

qpp
0.1

Generated by Doxygen 1.8.5

Sat Apr 5 2014 05:14:44

Contents

1	Namespace Index	1
1.1	Namespace List	1
2	Hierarchical Index	3
2.1	Class Hierarchy	3
3	Class Index	5
3.1	Class List	5
4	File Index	7
4.1	File List	7
5	Namespace Documentation	9
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	12
5.1.1.4	anticomm	13
5.1.1.5	comm	13
5.1.1.6	conjugate	13
5.1.1.7	cosm	14
5.1.1.8	disp	14
5.1.1.9	disp	14
5.1.1.10	disp	14
5.1.1.11	disp	14
5.1.1.12	displn	15
5.1.1.13	displn	15
5.1.1.14	displn	15
5.1.1.15	displn	16
5.1.1.16	dya	16
5.1.1.17	evals	16
5.1.1.18	evects	17

5.1.1.19	expandout	17
5.1.1.20	expm	18
5.1.1.21	fun	18
5.1.1.22	funm	18
5.1.1.23	hevals	19
5.1.1.24	hevects	20
5.1.1.25	kron	20
5.1.1.26	kronlist	20
5.1.1.27	kronpow	21
5.1.1.28	load	21
5.1.1.29	loadMATLABmatrix	21
5.1.1.30	loadMATLABmatrix	21
5.1.1.31	loadMATLABmatrix	21
5.1.1.32	logm	21
5.1.1.33	norm	22
5.1.1.34	powm	22
5.1.1.35	proj	22
5.1.1.36	ptrace	23
5.1.1.37	ptrace2	23
5.1.1.38	ptranspose	24
5.1.1.39	rand	24
5.1.1.40	rand	24
5.1.1.41	rand	24
5.1.1.42	rand	24
5.1.1.43	randH	25
5.1.1.44	randket	25
5.1.1.45	randn	25
5.1.1.46	randn	25
5.1.1.47	randn	26
5.1.1.48	randn	26
5.1.1.49	randrho	26
5.1.1.50	randU	26
5.1.1.51	renyi	27
5.1.1.52	renyi_inf	27
5.1.1.53	reshape	27
5.1.1.54	save	28
5.1.1.55	saveMATLABmatrix	28
5.1.1.56	saveMATLABmatrix	28
5.1.1.57	saveMATLABmatrix	28
5.1.1.58	shannon	29

5.1.1.59	sinm	29
5.1.1.60	spectralpowm	29
5.1.1.61	sqrtm	30
5.1.1.62	sum	30
5.1.1.63	syspermute	31
5.1.1.64	trace	31
5.1.1.65	transpose	32
5.2	qpp::ct Namespace Reference	32
5.2.1	Function Documentation	32
5.2.1.1	omega	32
5.2.2	Variable Documentation	32
5.2.2.1	chop	32
5.2.2.2	ee	32
5.2.2.3	ii	32
5.2.2.4	pi	32
5.3	qpp::gt Namespace Reference	32
5.3.1	Function Documentation	33
5.3.1.1	_init_gates	33
5.3.1.2	CTRL	33
5.3.1.3	Fd	34
5.3.1.4	Id	34
5.3.1.5	Rtheta	34
5.3.1.6	TOF	34
5.3.1.7	Xd	34
5.3.1.8	Zd	34
5.3.2	Variable Documentation	34
5.3.2.1	CNOT	35
5.3.2.2	CP	35
5.3.2.3	H	35
5.3.2.4	Id2	35
5.3.2.5	S	35
5.3.2.6	T	35
5.3.2.7	TOF	35
5.3.2.8	X	35
5.3.2.9	Y	35
5.3.2.10	Z	35
5.4	qpp::internal Namespace Reference	35
5.4.1	Function Documentation	35
5.4.1.1	_check_col_vector	35
5.4.1.2	_check_dims	36

5.4.1.3	_check_dims_match_mat	36
5.4.1.4	_check_eq_dims	36
5.4.1.5	_check_nonzero_size	36
5.4.1.6	_check_perm	36
5.4.1.7	_check_row_vector	36
5.4.1.8	_check_square_mat	36
5.4.1.9	_check_subsys	36
5.4.1.10	_check_vector	36
5.4.1.11	_multiidx2n	36
5.4.1.12	_n2multiidx	36
5.4.1.13	_pttranspose_worker	36
5.4.1.14	_syspermute_worker	37
5.5	qpp::stat Namespace Reference	37
5.5.1	Variable Documentation	37
5.5.1.1	_rd	37
5.5.1.2	_rng	37
5.6	qpp::types Namespace Reference	37
5.6.1	Typedef Documentation	38
5.6.1.1	cmat	38
5.6.1.2	cplx	38
5.6.1.3	dmat	38
5.6.1.4	DynMat	38
5.6.1.5	Expression2DynMat	38
5.6.1.6	fmat	38
5.6.1.7	imat	38
6	Class Documentation	39
6.1	qpp::stat::DiscreteDistribution Class Reference	39
6.1.1	Constructor & Destructor Documentation	39
6.1.1.1	DiscreteDistribution	39
6.1.1.2	DiscreteDistribution	39
6.1.1.3	DiscreteDistribution	39
6.1.2	Member Function Documentation	39
6.1.2.1	probabilities	39
6.1.2.2	sample	39
6.1.3	Member Data Documentation	39
6.1.3.1	_d	39
6.2	qpp::stat::DiscreteDistributionFromComplex Class Reference	40
6.2.1	Constructor & Destructor Documentation	40
6.2.1.1	DiscreteDistributionFromComplex	40

6.2.1.2	DiscreteDistributionFromComplex	41
6.2.1.3	DiscreteDistributionFromComplex	41
6.2.1.4	DiscreteDistributionFromComplex	41
6.2.2	Member Function Documentation	41
6.2.2.1	cplx2amplitudes	42
6.2.2.2	probabilities	42
6.2.2.3	sample	42
6.2.3	Member Data Documentation	42
6.2.3.1	_d	42
6.3	qpp::Exception Class Reference	42
6.3.1	Member Enumeration Documentation	43
6.3.1.1	Type	43
6.3.2	Constructor & Destructor Documentation	44
6.3.2.1	Exception	44
6.3.2.2	Exception	44
6.3.2.3	~Exception	44
6.3.3	Member Function Documentation	44
6.3.3.1	_construct_exception_msg	44
6.3.3.2	what	44
6.3.4	Member Data Documentation	44
6.3.4.1	_custom	44
6.3.4.2	_msg	44
6.3.4.3	_type	44
6.3.4.4	_where	44
6.4	qpp::stat::NormalDistribution Class Reference	45
6.4.1	Constructor & Destructor Documentation	45
6.4.1.1	NormalDistribution	45
6.4.2	Member Function Documentation	45
6.4.2.1	sample	45
6.4.3	Member Data Documentation	45
6.4.3.1	_d	45
6.5	qpp::Timer Class Reference	45
6.5.1	Constructor & Destructor Documentation	46
6.5.1.1	Timer	46
6.5.1.2	~Timer	46
6.5.2	Member Function Documentation	46
6.5.2.1	seconds	46
6.5.2.2	tic	46
6.5.2.3	toc	46
6.5.3	Friends And Related Function Documentation	46

6.5.3.1	<code>operator<<</code>	46
6.5.4	Member Data Documentation	46
6.5.4.1	<code>_end</code>	46
6.5.4.2	<code>_start</code>	46
6.6	<code>qpp::stat::UniformRealDistribution</code> Class Reference	46
6.6.1	Constructor & Destructor Documentation	46
6.6.1.1	<code>UniformRealDistribution</code>	46
6.6.2	Member Function Documentation	46
6.6.2.1	<code>sample</code>	46
6.6.3	Member Data Documentation	46
6.6.3.1	<code>_d</code>	47
7	File Documentation	49
7.1	<code>include/constants.h</code> File Reference	49
7.2	<code>include/entropies.h</code> File Reference	50
7.3	<code>include/exception.h</code> File Reference	52
7.4	<code>include/functions.h</code> File Reference	53
7.5	<code>include/gates.h</code> File Reference	55
7.6	<code>include/internal.h</code> File Reference	57
7.7	<code>include/io.h</code> File Reference	58
7.8	<code>include/matlab.h</code> File Reference	59
7.9	<code>include/qpp.h</code> File Reference	60
7.10	<code>include/random.h</code> File Reference	61
7.11	<code>include/stat.h</code> File Reference	63
7.12	<code>include/timer.h</code> File Reference	64
7.13	<code>include/types.h</code> File Reference	65
7.14	<code>src/main.cpp</code> File Reference	67
7.14.1	Function Documentation	67
7.14.1.1	<code>main</code>	68

Chapter 1

Namespace Index

1.1 Namespace List

Here is a list of all namespaces with brief descriptions:

qpp	9
qpp::ct	32
qpp::gt	32
qpp::internal	35
qpp::stat	37
qpp::types	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::stat::NormalDistribution	45
qpp::Timer	45
qpp::stat::UniformRealDistribution	46

