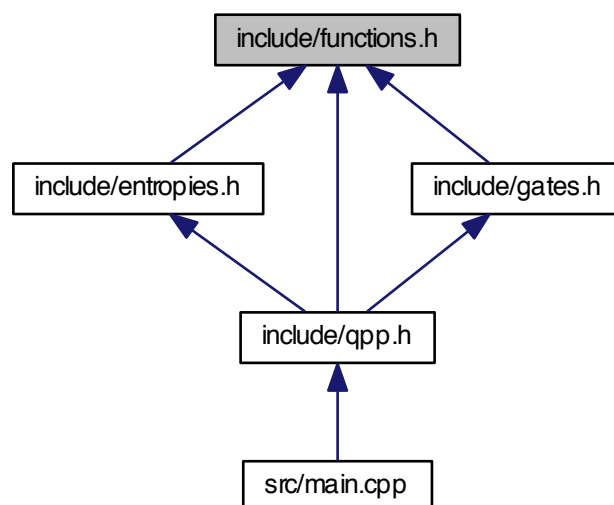
## 7.5 include/functions.h File Reference

```
#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)`

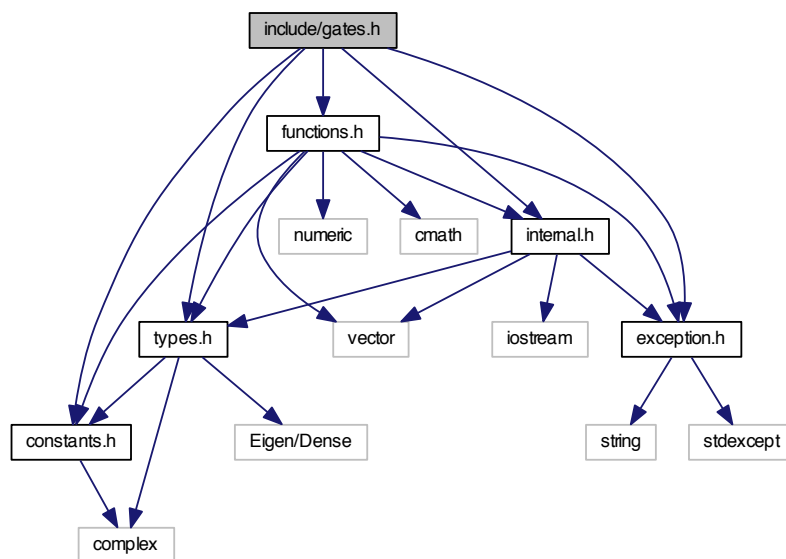
- `template<typename Scalar >`  
`types::DynMat< Scalar > qpp::reshape (const types::DynMat< Scalar > &A, size_t rows, size_t cols)`
- `template<typename Scalar >`  
`types::DynMat< Scalar > qpp::syspermute (const types::DynMat< Scalar > &A, const std::vector< size_t > perm, const std::vector< size_t > &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

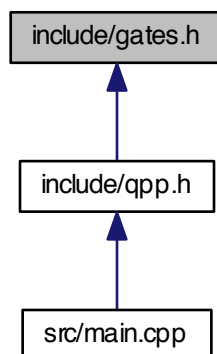
```
#include "types.h"
#include "constants.h"
#include "functions.h"
#include "internal.h"
#include "exception.h"
```



Include dependency graph for gates.h:



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



## Classes

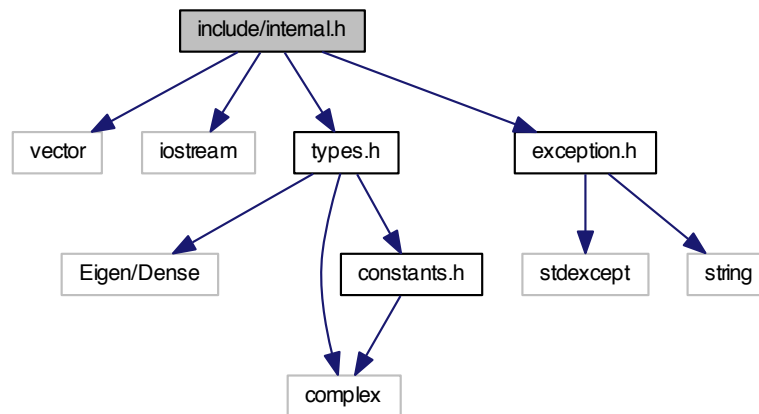
- class `qpp::Gates`

## Namespaces

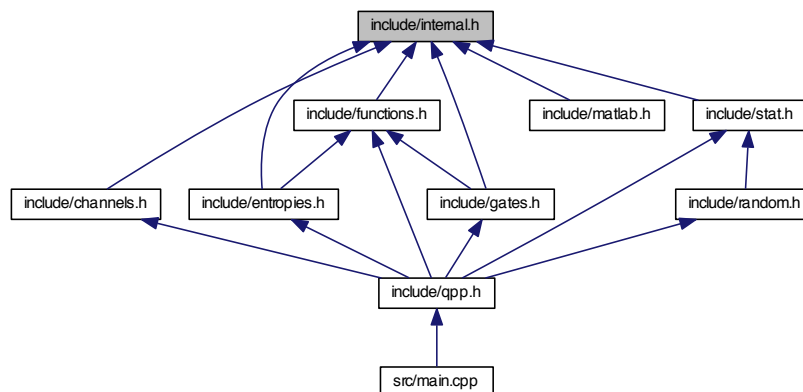
- `qpp`

## 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

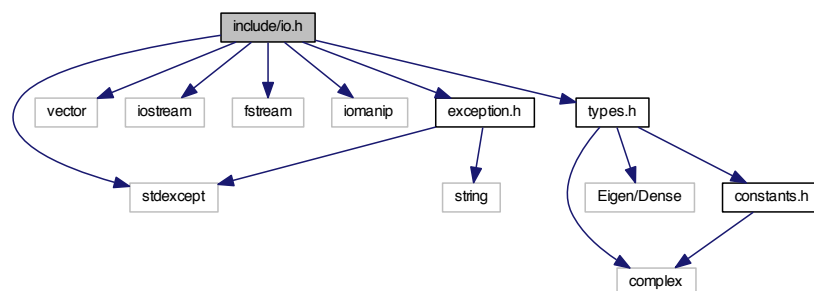
- void `qpp::internal::_n2multiidx` (size\_t n, size\_t numdims, const size\_t \*dims, size\_t \*result)

- `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`)

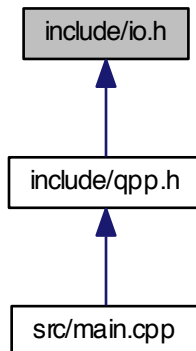
## 7.8 include/io.h File Reference