Chapter 3

Class Index

3.1 Class List

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

qpp::stat::DiscreteDistribution	39
qpp::stat::DiscreteDistributionFromComplex	40
qpp::Exception	42
qpp::stat::NormalDistribution	45
qpp::Timer	45
qpp::stat::UniformRealDistribution	46

Chapter 4

File Index

4.1 File List

Here is a list of all files with brief descriptions:

include/	constants.h	49
include/	entropies.h	50
include/	exception.h	52
include/	functions.h	53
include/	gates.h	55
include/	internal.h	57
include/	io.h	58
include/	matlab.h	59
include/	qpp.h	60
include/	random.h	61
include/	stat.h	63
include/	timer.h	64
include/	types.h	65
src/	main.cpp	67

Chapter 5

Namespace Documentation

5.1 qpp Namespace Reference

Namespaces

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

Classes

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

Functions

- `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 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 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 >` `dya` (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 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 randH (size_t D)`
- `types::cmat randket (size_t D)`
- `types::cmat randrho (size_t D)`

5.1.1 Function Documentation

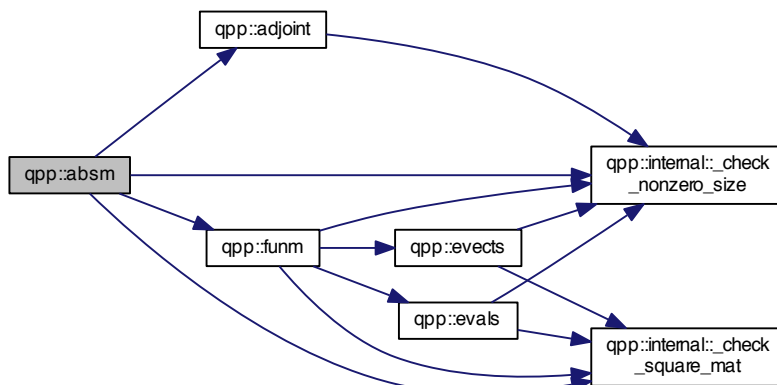
5.1.1.1 `int qpp::_init ()`

Here is the call graph for this function:



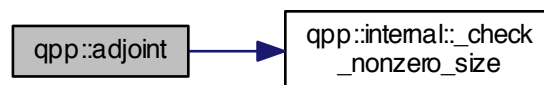
5.1.1.2 `template<typename Scalar > types::cmat qpp::absm (const types::DynMat< Scalar > & A)`

Here is the call graph for this function:



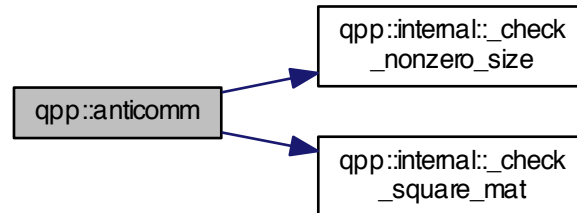
5.1.1.3 `template<typename Scalar > types::DynMat<Scalar> qpp::adjoint (const types::DynMat< Scalar > & A)`

Here is the call graph for this function:



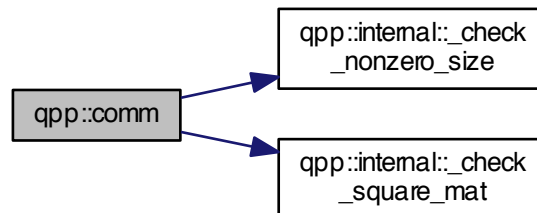
5.1.1.4 `template<typename Scalar > types::DynMat<Scalar> qpp::anticomm (const types::DynMat< Scalar > & A, const types::DynMat< Scalar > & B)`

Here is the call graph for this function:



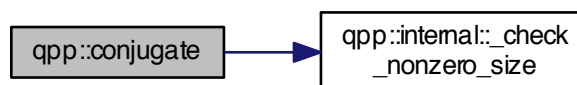
5.1.1.5 `template<typename Scalar > types::DynMat<Scalar> qpp::comm (const types::DynMat< Scalar > & A, const types::DynMat< Scalar > & B)`

Here is the call graph for this function:



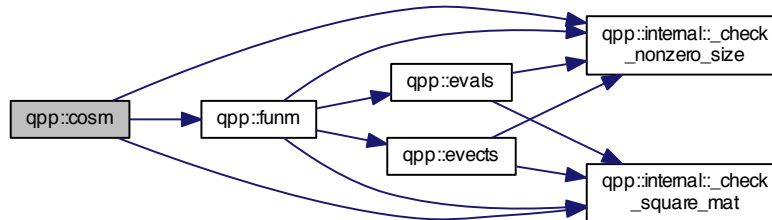
5.1.1.6 `template<typename Scalar > types::DynMat<Scalar> qpp::conjugate (const types::DynMat< Scalar > & A)`

Here is the call graph for this function:



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

Here is the call graph for this function:



5.1.1.8 `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)`

5.1.1.9 `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)`

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

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

Here is the call graph for this function:



5.1.1.12 `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:



5.1.1.13 `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:



5.1.1.14 `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:



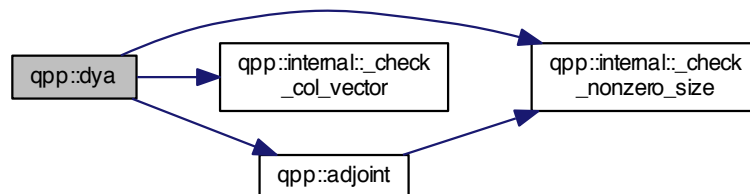
5.1.1.15 `void qpp::displn (const types::cplx c, double chop = ct::chop, std::ostream & os = std::cout)`
`[inline]`

Here is the call graph for this function:



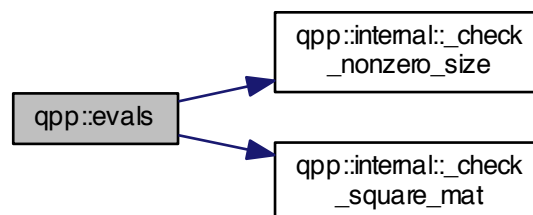
5.1.1.16 `template<typename Scalar> types::DynMat<Scalar> qpp::dya (const types::DynMat<Scalar> & V)`

Here is the call graph for this function:



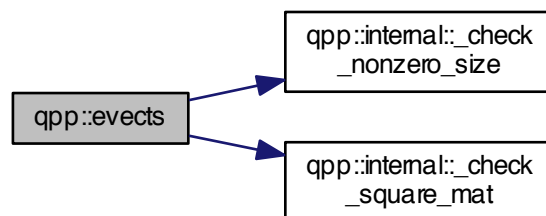
5.1.1.17 `template<typename Scalar> types::cmat qpp::evals (const types::DynMat<Scalar> & A)`

Here is the call graph for this function:



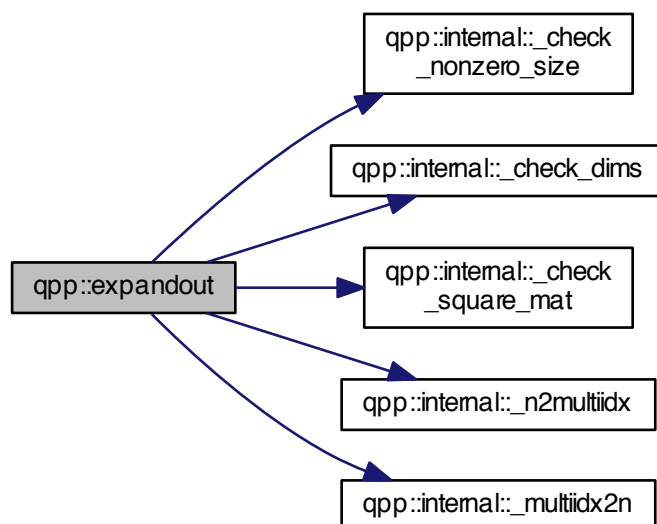
5.1.1.18 `template<typename Scalar > types::cmat qpp::evecs (const types::DynMat< Scalar > & A)`

Here is the call graph for this function:



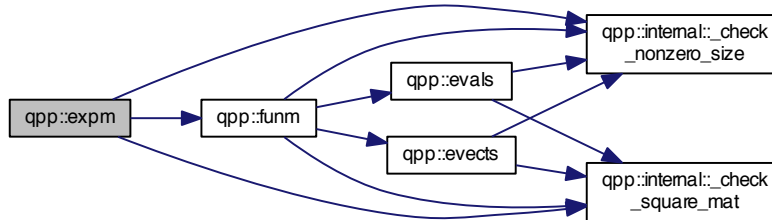
5.1.1.19 `template<typename Scalar > types::DynMat<Scalar> qpp::expandout (const types::DynMat< Scalar > & A, size_t pos, const std::vector< size_t > & dims)`

Here is the call graph for this function:



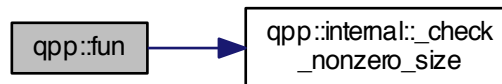
5.1.1.20 `template<typename Scalar > types::cmat qpp::expm (const types::DynMat< Scalar > & A)`

Here is the call graph for this function:



5.1.1.21 `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:



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

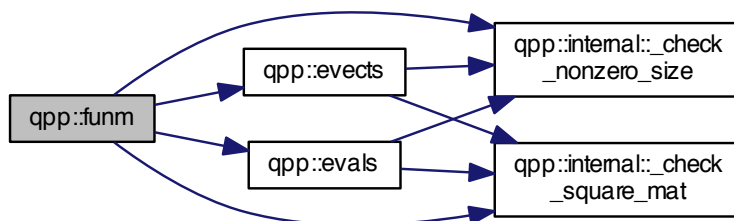
Parameters

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

Returns

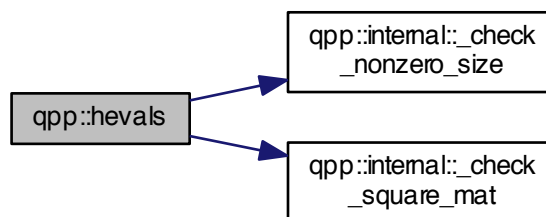
[types::cmat](#)

Here is the call graph for this function:



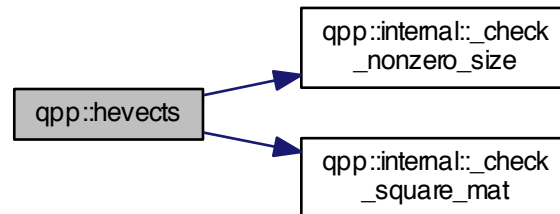
5.1.1.23 `template<typename Scalar > types::cmat qpp::hevals (const types::DynMat< Scalar > & A)`

Here is the call graph for this function:



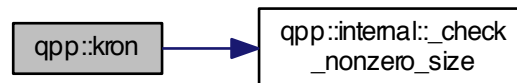
5.1.1.24 `template<typename Scalar > types::cmat qpp::hevects (const types::DynMat< Scalar > & A)`

Here is the call graph for this function:



5.1.1.25 `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:



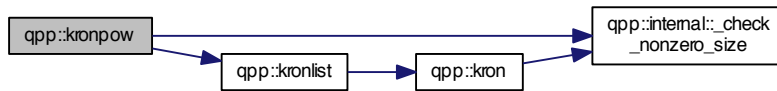
5.1.1.26 `template<typename Scalar > types::DynMat<Scalar> qpp::kronlist (const std::vector< types::DynMat< Scalar >> & list)`

Here is the call graph for this function:



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

Here is the call graph for this function:



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

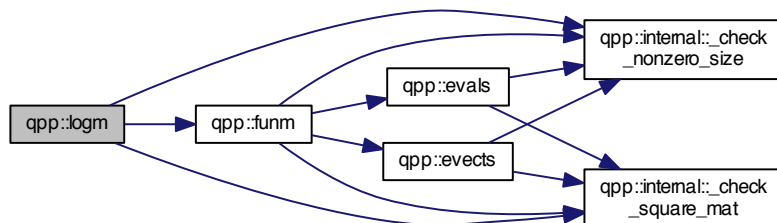
5.1.1.29 `template<typename Scalar > types::DynMat<Scalar> qpp::loadMATLABmatrix (const std::string & mat_file, const std::string & var_name)`

5.1.1.30 `template<> types::DynMat<double> qpp::loadMATLABmatrix (const std::string & mat_file, const std::string & var_name) [inline]`

5.1.1.31 `template<> types::DynMat<types::cplx> qpp::loadMATLABmatrix (const std::string & mat_file, const std::string & var_name) [inline]`

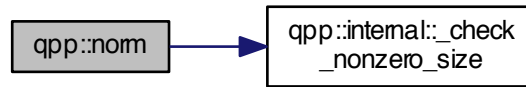
5.1.1.32 `template<typename Scalar > types::cmat qpp::logm (const types::DynMat< Scalar > & A)`

Here is the call graph for this function:



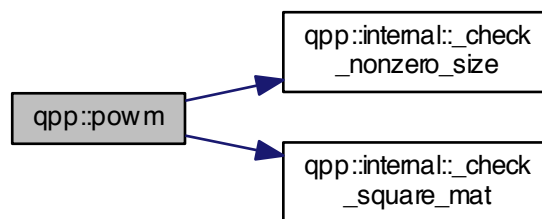
5.1.1.33 `template<typename Scalar > double qpp::norm (const types::DynMat< Scalar > & A)`

Here is the call graph for this function:



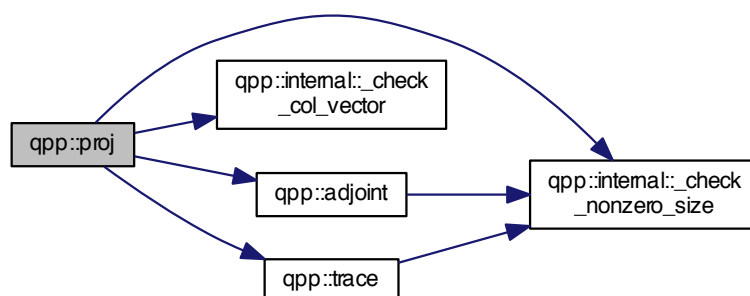
5.1.1.34 `template<typename Scalar > types::DynMat<Scalar> qpp::powm (const types::DynMat< Scalar > & A, size_t n)`

Here is the call graph for this function:



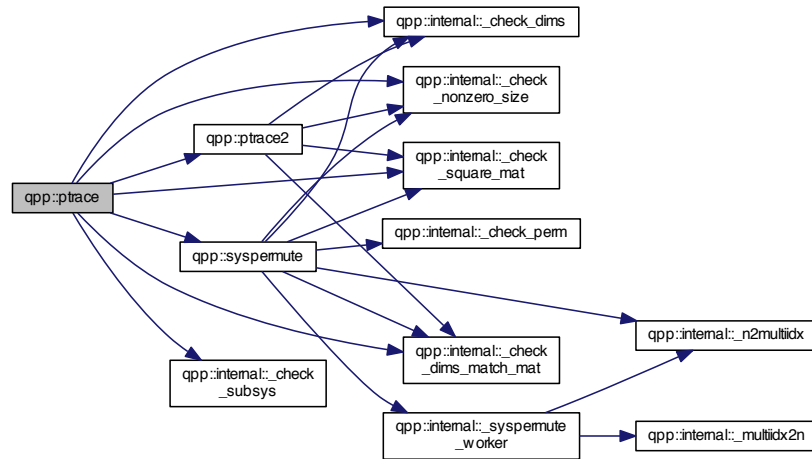
5.1.1.35 `template<typename Scalar > types::DynMat<Scalar> qpp::proj (const types::DynMat< Scalar > & V)`

Here is the call graph for this function:



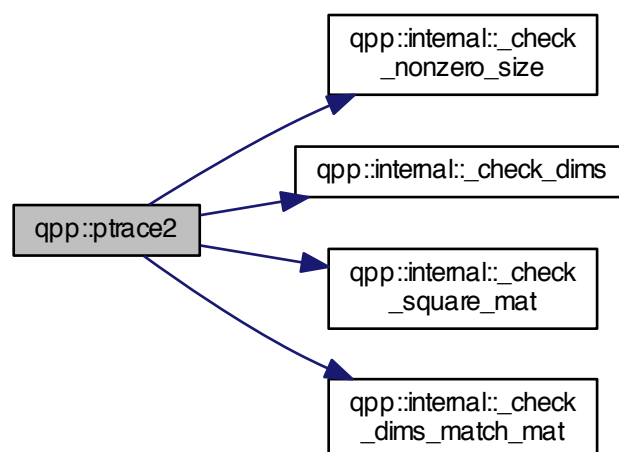
5.1.1.36 `template<typename Scalar> types::DynMat<Scalar> qpp::ptrace (const types::DynMat< Scalar> & A, const std::vector< size_t> & subsys, const std::vector< size_t> & dims)`

Here is the call graph for this function:



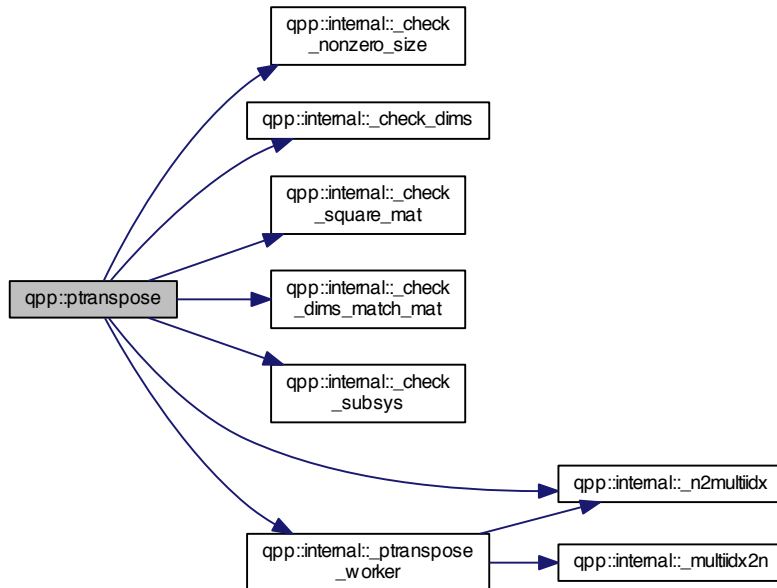
5.1.1.37 `template<typename Scalar> types::DynMat<Scalar> qpp::ptrace2 (const types::DynMat< Scalar> & A, const std::vector< size_t> dims)`

Here is the call graph for this function:



5.1.1.38 `template<typename Scalar > types::DynMat<Scalar> qpp::pttranspose (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:



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

5.1.1.40 `template<> types::DynMat<double> qpp::rand (size_t rows, size_t cols, double a, double b) [inline]`

5.1.1.41 `template<> types::DynMat<types::cplx> qpp::rand (size_t rows, size_t cols, double a, double b) [inline]`

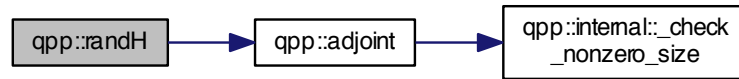
5.1.1.42 `double qpp::rand (double a = 0, double b = 1) [inline]`

Here is the call graph for this function:



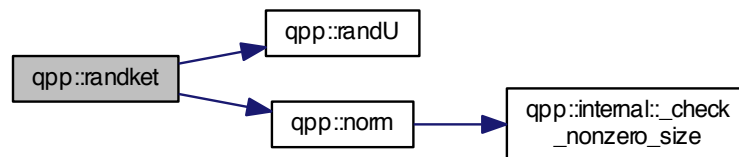
5.1.1.43 `types::cmat qpp::randH (size_t D) [inline]`

Here is the call graph for this function:



5.1.1.44 `types::cmat qpp::randket (size_t D) [inline]`

Here is the call graph for this function:



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

5.1.1.46 `template<> types::DynMat<double> qpp::randn (size_t rows, size_t cols, double mean, double sigma) [inline]`

Here is the call graph for this function:



5.1.1.47 `template<> types::DynMat<types::cplx> qpp::randn (size_t rows, size_t cols, double mean, double sigma)`
`[inline]`

Here is the call graph for this function:



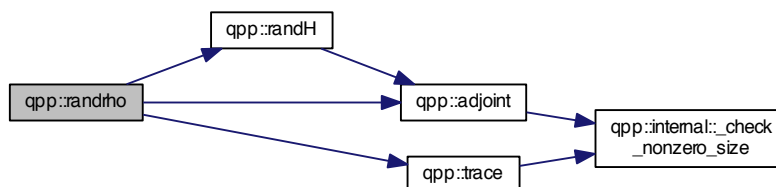
5.1.1.48 `double qpp::randn (double mean = 0, double sigma = 1)` `[inline]`

Here is the call graph for this function:



5.1.1.49 `types::cmat qpp::randrho (size_t D)` `[inline]`

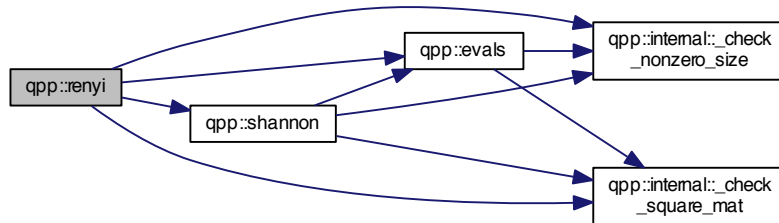
Here is the call graph for this function:



5.1.1.50 `types::cmat qpp::randU (size_t D)` `[inline]`

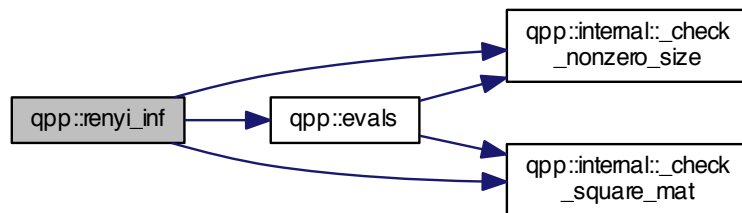
5.1.1.51 `template<typename Scalar > double qpp::renyi (const double alpha, const types::DynMat< Scalar > & A)`

Here is the call graph for this function:



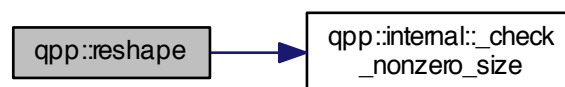
5.1.1.52 `template<typename Scalar > double qpp::renyi_inf (const types::DynMat< Scalar > & A)`

Here is the call graph for this function:



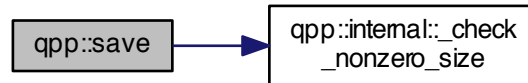
5.1.1.53 `template<typename Scalar > types::DynMat<Scalar> qpp::reshape (const types::DynMat< Scalar > & A, size_t rows, size_t cols)`

Here is the call graph for this function:



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

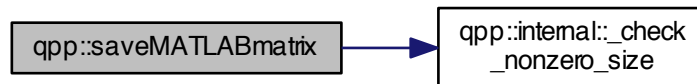
Here is the call graph for this function:



5.1.1.55 `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)`

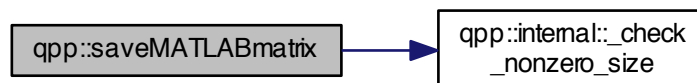
5.1.1.56 `template<> void qpp::saveMATLABmatrix (const types::DynMat< double > & A, const std::string & mat_file, const std::string & var_name, const std::string & mode)`

Here is the call graph for this function:



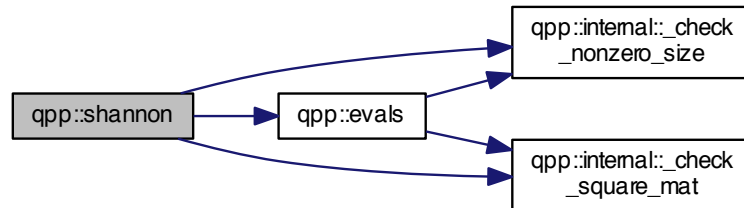
5.1.1.57 `template<> void qpp::saveMATLABmatrix (const types::DynMat< types::cplx > & A, const std::string & mat_file, const std::string & var_name, const std::string & mode)`

Here is the call graph for this function:



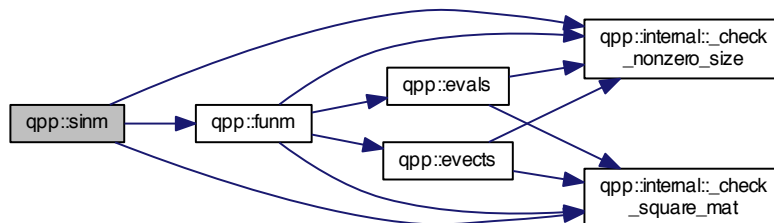
5.1.1.58 `template<typename Scalar > double qpp::shannon (const types::DynMat< Scalar > & A)`

Here is the call graph for this function:



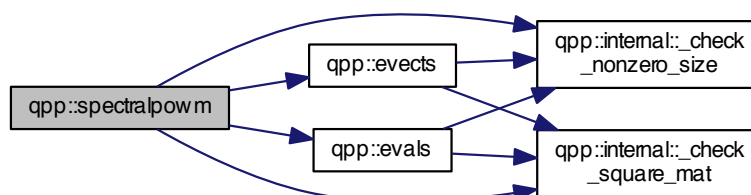
5.1.1.59 `template<typename Scalar > types::cmat qpp::sinm (const types::DynMat< Scalar > & A)`

Here is the call graph for this function:



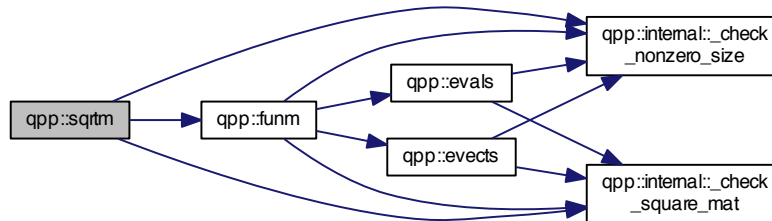
5.1.1.60 `template<typename Scalar > types::cmat qpp::spectralpowm (const types::DynMat< Scalar > & A, const types::cplx z)`