```
#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

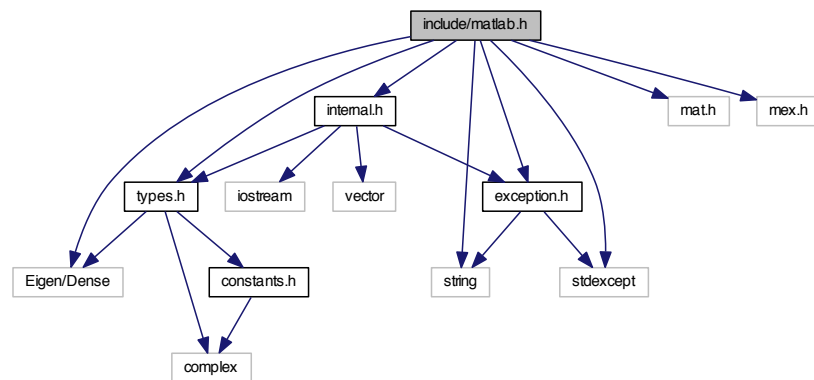
- `template<typename T >`  
`void qpp::disp (const T &x, const std::string &separator=" ", const std::string &start="[", const std::string &end="]", std::ostream &os=std::cout)`
- `template<typename T >`  
`void qpp::displn (const T &x, const std::string &separator=" ", const std::string &start="[",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

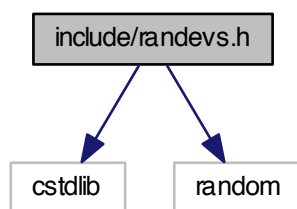
- `template<typename Scalar >`  
`types::DynMat< Scalar > qpp::loadMATLABmatrix (const std::string &mat_file, const std::string &var_name)`
- `template<>`  
`types::DynMat< double > qpp::loadMATLABmatrix (const std::string &mat_file, const std::string &var_name)`
- `template<>`  
`types::DynMat< types::cplx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

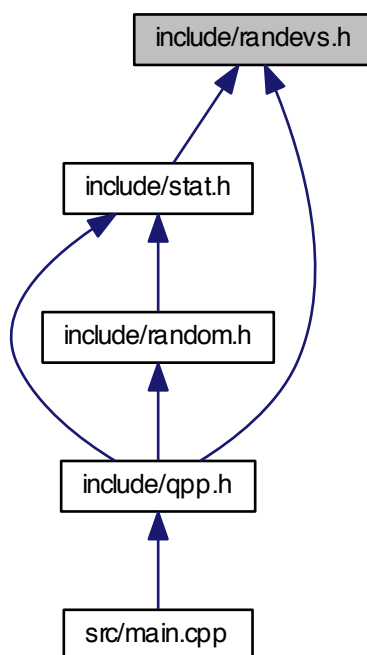
```
#include "types.h"
```



Include dependency graph for randevs.h:



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



## Classes

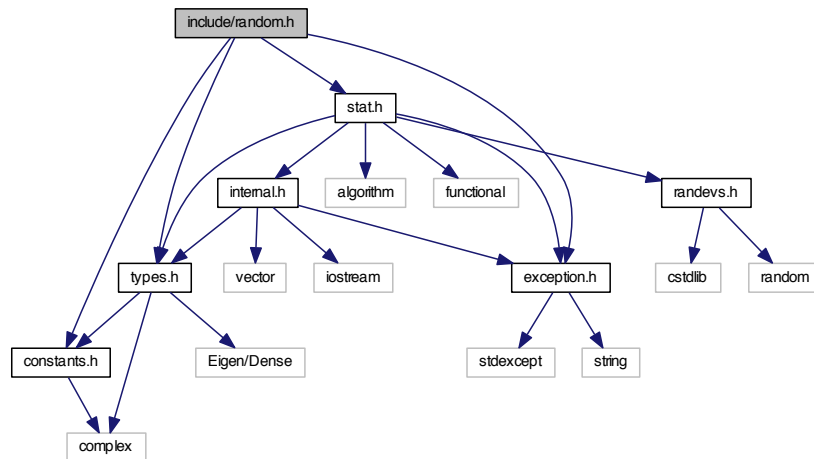
- class [qpp::RandomDevices](#)

## Namespaces

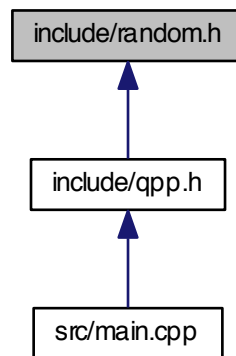
- [qpp](#)

## 7.12 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

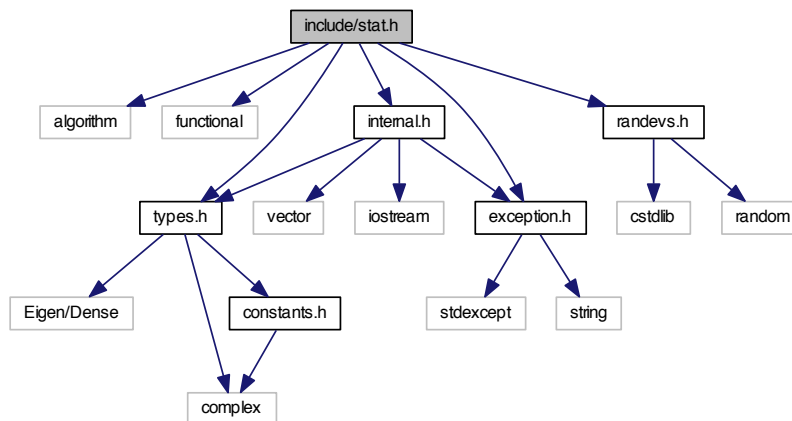
- `template<typename Scalar >`



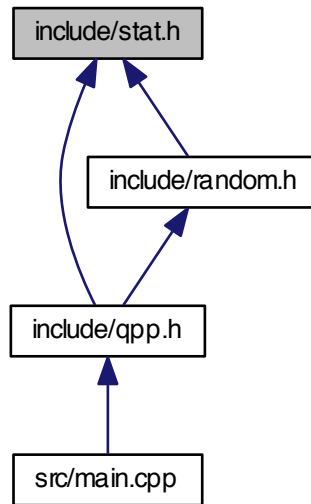
- types::DynMat< Scalar > [qpp::rand](#) (size\_t rows, size\_t cols, double a=0, double b=1)
- template<>  
types::DynMat< double > [qpp::rand](#) (size\_t rows, size\_t cols, double a, double b)
  - template<>  
types::DynMat< types::cplx > [qpp::rand](#) (size\_t rows, size\_t cols, double a, double 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)

## 7.13 include/stat.h File Reference

```
#include <algorithm>
#include <functional>
#include "types.h"
#include "internal.h"
#include "exception.h"
#include "randevs.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](#)

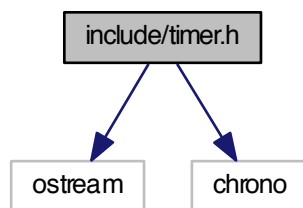
## Namespaces

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

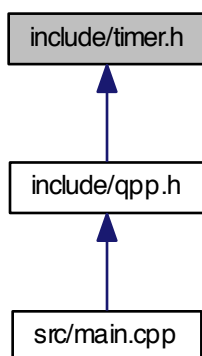
## 7.14 include/timer.h File Reference

```
#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](#)

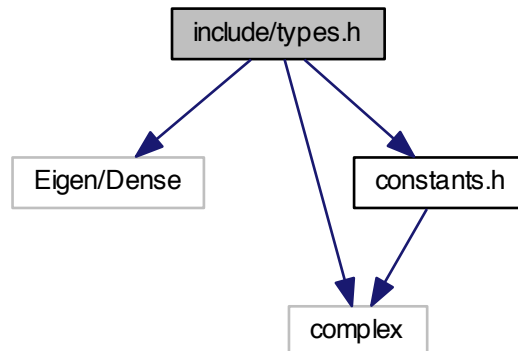
## Namespaces

- [qpp](#)

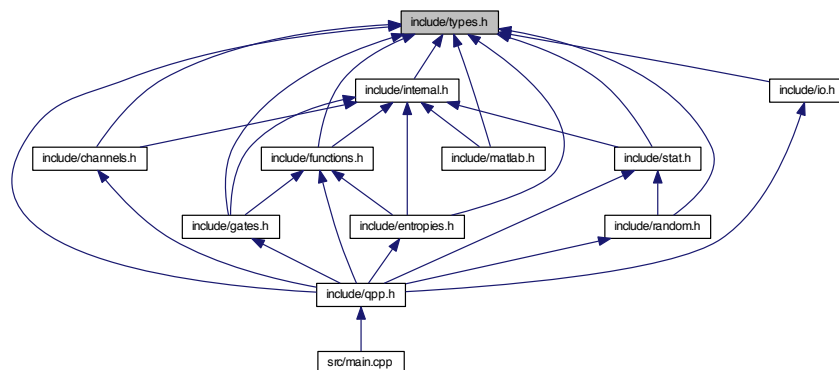
## 7.15 include/types.h File Reference

```
#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

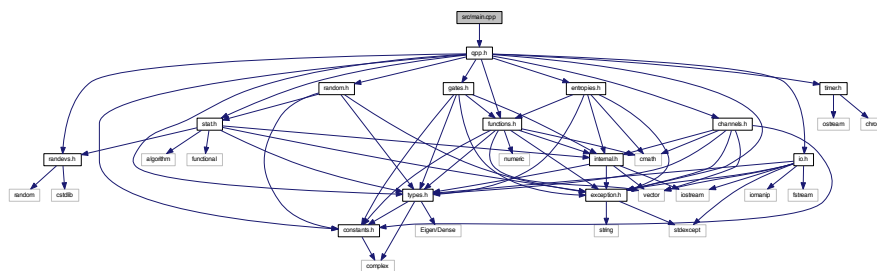
- `qpp`
- `qpp::types`

## 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 >`

- ## 7.16 src/main.cpp File Reference

Include dependency graph for main.cpp:



- `template<typename T >`  
`types::Expression2DynMat< T > test (const Eigen::MatrixBase< T > &A)`
- `int main ()`

### 7.16.1.1 int main ( )

```
7.16.1.2 template<typename T> types::Expression2DynMat<T> test ( const Eigen::MatrixBase< T > & A )
```