Here is the call graph for this function:



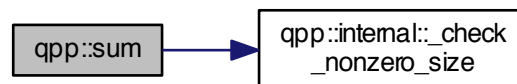
5.1.1.61 `template<typename Scalar > types::cmat qpp::sqrtm (const types::DynMat< Scalar > & A)`

Here is the call graph for this function:



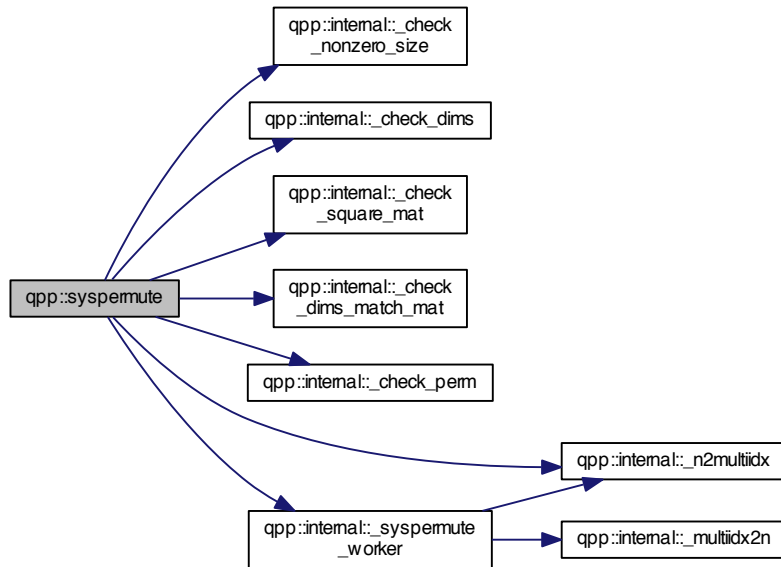
5.1.1.62 `template<typename Scalar > Scalar qpp::sum (const types::DynMat< Scalar > & A)`

Here is the call graph for this function:



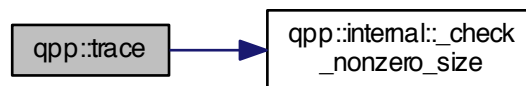
5.1.1.63 `template<typename Scalar > types::DynMat<Scalar> qpp::syspermute (const types::DynMat< Scalar > & A, const std::vector< size_t > perm, const std::vector< size_t > & dims)`

Here is the call graph for this function:



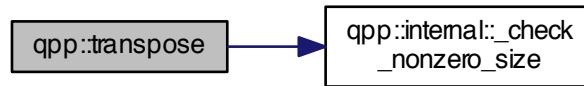
5.1.1.64 `template<typename Scalar > Scalar qpp::trace (const types::DynMat< Scalar > & A)`

Here is the call graph for this function:



5.1.1.65 `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 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) [inline]`

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

5.2.2.4 `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)
- [types::cmat TOF](#) (8, 8)

Variables

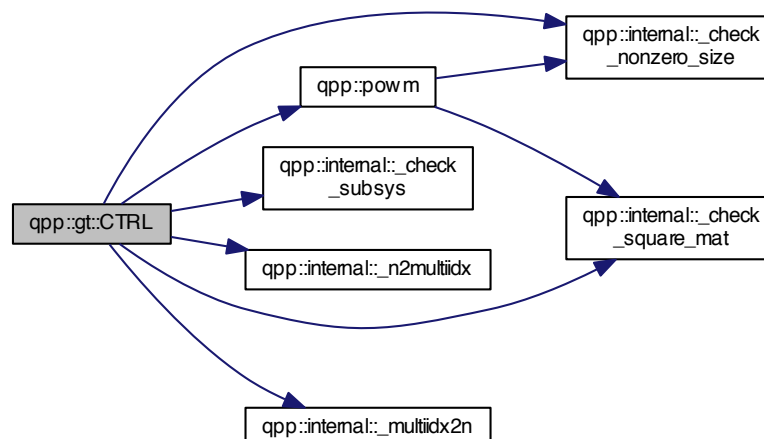
- [types::cmat H](#)
- [types::cmat Id2](#)
- [types::cmat X](#)
- [types::cmat Y](#)
- [types::cmat Z](#)
- [types::cmat S](#)
- [types::cmat T](#)
- [types::cmat CNOT](#)
- [types::cmat CP](#)
- [types::cmat TOF](#)

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::TOF (8, 8)`

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

Here is the call graph for this function:



5.3.1.8 `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::CNOT`

5.3.2.2 `types::cmat qpp::gt::CP`

5.3.2.3 `types::cmat qpp::gt::H`

5.3.2.4 `types::cmat qpp::gt::Id2`

5.3.2.5 `types::cmat qpp::gt::S`

5.3.2.6 `types::cmat qpp::gt::T`

5.3.2.7 `types::cmat qpp::gt::TOF`

5.3.2.8 `types::cmat qpp::gt::X`

5.3.2.9 `types::cmat qpp::gt::Y`

5.3.2.10 `types::cmat qpp::gt::Z`

5.4 qpp::internal Namespace Reference

Functions

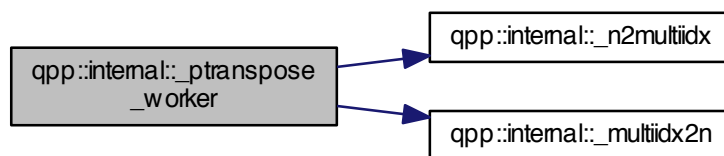
- `void _n2multiidx (size_t n, size_t numdims, const size_t *dims, size_t *result)`
- `size_t _multiidx2n (const size_t *midx, size_t numdims, const size_t *dims)`
- `template<typename Scalar >`
`bool _check_square_mat (const types::DynMat< Scalar > &A)`
- `template<typename Scalar >`
`bool _check_vector (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)`

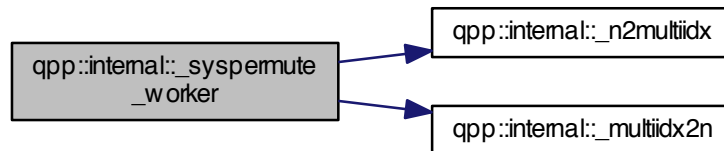
- 5.4.1.2 `bool qpp::internal::_check_dims (const std::vector< size_t > & dims)` `[inline]`
- 5.4.1.3 `template<typename Scalar > bool qpp::internal::_check_dims_match_mat (const std::vector< size_t > & dims, const types::DynMat< Scalar > & A)`
- 5.4.1.4 `bool qpp::internal::_check_eq_dims (const std::vector< size_t > & dims, size_t dim)` `[inline]`
- 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)` `[inline]`
- 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)` `[inline]`
- 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)` `[inline]`
- 5.4.1.12 `void qpp::internal::_n2multiidx (size_t n, size_t numdims, const size_t * dims, size_t * result)` `[inline]`
- 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)` `[inline]`

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) [inline]`

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 >`

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`

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)

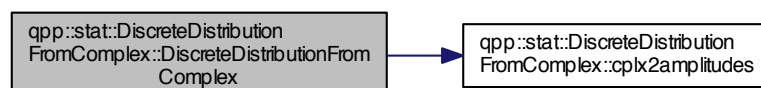
Protected Attributes

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

6.2.1 Constructor & Destructor Documentation

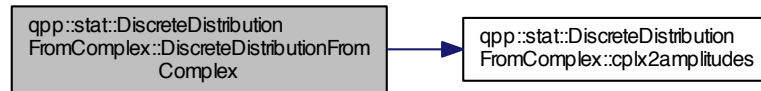
- 6.2.1.1 `template<typename InputIterator > qpp::stat::DiscreteDistributionFromComplex::DiscreteDistributionFromComplex (`
`InputIterator first, InputIterator last)` `[inline]`

Here is the call graph for this function:



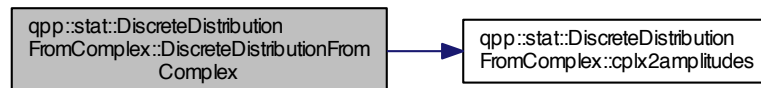
6.2.1.2 `qpp::stat::DiscreteDistributionFromComplex::DiscreteDistributionFromComplex (std::initializer_list< types::cplx > amplitudes) [inline]`

Here is the call graph for this function:



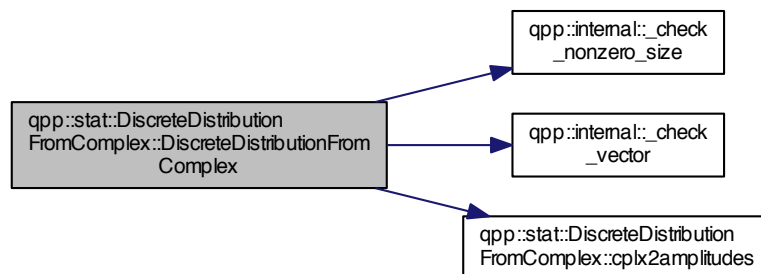
6.2.1.3 `qpp::stat::DiscreteDistributionFromComplex::DiscreteDistributionFromComplex (std::vector< types::cplx > amplitudes) [inline]`

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]`

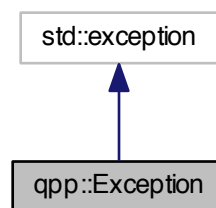
The documentation for this class was generated from the following file:

- include/[stat.h](#)

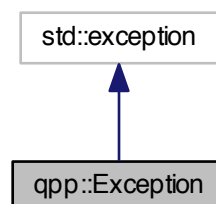
6.3 qpp::Exception Class Reference

```
#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]

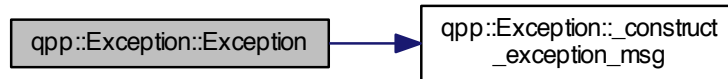
Enumerator

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

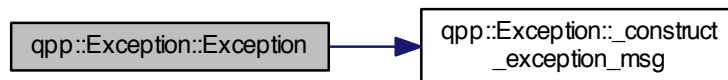
6.3.2.1 `qpp::Exception::Exception (const std::string & where, const Type & type)` `[inline]`

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](#)

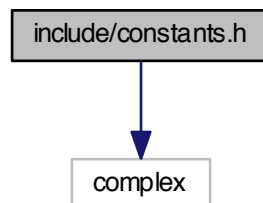
Chapter 7

File Documentation

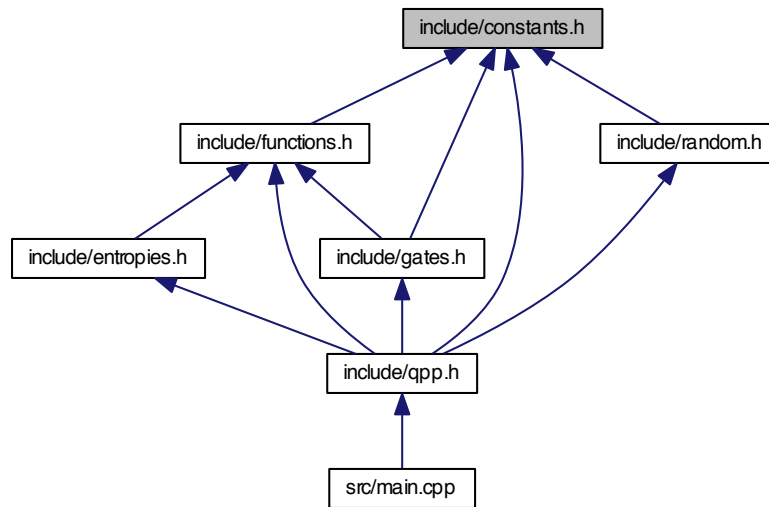
7.1 include/constants.h File Reference

```
#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 std::complex< double > qpp::ct::ii = { 0, 1 }`
- `const double qpp::ct::pi = 3.141592653589793238462643383279502884`
- `const double qpp::ct::ee = 2.718281828459045235360287471352662497`

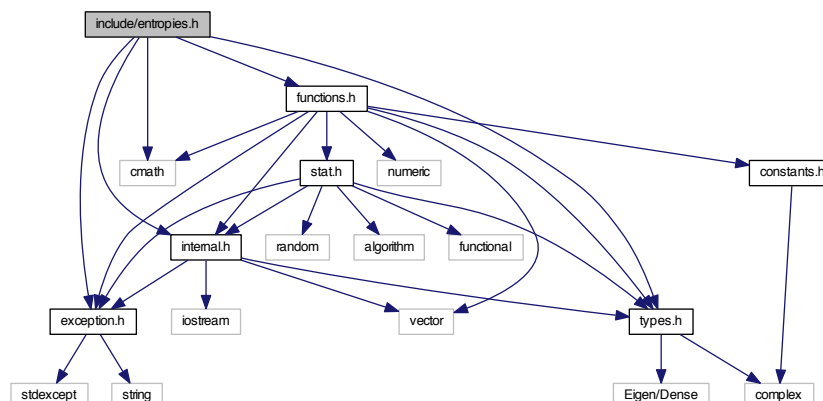
7.2 include/entropies.h File Reference

```

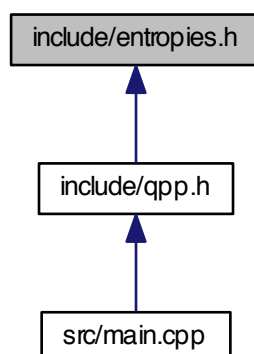
#include <cmath>
#include "types.h"
#include "functions.h"
#include "internal.h"
#include "exception.h"

```

Include dependency graph for entropies.h:



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



Namespaces

- [qpp](#)

Functions

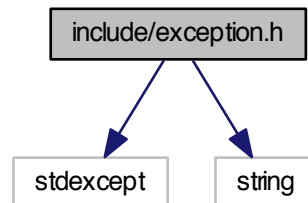
- `template<typename Scalar >`
`double qpp::shannon (const types::DynMat< Scalar > &A)`
- `template<typename Scalar >`
`double qpp::renyi (const double alpha, const types::DynMat< Scalar > &A)`
- `template<typename Scalar >`
`double qpp::renyi_inf (const types::DynMat< Scalar > &A)`

7.3 include/exception.h File Reference

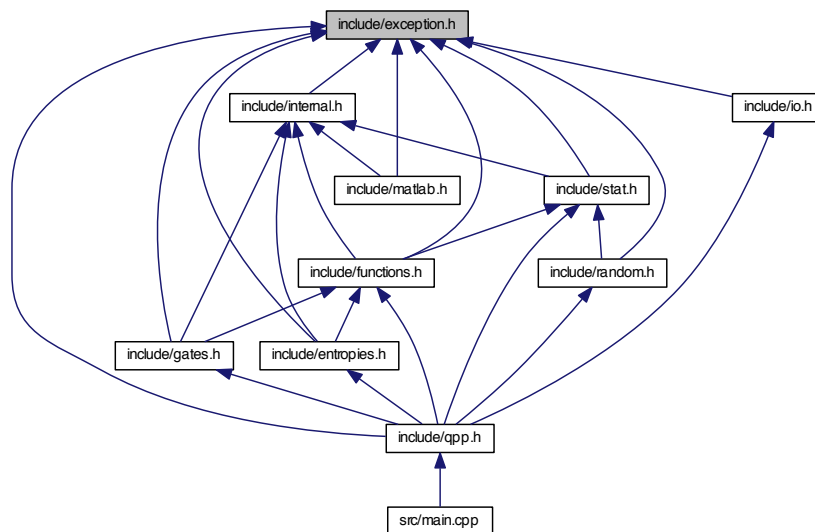
```
#include <stdexcept>
```

```
#include <string>
```

Include dependency graph for exception.h:



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



Classes

- class [qpp::Exception](#)

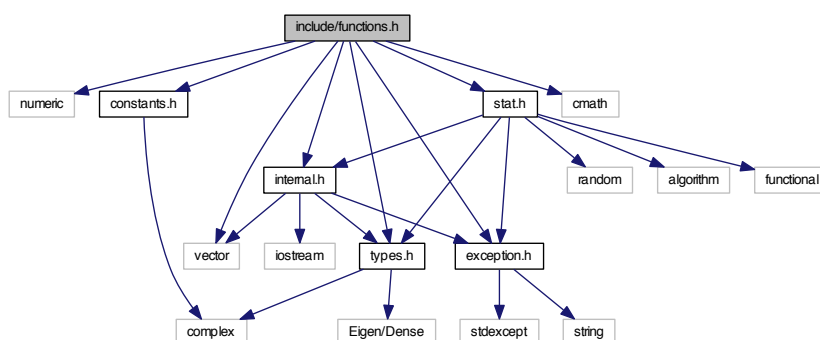
Namespaces

- [qpp](#)

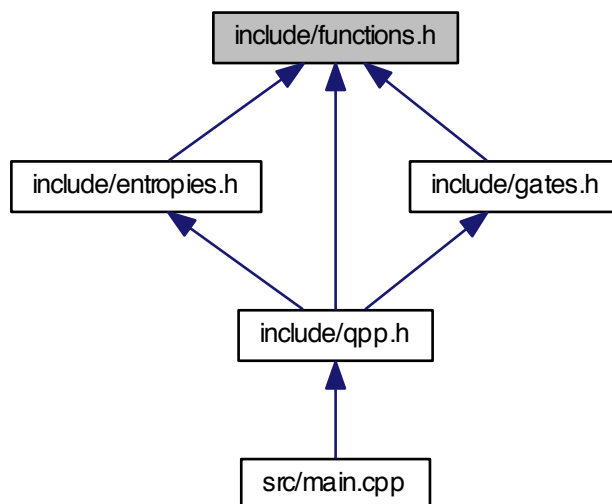
7.4 include/functions.h File Reference

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

Include dependency graph for functions.h:



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



Namespaces

- [qpp](#)

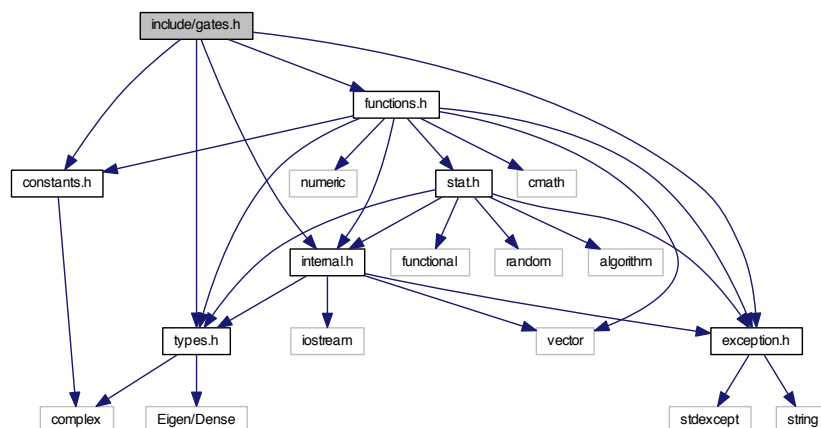
Functions

- `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::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)`

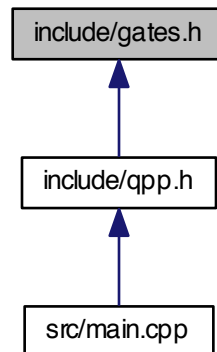
- `template<typename Scalar >`
`types::DynMat< Scalar > qpp::ptrace2 (const types::DynMat< Scalar > &A, const std::vector< size_t > dims)`
- `template<typename Scalar >`
`types::DynMat< Scalar > qpp::ptrace (const types::DynMat< Scalar > &A, const std::vector< size_t > &subsys, const std::vector< size_t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

7.5 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

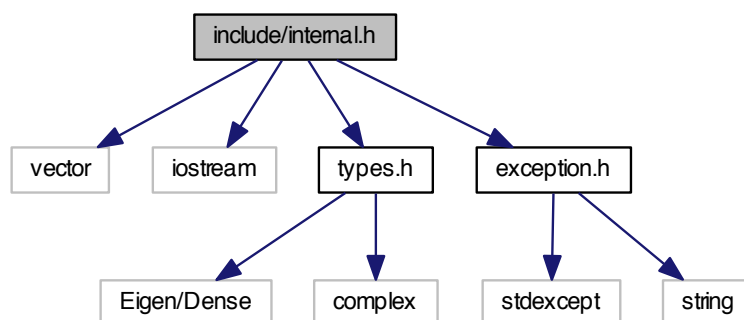
- void [qpp::gt::_init_gates](#) ()
- types::cmat [qpp::gt::Rtheta](#) (double theta)
- types::cmat [qpp::gt::Id](#) (size_t D)
- types::cmat [qpp::gt::Zd](#) (size_t D)
- types::cmat [qpp::gt::Fd](#) (size_t D)
- types::cmat [qpp::gt::Xd](#) (size_t D)
- types::cmat [qpp::gt::CTRL](#) (const types::cmat &A, const std::vector< size_t > &ctrl, const std::vector< size_t > &gate, size_t n, size_t D=2)

Variables

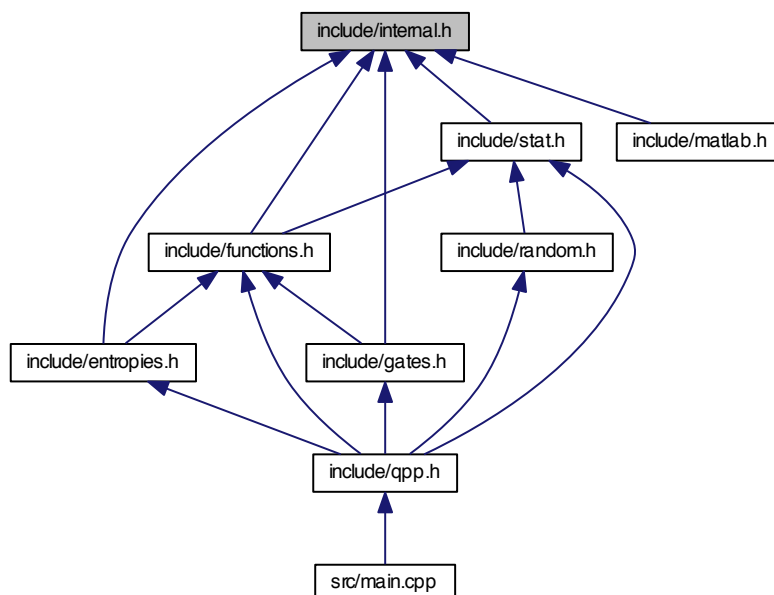
- types::cmat [qpp::gt::H](#)
- types::cmat [qpp::gt::Id2](#)
- types::cmat [qpp::gt::X](#)
- types::cmat [qpp::gt::Y](#)
- types::cmat [qpp::gt::Z](#)
- types::cmat [qpp::gt::S](#)
- types::cmat [qpp::gt::T](#)
- types::cmat [qpp::gt::CNOT](#)
- types::cmat [qpp::gt::CP](#)
- types::cmat [qpp::gt::TOF](#)

7.6 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

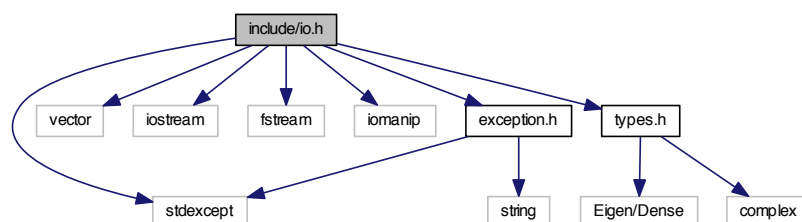
- `qpp`
- `qpp::internal`

Functions

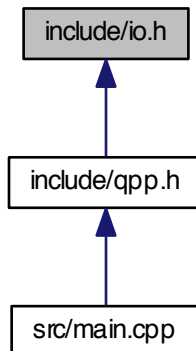
- void [qpp::internal::_n2multiidx](#) (size_t n, size_t numdims, const size_t *dims, size_t *result)
- size_t [qpp::internal::_multiidx2n](#) (const size_t *midx, size_t numdims, const size_t *dims)
- template<typename Scalar >
bool [qpp::internal::_check_square_mat](#) (const types::DynMat< Scalar > &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)

7.7 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

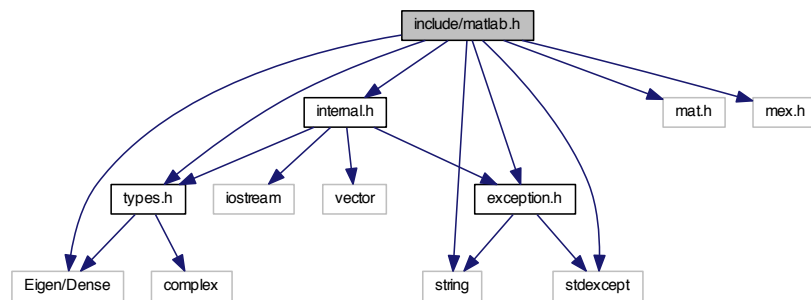
- `template<typename T >`
`void qpp::disp (const T &x, const std::string &separator=" ", const std::string &start="[", const std::string &end="]", std::ostream &os=std::cout)`
- `template<typename T >`
`void qpp::displn (const T &x, const std::string &separator=" ", const std::string &start="[", const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

7.8 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)`

7.9 include/qpp.h File Reference

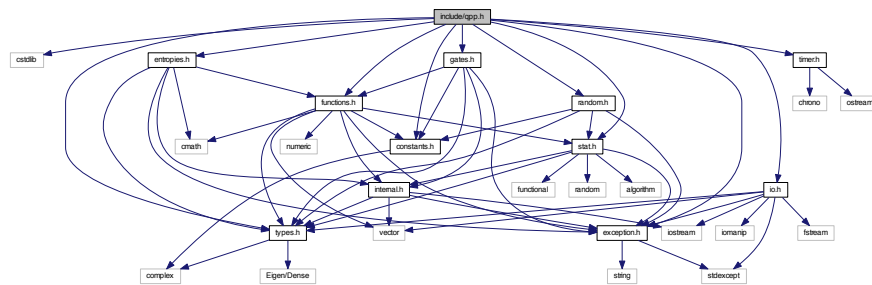
```
#include <cstdlib>
```

```

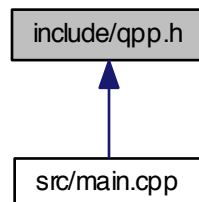
#include "types.h"
#include "constants.h"
#include "gates.h"
#include "stat.h"
#include "functions.h"
#include "random.h"
#include "entropies.h"
#include "io.h"
#include "timer.h"
#include "exception.h"

```

Include dependency graph for qpp.h:



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



Namespaces

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

Functions

- [types::cmat](#) [qpp::gt::TOF](#) (8, 8)
- [int](#) [qpp::_init](#) ()

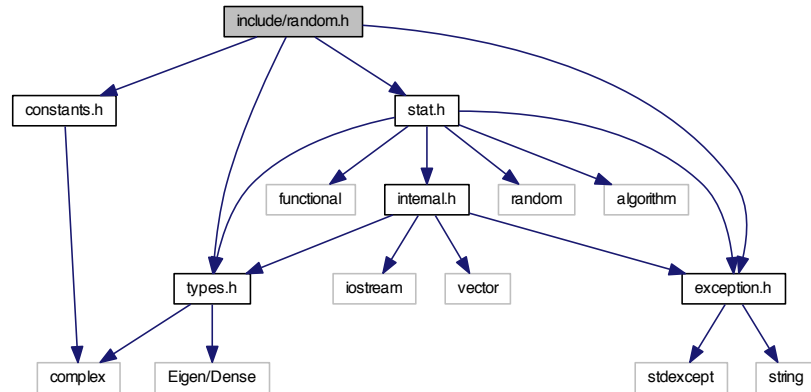
7.10 include/random.h File Reference

```

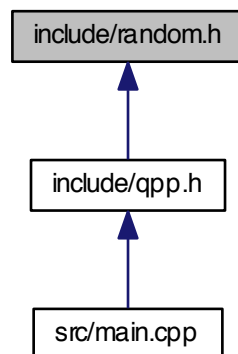
#include "types.h"

```

```
#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](#)

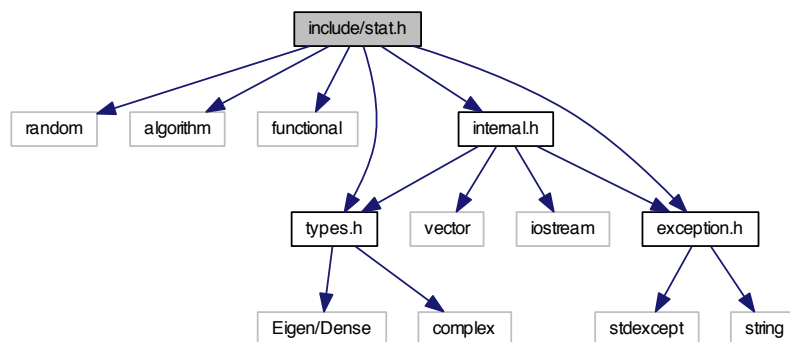
Functions

- `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)`

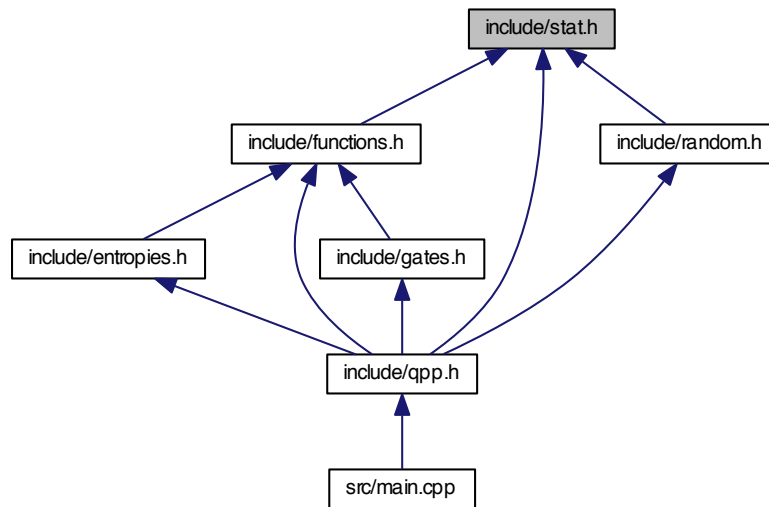
- double `qpp::rand` (double a=0, double b=1)
- `template<typename Scalar >`
`types::DynMat< Scalar > qpp::randn` (size_t rows, size_t cols, double mean=0, double sigma=1)
- `template<>`
`types::DynMat< double > qpp::randn` (size_t rows, size_t cols, double mean, double sigma)
- `template<>`
`types::DynMat< 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::randH` (size_t D)
- `types::cmat qpp::randket` (size_t D)
- `types::cmat qpp::randrho` (size_t D)

7.11 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](#)

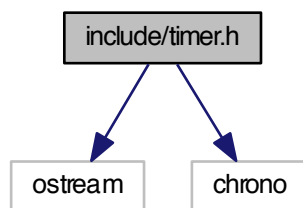
Variables

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

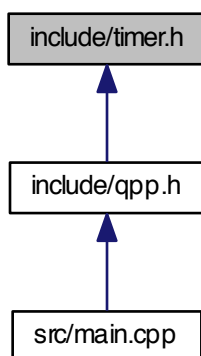
7.12 include/timer.h File Reference

```
#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`

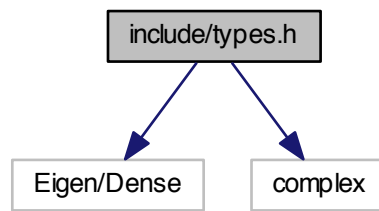
Namespaces

- `qpp`

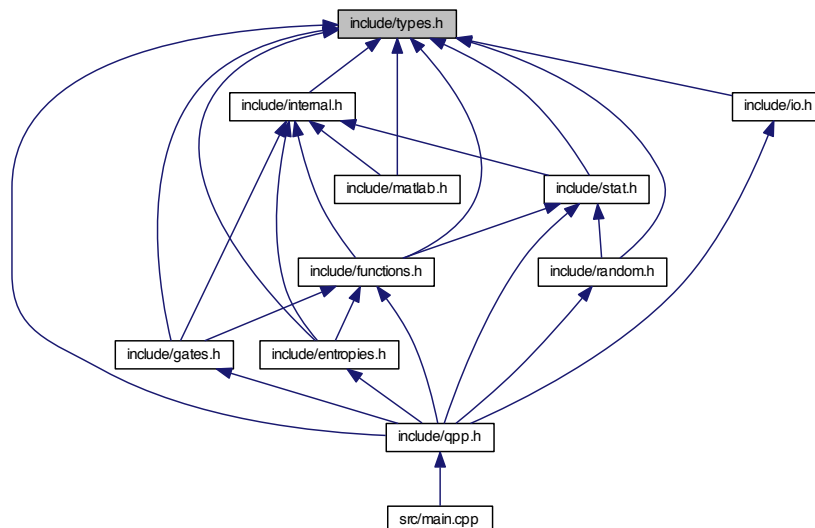
7.13 include/types.h File Reference

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

Include dependency graph for types.h:



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



Namespaces

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

Typedefs

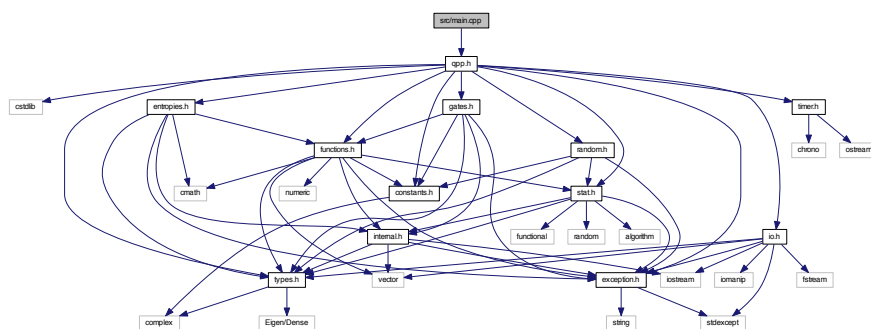
- `typedef std::complex< double > qpp::types::cplx`
- `typedef Eigen::MatrixXcd qpp::types::cmat`
- `typedef Eigen::MatrixXd qpp::types::dmat`
- `typedef Eigen::MatrixXf qpp::types::fmat`
- `typedef Eigen::MatrixXi qpp::types::imat`
- `template<typename Expression >`
`using qpp::types::Expression2DynMat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

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

7.14 src/main.cpp File Reference

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

Include dependency graph for main.cpp:



Functions

- `int main ()`

7.14.1 Function Documentation

7.14.1.1 int main ()

Here is the call graph for this function:

