

qpp  
0.1

Generated by Doxygen 1.8.5

Thu Apr 3 2014 23:31:15



# 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 . . . . .	11
5.1.1.1	_init . . . . .	11
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	displn . . . . .	14
5.1.1.12	displn . . . . .	15
5.1.1.13	displn . . . . .	15
5.1.1.14	evals . . . . .	15
5.1.1.15	evects . . . . .	16
5.1.1.16	expm . . . . .	16
5.1.1.17	fun . . . . .	16
5.1.1.18	funm . . . . .	17

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

5.1.1.59	<a href="#">trace</a>	30
5.1.1.60	<a href="#">transpose</a>	30
5.2	<a href="#">qpp::ct Namespace Reference</a>	30
5.2.1	<a href="#">Function Documentation</a>	30
5.2.1.1	<a href="#">omega</a>	30
5.2.2	<a href="#">Variable Documentation</a>	30
5.2.2.1	<a href="#">chop</a>	30
5.2.2.2	<a href="#">ee</a>	31
5.2.2.3	<a href="#">ii</a>	31
5.2.2.4	<a href="#">pi</a>	31
5.3	<a href="#">qpp::gt Namespace Reference</a>	31
5.3.1	<a href="#">Function Documentation</a>	31
5.3.1.1	<a href="#">_init_gates</a>	31
5.3.1.2	<a href="#">CU</a>	31
5.3.1.3	<a href="#">CUd</a>	31
5.3.1.4	<a href="#">Fd</a>	32
5.3.1.5	<a href="#">Rtheta</a>	32
5.3.1.6	<a href="#">TOF</a>	32
5.3.1.7	<a href="#">Xd</a>	32
5.3.1.8	<a href="#">Zd</a>	32
5.3.2	<a href="#">Variable Documentation</a>	32
5.3.2.1	<a href="#">CNOT</a>	32
5.3.2.2	<a href="#">CP</a>	33
5.3.2.3	<a href="#">H</a>	33
5.3.2.4	<a href="#">Id2</a>	33
5.3.2.5	<a href="#">S</a>	33
5.3.2.6	<a href="#">T</a>	33
5.3.2.7	<a href="#">TOF</a>	33
5.3.2.8	<a href="#">X</a>	33
5.3.2.9	<a href="#">Y</a>	33
5.3.2.10	<a href="#">Z</a>	33
5.4	<a href="#">qpp::internal Namespace Reference</a>	33
5.4.1	<a href="#">Function Documentation</a>	33
5.4.1.1	<a href="#">_check_dims</a>	33
5.4.1.2	<a href="#">_check_dims_match_mat</a>	33
5.4.1.3	<a href="#">_check_eq_dims</a>	33
5.4.1.4	<a href="#">_check_nonzero_size</a>	34
5.4.1.5	<a href="#">_check_perm</a>	34
5.4.1.6	<a href="#">_check_square_mat</a>	34
5.4.1.7	<a href="#">_check_subsys</a>	34

5.4.1.8	<a href="#">_check_vector</a>	34
5.4.1.9	<a href="#">_multiidx2n</a>	34
5.4.1.10	<a href="#">_n2multiidx</a>	34
5.4.1.11	<a href="#">_pttranspose_worker</a>	34
5.4.1.12	<a href="#">_syspermute_worker</a>	34
5.5	<a href="#">qpp::stat Namespace Reference</a>	35
5.5.1	<a href="#">Variable Documentation</a>	35
5.5.1.1	<a href="#">_rd</a>	35
5.5.1.2	<a href="#">_rng</a>	35
5.6	<a href="#">qpp::types Namespace Reference</a>	35
5.6.1	<a href="#">Typedef Documentation</a>	35
5.6.1.1	<a href="#">cmat</a>	35
5.6.1.2	<a href="#">cplx</a>	35
5.6.1.3	<a href="#">dmat</a>	35
5.6.1.4	<a href="#">DynMat</a>	35
5.6.1.5	<a href="#">Expression2DynMat</a>	35
5.6.1.6	<a href="#">fmat</a>	35
5.6.1.7	<a href="#">imat</a>	35
<b>6</b>	<b><a href="#">Class Documentation</a></b>	<b>37</b>
6.1	<a href="#">qpp::stat::DiscreteDistribution Class Reference</a>	37
6.1.1	<a href="#">Constructor &amp; Destructor Documentation</a>	37
6.1.1.1	<a href="#">DiscreteDistribution</a>	37
6.1.1.2	<a href="#">DiscreteDistribution</a>	37
6.1.1.3	<a href="#">DiscreteDistribution</a>	37
6.1.2	<a href="#">Member Function Documentation</a>	37
6.1.2.1	<a href="#">probabilities</a>	37
6.1.2.2	<a href="#">sample</a>	37
6.1.3	<a href="#">Member Data Documentation</a>	37
6.1.3.1	<a href="#">_d</a>	37
6.2	<a href="#">qpp::stat::DiscreteDistributionFromComplex Class Reference</a>	38
6.2.1	<a href="#">Constructor &amp; Destructor Documentation</a>	38
6.2.1.1	<a href="#">DiscreteDistributionFromComplex</a>	38
6.2.1.2	<a href="#">DiscreteDistributionFromComplex</a>	39
6.2.1.3	<a href="#">DiscreteDistributionFromComplex</a>	39
6.2.1.4	<a href="#">DiscreteDistributionFromComplex</a>	39
6.2.2	<a href="#">Member Function Documentation</a>	39
6.2.2.1	<a href="#">cplx2amplitudes</a>	40
6.2.2.2	<a href="#">probabilities</a>	40
6.2.2.3	<a href="#">sample</a>	40

6.2.3	Member Data Documentation . . . . .	40
6.2.3.1	_d . . . . .	40
6.3	qpp::Exception Class Reference . . . . .	40
6.3.1	Member Enumeration Documentation . . . . .	41
6.3.1.1	Type . . . . .	41
6.3.2	Constructor & Destructor Documentation . . . . .	42
6.3.2.1	Exception . . . . .	42
6.3.2.2	Exception . . . . .	42
6.3.2.3	~Exception . . . . .	42
6.3.3	Member Function Documentation . . . . .	42
6.3.3.1	_construct_exception_msg . . . . .	42
6.3.3.2	what . . . . .	42
6.3.4	Member Data Documentation . . . . .	42
6.3.4.1	_custom . . . . .	42
6.3.4.2	_msg . . . . .	42
6.3.4.3	_type . . . . .	42
6.3.4.4	_where . . . . .	42
6.4	qpp::stat::NormalDistribution Class Reference . . . . .	43
6.4.1	Constructor & Destructor Documentation . . . . .	43
6.4.1.1	NormalDistribution . . . . .	43
6.4.2	Member Function Documentation . . . . .	43
6.4.2.1	sample . . . . .	43
6.4.3	Member Data Documentation . . . . .	43
6.4.3.1	_d . . . . .	43
6.5	qpp::Timer Class Reference . . . . .	43
6.5.1	Constructor & Destructor Documentation . . . . .	44
6.5.1.1	Timer . . . . .	44
6.5.1.2	~Timer . . . . .	44
6.5.2	Member Function Documentation . . . . .	44
6.5.2.1	seconds . . . . .	44
6.5.2.2	tic . . . . .	44
6.5.2.3	toc . . . . .	44
6.5.3	Friends And Related Function Documentation . . . . .	44
6.5.3.1	operator<< . . . . .	44
6.5.4	Member Data Documentation . . . . .	44
6.5.4.1	_end . . . . .	44
6.5.4.2	_start . . . . .	44
6.6	qpp::stat::UniformRealDistribution Class Reference . . . . .	44
6.6.1	Constructor & Destructor Documentation . . . . .	44
6.6.1.1	UniformRealDistribution . . . . .	44

6.6.2	Member Function Documentation	44
6.6.2.1	sample	44
6.6.3	Member Data Documentation	44
6.6.3.1	_d	45
<b>7</b>	<b>File Documentation</b>	<b>47</b>
7.1	include/constants.h File Reference	47
7.2	include/entropies.h File Reference	48
7.3	include/exception.h File Reference	50
7.4	include/functions.h File Reference	51
7.5	include/gates.h File Reference	53
7.6	include/internal.h File Reference	55
7.7	include/io.h File Reference	56
7.8	include/matlab.h File Reference	57
7.9	include/qpp.h File Reference	58
7.10	include/random.h File Reference	59
7.11	include/stat.h File Reference	61
7.12	include/timer.h File Reference	62
7.13	include/types.h File Reference	63
7.14	src/main.cpp File Reference	65
7.14.1	Function Documentation	65
7.14.1.1	main	65



# 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>	30
<a href="#">qpp::gt</a>	31
<a href="#">qpp::internal</a>	33
<a href="#">qpp::stat</a>	35
<a href="#">qpp::types</a>	35



## Chapter 2

# Hierarchical Index

### 2.1 Class Hierarchy

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

qpp::stat::DiscreteDistribution . . . . .	37
qpp::stat::DiscreteDistributionFromComplex . . . . .	38
exception	
qpp::Exception . . . . .	40
qpp::stat::NormalDistribution . . . . .	43
qpp::Timer . . . . .	43
qpp::stat::UniformRealDistribution . . . . .	44



## 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>	37
<a href="#">qpp::stat::DiscreteDistributionFromComplex</a>	38
<a href="#">qpp::Exception</a>	40
<a href="#">qpp::stat::NormalDistribution</a>	43
<a href="#">qpp::Timer</a>	43
<a href="#">qpp::stat::UniformRealDistribution</a>	44



# Chapter 4

## File Index

### 4.1 File List

Here is a list of all files with brief descriptions:

include/	constants.h	47
include/	entropies.h	48
include/	exception.h	50
include/	functions.h	51
include/	gates.h	53
include/	internal.h	55
include/	io.h	56
include/	matlab.h	57
include/	qpp.h	58
include/	random.h	59
include/	stat.h	61
include/	timer.h	62
include/	types.h	63
src/	main.cpp	65





## 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 powm` (const `types::DynMat< Scalar >` &A, const `types::cplx` z)
- `template<typename Scalar >`  
`types::DynMat< Scalar >` `powm_int` (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 >` `kron_list` (const `std::vector< types::DynMat< Scalar >>` &list)
- `template<typename Scalar >`  
`types::DynMat< Scalar >` `kron_pow` (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 T >`  
void `disp` (const T &x, const `std::string` &separator=" ", `std::ostream` &os=`std::cout`)
- `template<typename T >`  
void `displin` (const T &x, const `std::string` &separator=" ", `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 `displin` (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

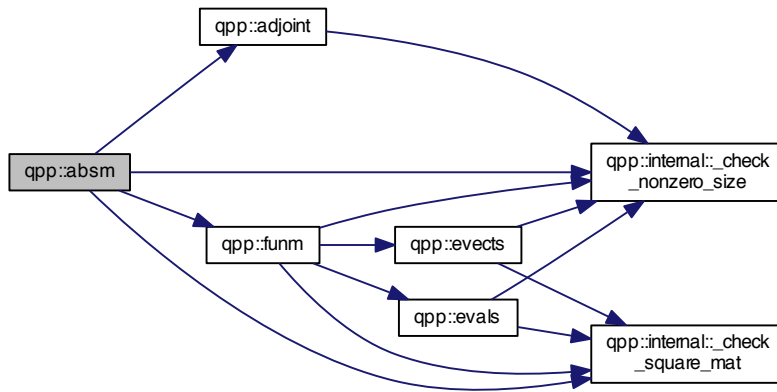
#### 5.1.1.1 int qpp::\_init ( )

Here is the call graph for this function:



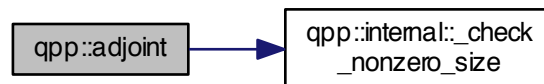
5.1.1.2 `template<typename Scalar > types::cmat qpp::absm ( const types::DynMat< Scalar > & A )`

Here is the call graph for this function:



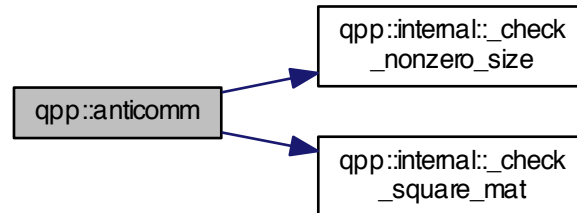
5.1.1.3 `template<typename Scalar > types::DynMat<Scalar> qpp::adjoint ( const types::DynMat< Scalar > & A )`

Here is the call graph for this function:



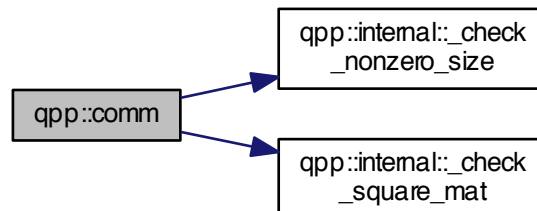
5.1.1.4 `template<typename Scalar > types::DynMat<Scalar> qpp::anticomm ( const types::DynMat< Scalar > & A, const types::DynMat< Scalar > & B )`

Here is the call graph for this function:



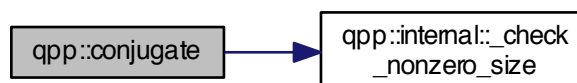
5.1.1.5 `template<typename Scalar > types::DynMat<Scalar> qpp::comm ( const types::DynMat< Scalar > & A, const types::DynMat< Scalar > & B )`

Here is the call graph for this function:



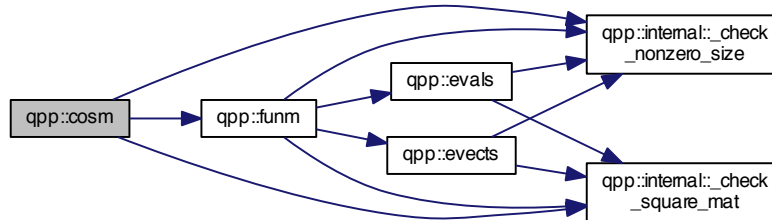
5.1.1.6 `template<typename Scalar > types::DynMat<Scalar> qpp::conjugate ( const types::DynMat< Scalar > & A )`

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 = " ", std::ostream & os = std::cout )`

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

5.1.1.10 `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:



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

Here is the call graph for this function:



5.1.1.12 `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:



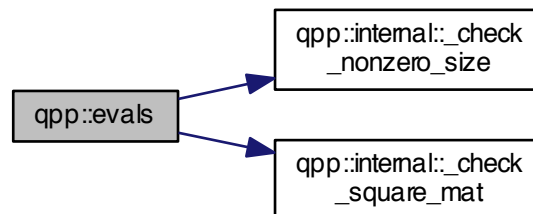
5.1.1.13 `void qpp::displn ( const types::cplx c, double chop = ct::chop, std::ostream & os = std::cout )`  
`[inline]`

Here is the call graph for this function:



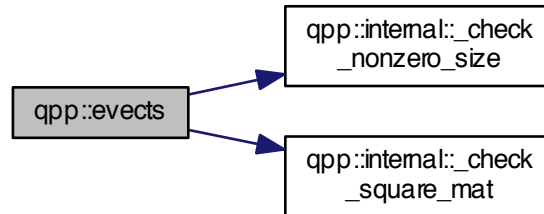
5.1.1.14 `template<typename Scalar > types::cmat qpp::evals ( const types::DynMat< Scalar > & A )`

Here is the call graph for this function:



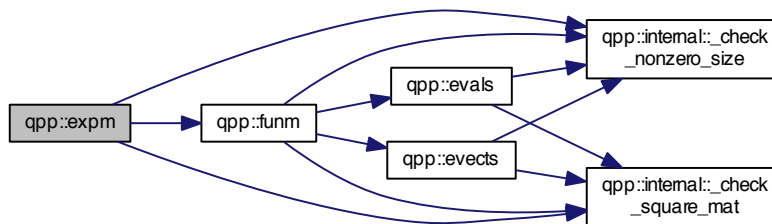
5.1.1.15 `template<typename Scalar > types::cmat qpp::evecs ( const types::DynMat< Scalar > & A )`

Here is the call graph for this function:



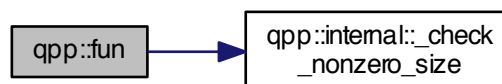
5.1.1.16 `template<typename Scalar > types::cmat qpp::expm ( const types::DynMat< Scalar > & A )`

Here is the call graph for this function:



5.1.1.17 `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:





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

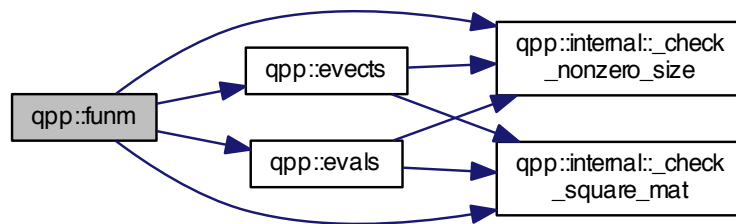
## Parameters

$A$	input matrix
$f$	function pointer

## Returns

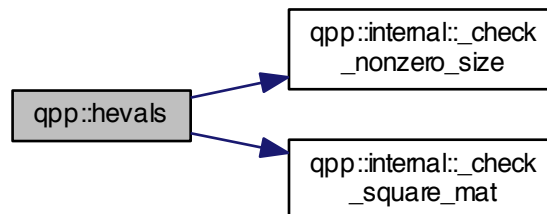
[types::cmat](#)

Here is the call graph for this function:



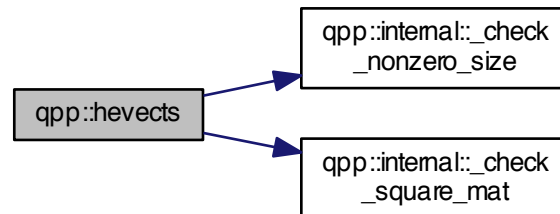
5.1.1.19 `template<typename Scalar > types::cmat qpp::hevals ( const types::DynMat< Scalar > & A )`

Here is the call graph for this function:



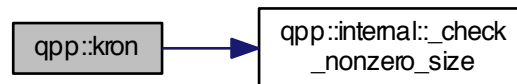
5.1.1.20 `template<typename Scalar > types::cmat qpp::hevects ( const types::DynMat< Scalar > & A )`

Here is the call graph for this function:



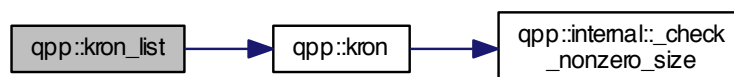
5.1.1.21 `template<typename Scalar > types::DynMat<Scalar> qpp::kron ( const types::DynMat< Scalar > & A, const types::DynMat< Scalar > & B )`

Here is the call graph for this function:



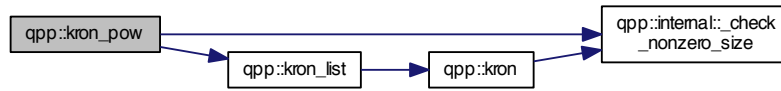
5.1.1.22 `template<typename Scalar > types::DynMat<Scalar> qpp::kron_list ( const std::vector< types::DynMat< Scalar >> & list )`

Here is the call graph for this function:



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

Here is the call graph for this function:



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

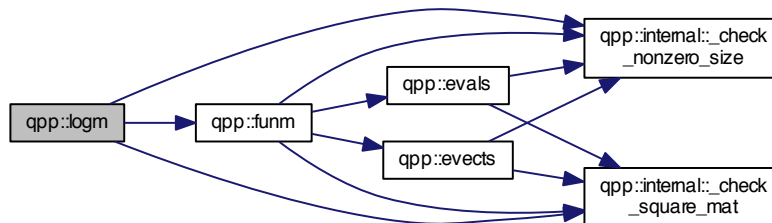
5.1.1.25 `template<typename Scalar > types::DynMat<Scalar> qpp::loadMATLABmatrix ( const std::string & mat_file, const std::string & var_name )`

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

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

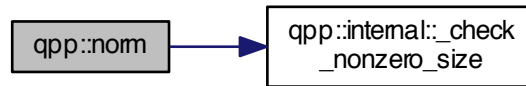
5.1.1.28 `template<typename Scalar > types::cmat qpp::logm ( const types::DynMat< Scalar > & A )`

Here is the call graph for this function:



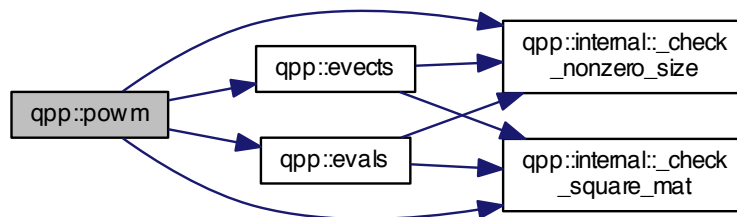
5.1.1.29 `template<typename Scalar > double qpp::norm ( const types::DynMat< Scalar > & A )`

Here is the call graph for this function:



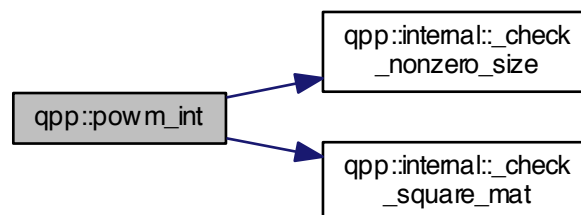
5.1.1.30 `template<typename Scalar > types::cmat qpp::powm ( const types::DynMat< Scalar > & A, const types::cplx z )`

Here is the call graph for this function:

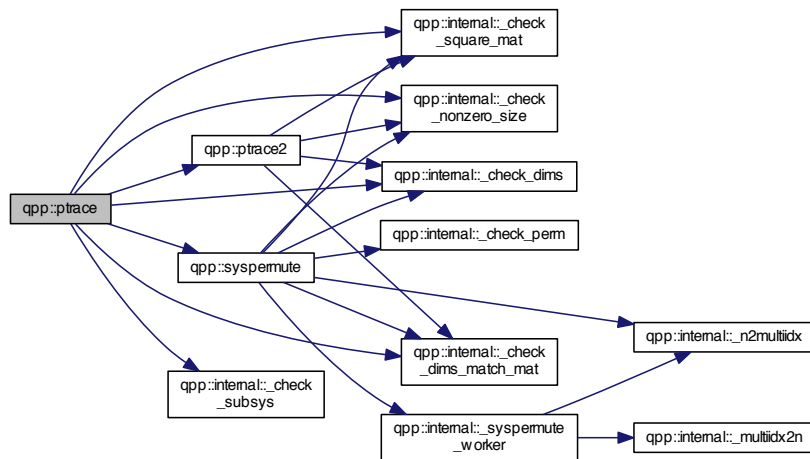


5.1.1.31 `template<typename Scalar > types::DynMat<Scalar> qpp::powm_int ( const types::DynMat< Scalar > & A, size_t n )`

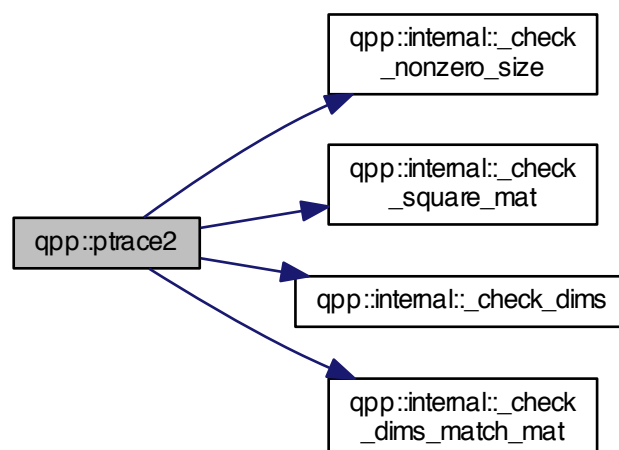
Here is the call graph for this function:



Here is the call graph for this function:

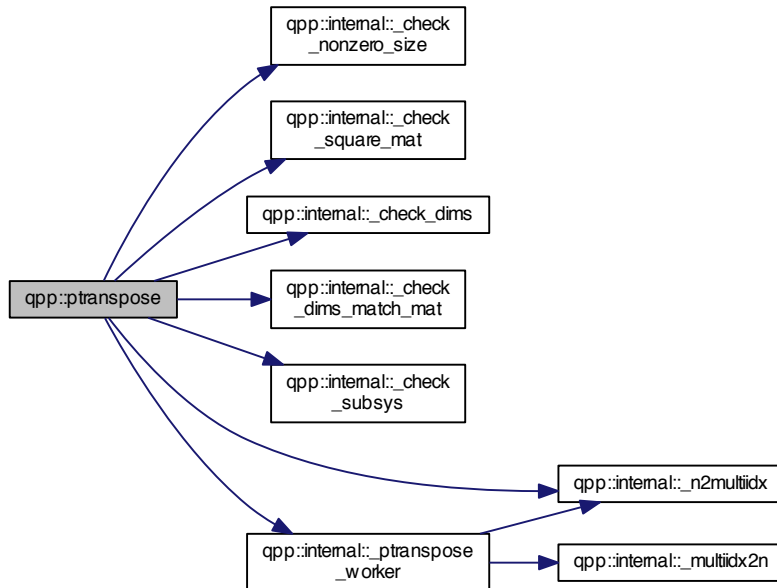


Here is the call graph for this function:



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

Here is the call graph for this function:



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

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

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

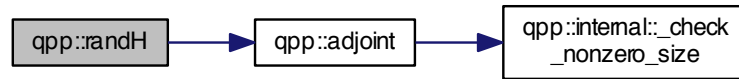
5.1.1.38 `double qpp::rand ( double a = 0, double b = 1 ) [inline]`

Here is the call graph for this function:



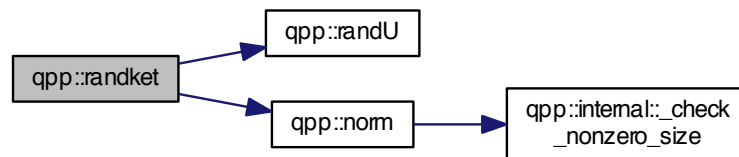
#### 5.1.1.39 `types::cmat qpp::randH ( size_t D ) [inline]`

Here is the call graph for this function:



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

Here is the call graph for this function:



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

#### 5.1.1.42 `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:





5.1.1.43 `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:



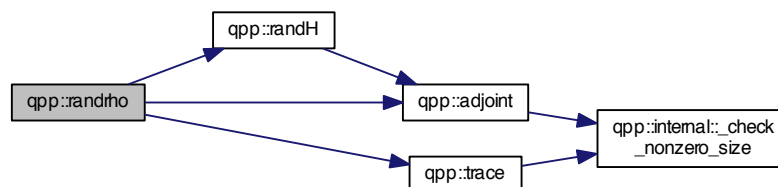
5.1.1.44 `double qpp::randn ( double mean = 0, double sigma = 1 )` `[inline]`

Here is the call graph for this function:



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

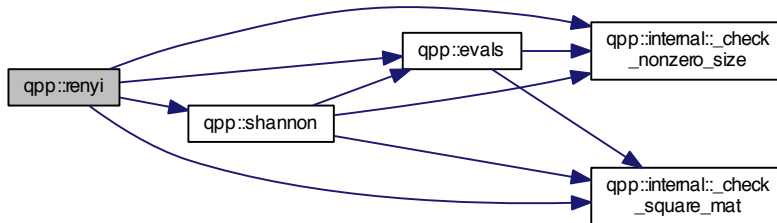
Here is the call graph for this function:



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

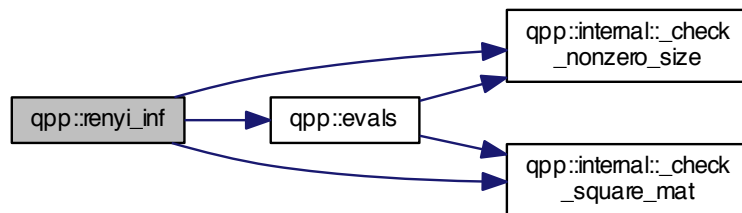
5.1.1.47 `template<typename Scalar > double qpp::renyi ( const double alpha, const types::DynMat< Scalar > & A )`

Here is the call graph for this function:



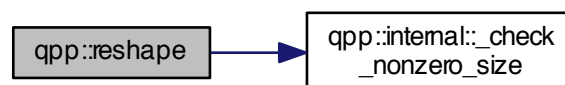
5.1.1.48 `template<typename Scalar > double qpp::renyi_inf ( const types::DynMat< Scalar > & A )`

Here is the call graph for this function:



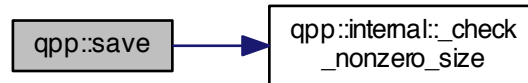
5.1.1.49 `template<typename Scalar > types::DynMat<Scalar> qpp::reshape ( const types::DynMat< Scalar > & A, size_t rows, size_t cols )`

Here is the call graph for this function:



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

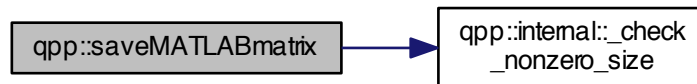
Here is the call graph for this function:



5.1.1.51 `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 )`

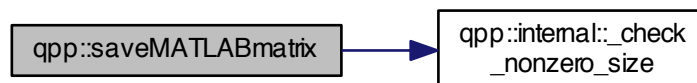
5.1.1.52 `template<> void qpp::saveMATLABmatrix ( const types::DynMat< double > & A, const std::string & mat_file, const std::string & var_name, const std::string & mode )`

Here is the call graph for this function:



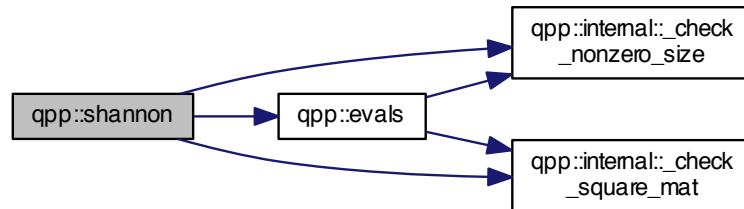
5.1.1.53 `template<> void qpp::saveMATLABmatrix ( const types::DynMat< types::cplx > & A, const std::string & mat_file, const std::string & var_name, const std::string & mode )`

Here is the call graph for this function:



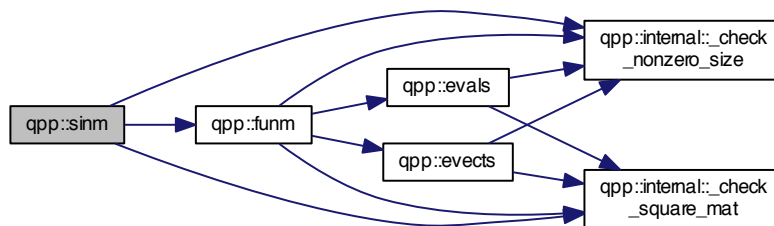
5.1.1.54 `template<typename Scalar > double qpp::shannon ( const types::DynMat< Scalar > & A )`

Here is the call graph for this function:



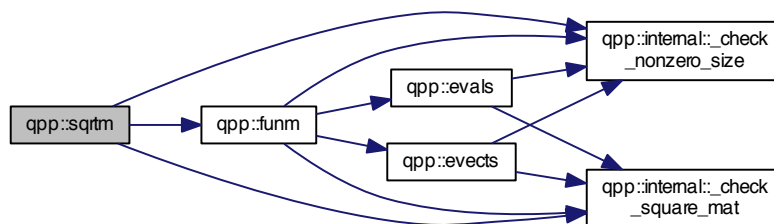
5.1.1.55 `template<typename Scalar > types::cmat qpp::sinm ( const types::DynMat< Scalar > & A )`

Here is the call graph for this function:



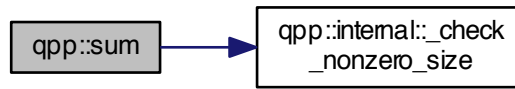
5.1.1.56 `template<typename Scalar > types::cmat qpp::sqrtm ( const types::DynMat< Scalar > & A )`

Here is the call graph for this function:



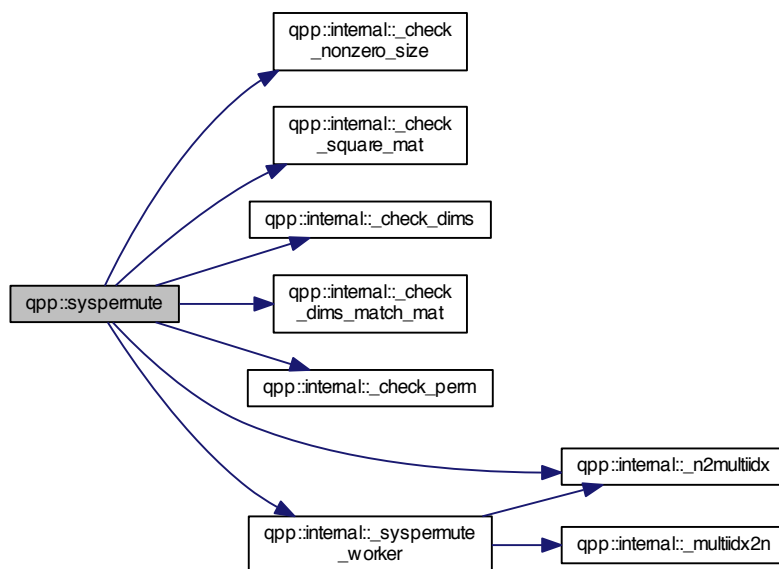
5.1.1.57 `template<typename Scalar > Scalar qpp::sum ( const types::DynMat< Scalar > & A )`

Here is the call graph for this function:



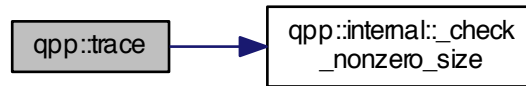
5.1.1.58 `template<typename Scalar > types::DynMat<Scalar> qpp::syspermute ( const types::DynMat< Scalar > & A, const std::vector< size_t > perm, const std::vector< size_t > & dims )`

Here is the call graph for this function:



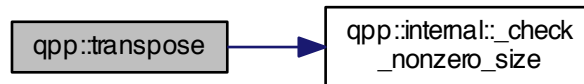
5.1.1.59 `template<typename Scalar > Scalar qpp::trace ( const types::DynMat< Scalar > & A )`

Here is the call graph for this function:



5.1.1.60 `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

- `types::cplx omega (size_t D)`

### Variables

- `const double chop = 1e-10`
- `const types::cplx ii = { 0, 1 }`
- `const double pi = 3.141592653589793238462643383279502884`
- `const double ee = 2.718281828459045235360287471352662497`

### 5.2.1 Function Documentation

5.2.1.1 `types::cplx 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 types::cplx 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 CU (const types::cmat &U)`
- `types::cmat Zd (size_t D)`
- `types::cmat Fd (size_t D)`
- `types::cmat Xd (size_t D)`
- `types::cmat CUd (const types::cmat &U)`
- `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`

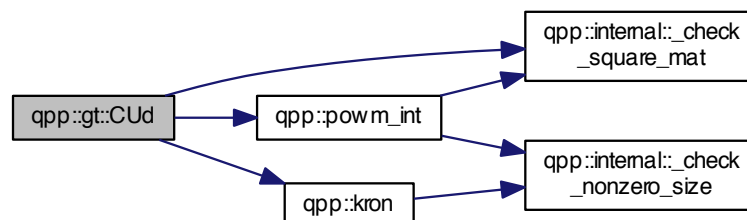
#### 5.3.1 Function Documentation

5.3.1.1 `void qpp::gt::_init_gates ( ) [inline]`

5.3.1.2 `types::cmat qpp::gt::CU ( const types::cmat & U ) [inline]`

5.3.1.3 `types::cmat qpp::gt::CUd ( const types::cmat & U ) [inline]`

Here is the call graph for this function:



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

Here is the call graph for this function:



#### 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::ld2`

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_nonzero_size (const types::DynMat< Scalar > &A)`
- `template<typename Scalar >`  
`bool _check_dims_match_mat (const std::vector< size_t > &dims, const types::DynMat< Scalar > &A)`
- `bool _check_dims (const std::vector< size_t > &dims)`
- `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

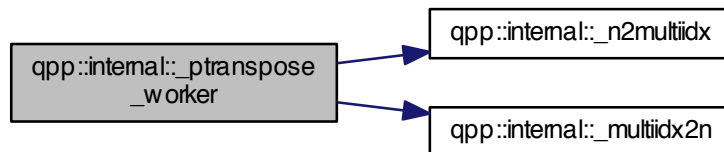
5.4.1.1 `bool qpp::internal::_check_dims ( const std::vector< size_t > & dims ) [inline]`

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

5.4.1.3 `bool qpp::internal::_check_eq_dims ( const std::vector< size_t > & dims, size_t dim ) [inline]`

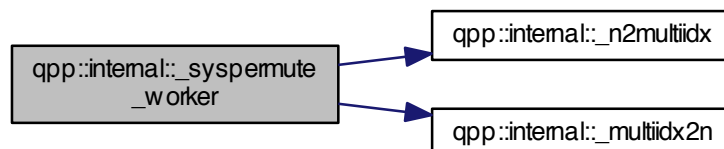
- 5.4.1.4 `template<typename Scalar > bool qpp::internal::_check_nonzero_size ( const types::DynMat< Scalar > & A )`
- 5.4.1.5 `bool qpp::internal::_check_perm ( const std::vector< size_t > & perm, const std::vector< size_t > & dims )`  
[inline]
- 5.4.1.6 `template<typename Scalar > bool qpp::internal::_check_square_mat ( const types::DynMat< Scalar > & A )`
- 5.4.1.7 `bool qpp::internal::_check_subsys ( const std::vector< size_t > & subsys, const std::vector< size_t > & dims )`  
[inline]
- 5.4.1.8 `template<typename Scalar > bool qpp::internal::_check_vector ( const types::DynMat< Scalar > & A )`
- 5.4.1.9 `size_t qpp::internal::_multiidx2n ( const size_t * midx, size_t numdims, const size_t * dims )` [inline]
- 5.4.1.10 `void qpp::internal::_n2multiidx ( size_t n, size_t numdims, const size_t * dims, size_t * result )` [inline]
- 5.4.1.11 `template<typename Scalar > void qpp::internal::_ptranpose_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:



- 5.4.1.12 `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

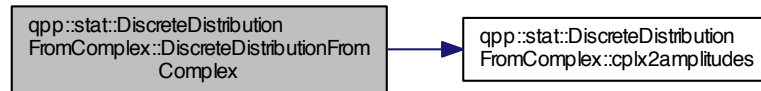
- 6.2.1.1 `template<typename InputIterator > qpp::stat::DiscreteDistributionFromComplex::DiscreteDistributionFromComplex (`  
`InputIterator first, InputIterator last )` `[inline]`

Here is the call graph for this function:



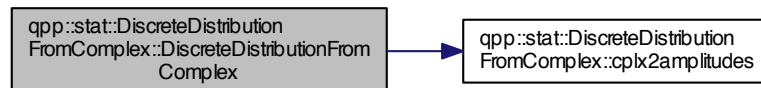
6.2.1.2 `qpp::stat::DiscreteDistributionFromComplex::DiscreteDistributionFromComplex ( std::initializer_list< types::cplx > amplitudes ) [inline]`

Here is the call graph for this function:



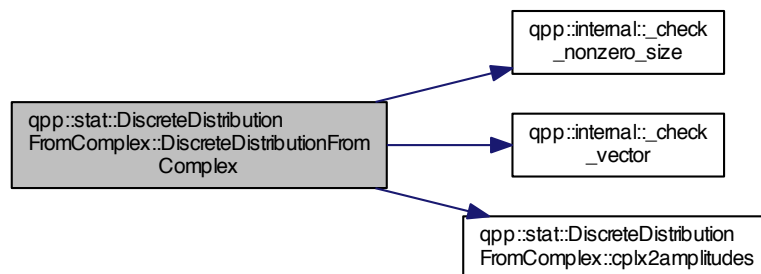
6.2.1.3 `qpp::stat::DiscreteDistributionFromComplex::DiscreteDistributionFromComplex ( std::vector< types::cplx > amplitudes ) [inline]`

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]`

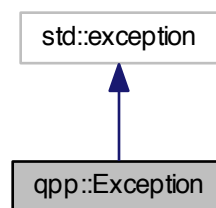
The documentation for this class was generated from the following file:

- [include/stat.h](#)

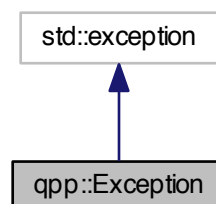
## 6.3 qpp::Exception Class Reference

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

Inheritance diagram for qpp::Exception:



Collaboration diagram for qpp::Exception:





## Public Types

- enum [Type](#) {  
[Type::UNKNOWN\\_EXCEPTION](#) = 0, [Type::MATRIX\\_NOT\\_SQUARE](#), [Type::MATRIX\\_NOT\\_VECTOR](#), [Type::MATRIX\\_ZERO\\_SIZE](#),  
[Type::DIMS\\_MISMATCH\\_MATRIX](#), [Type::DIMS\\_HAVE\\_ZERO](#), [Type::DIMS\\_NOT\\_EQUAL](#), [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]

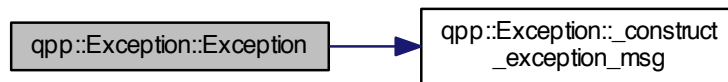
Enumerator

***UNKNOWN\_EXCEPTION***  
***MATRIX\_NOT\_SQUARE***  
***MATRIX\_NOT\_VECTOR***  
***MATRIX\_ZERO\_SIZE***  
***DIMS\_MISMATCH\_MATRIX***  
***DIMS\_HAVE\_ZERO***  
***DIMS\_NOT\_EQUAL***  
***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

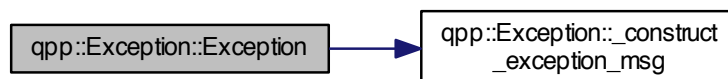
6.3.2.1 `qpp::Exception::Exception ( const std::string & where, const Type & type )` `[inline]`

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](#)



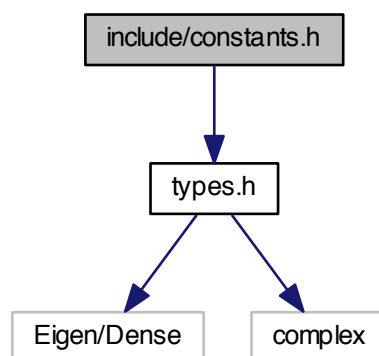
## Chapter 7

# File Documentation

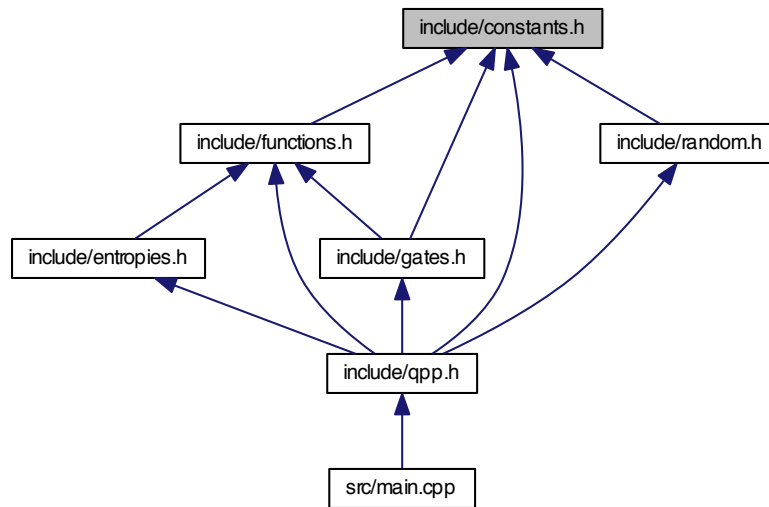
### 7.1 include/constants.h File Reference

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

Include dependency graph for constants.h:



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



## Namespaces

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

## Functions

- `types::cplx qpp::ct::omega (size_t D)`

## Variables

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

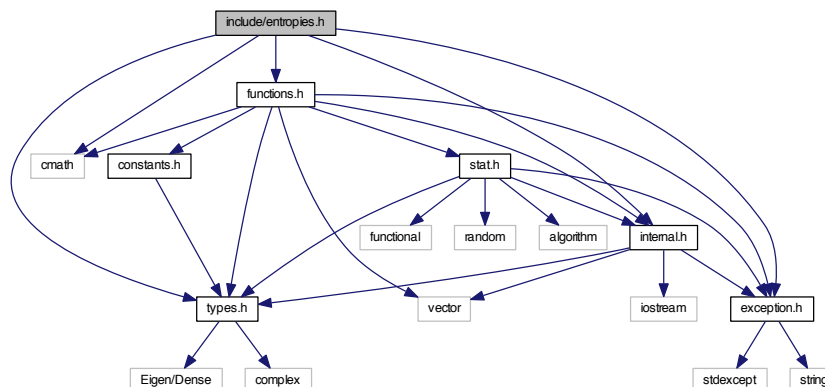
```

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

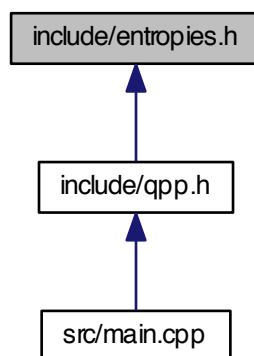
```



Include dependency graph for entropies.h:



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



## Namespaces

- [qpp](#)

## Functions

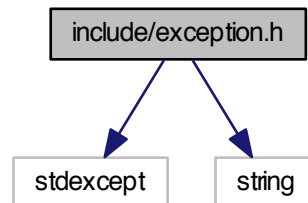
- `template<typename Scalar >`  
`double qpp::shannon (const types::DynMat< Scalar > &A)`
- `template<typename Scalar >`  
`double qpp::renyi (const double alpha, const types::DynMat< Scalar > &A)`
- `template<typename Scalar >`  
`double qpp::renyi\_inf (const types::DynMat< Scalar > &A)`

### 7.3 include/exception.h File Reference

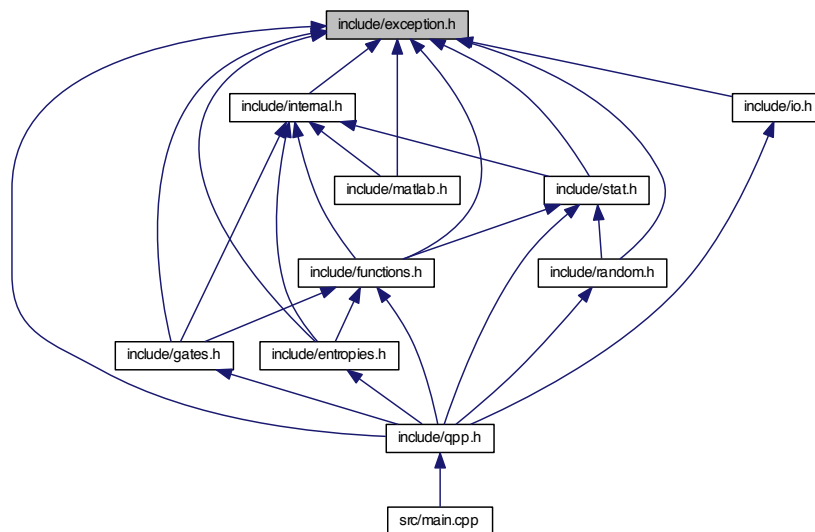
```
#include <stdexcept>
```

```
#include <string>
```

Include dependency graph for exception.h:



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



#### Classes

- class [qpp::Exception](#)

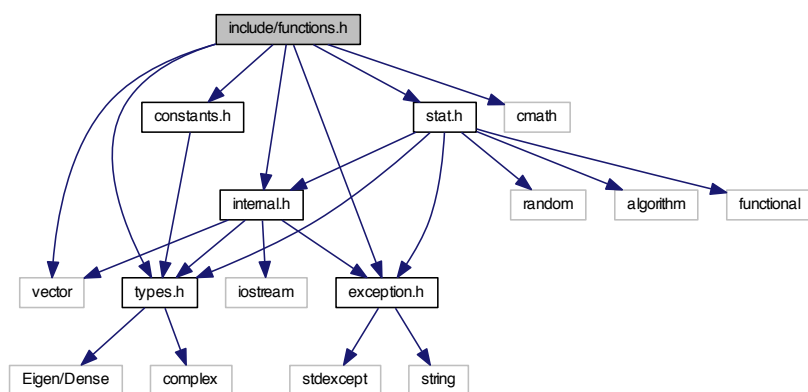
#### Namespaces

- [qpp](#)

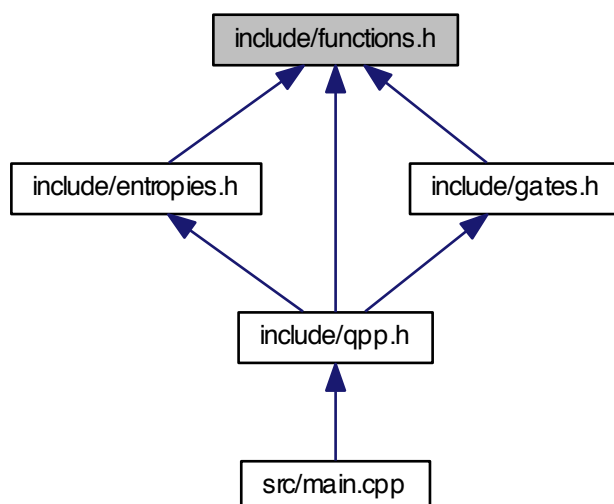
## 7.4 include/functions.h File Reference

```
#include <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](#)

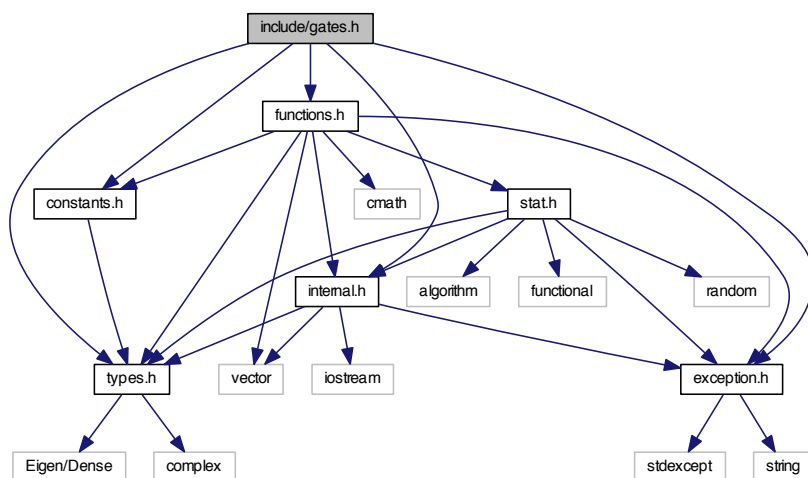
## Functions

- `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::powm (const types::DynMat< Scalar > &A, const types::cplx z)`
- `template<typename Scalar >`  
`types::DynMat< Scalar > qpp::powm\_int (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::kron\_list (const std::vector< types::DynMat< Scalar > > &list)`
- `template<typename Scalar >`  
`types::DynMat< Scalar > qpp::kron\_pow (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)`

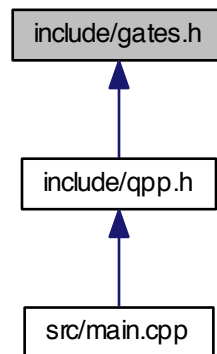
- `template<typename Scalar >`  
`types::DynMat< Scalar > qpp::syspermute (const types::DynMat< Scalar > &A, const std::vector< size_t > > perm, const std::vector< size_t > &dims)`
- `template<typename Scalar >`  
`types::DynMat< Scalar > qpp::ptrace2 (const types::DynMat< Scalar > &A, const std::vector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

## 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

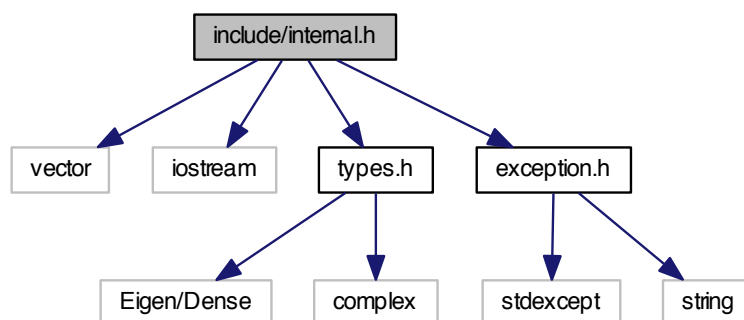
- void [qpp::gt::\\_init\\_gates](#) ()
- types::cmat [qpp::gt::Rtheta](#) (double theta)
- types::cmat [qpp::gt::CU](#) (const types::cmat &U)
- types::cmat [qpp::gt::Zd](#) (size\_t D)
- types::cmat [qpp::gt::Fd](#) (size\_t D)
- types::cmat [qpp::gt::Xd](#) (size\_t D)
- types::cmat [qpp::gt::CUd](#) (const types::cmat &U)

## Variables

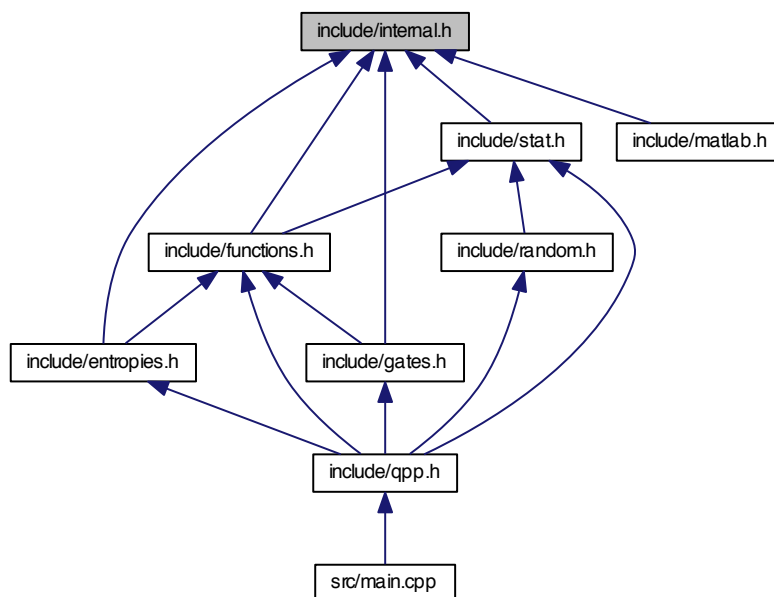
- types::cmat [qpp::gt::H](#)
- types::cmat [qpp::gt::Id2](#)
- types::cmat [qpp::gt::X](#)
- types::cmat [qpp::gt::Y](#)
- types::cmat [qpp::gt::Z](#)
- types::cmat [qpp::gt::S](#)
- types::cmat [qpp::gt::T](#)
- types::cmat [qpp::gt::CNOT](#)
- types::cmat [qpp::gt::CP](#)
- types::cmat [qpp::gt::TOF](#)

## 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`

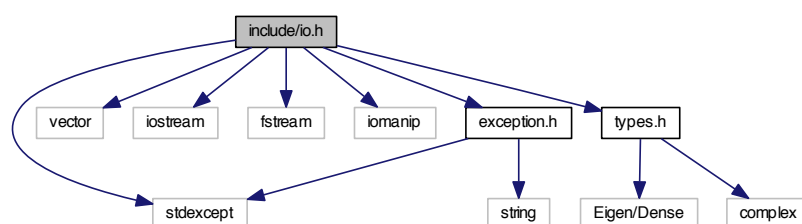
## Functions

- 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\\_nonzero\\_size](#) (const types::DynMat< Scalar > &A)
- 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\\_dims](#) (const std::vector< size\_t > &dims)
- 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

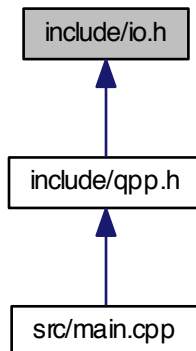
```
#include <stdexcept>
#include <vector>
#include <iostream>
#include <fstream>
#include <iomanip>
#include "types.h"
#include "exception.h"
```

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

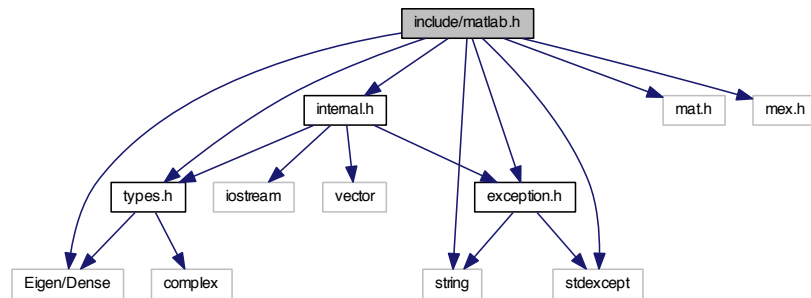
## 7.8 include/matlab.h File Reference

```

#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"

```

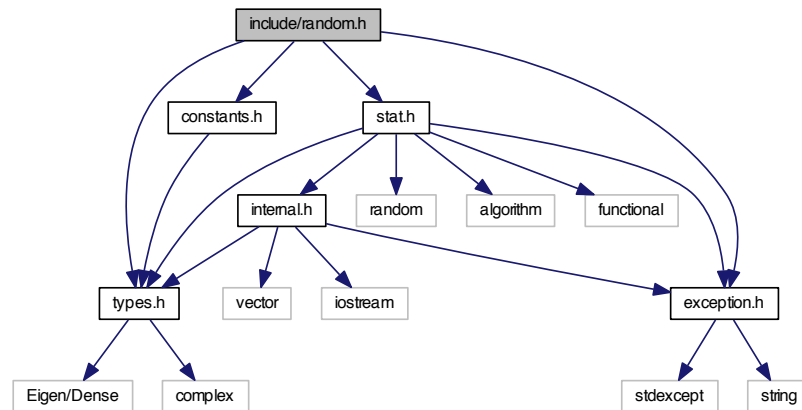
```
graph BT; src/main.cpp --> include/qpp.h
```

- qpp
- qpp::gt

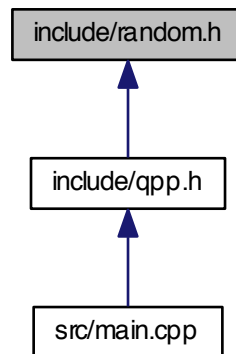
- types::cmat `qpp::gt::TOF` (8, 8)
- int `qpp::init` ()

```
#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](#)

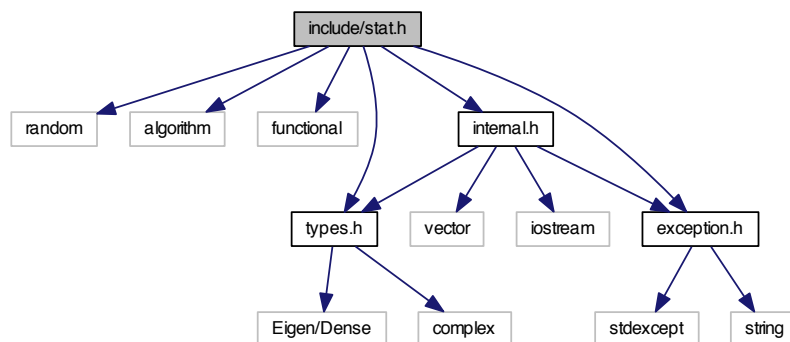
## Functions

- `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)`

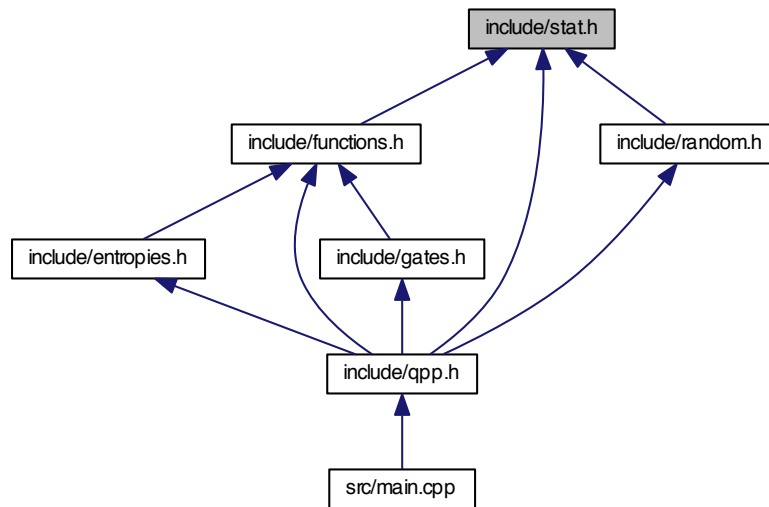
- `template<typename Scalar >`  
`types::DynMat< Scalar > qpp::randn (size_t rows, size_t cols, double mean=0, double sigma=1)`
- `template<>`  
`types::DynMat< double > qpp::randn (size_t rows, size_t cols, double mean, double sigma)`
- `template<>`  
`types::DynMat< types::cplx > qpp::randn (size_t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

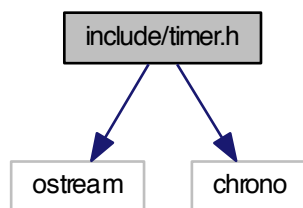
## Variables

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

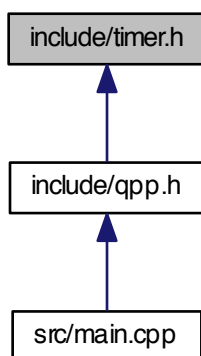
## 7.12 include/timer.h File Reference

```
#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`

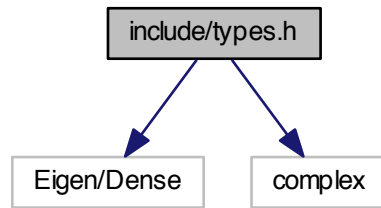
## Namespaces

- `qpp`

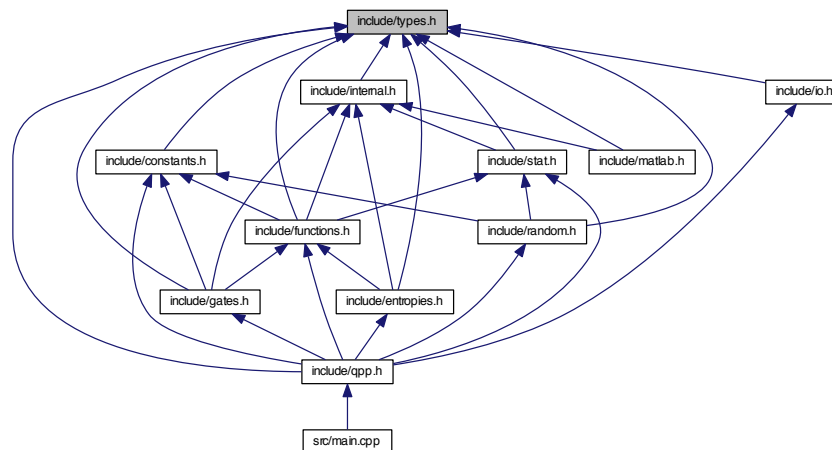
## 7.13 include/types.h File Reference

```
#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](#)

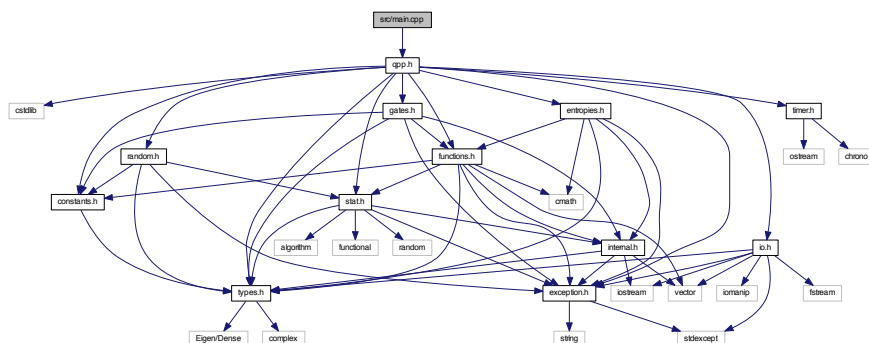
## Typedefs

- `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 >`



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

Include dependency graph for main.cpp:



- `int main ()`

#### 7.14.1.1 int main ( )

```

graph LR
    main((main)) --> qpp_init[qpp::init]
    main --> qpp_disp[qpp::disp]
    main --> qpp_stat[qpp::stat::DiscreteDistributionFromComplex::probabilities]
    main --> qpp_timer_toc[qpp::Timer::toc]
    main --> qpp_timer_tic[qpp::Timer::tic]
    main --> qpp_prace[qpp::prace]
    main --> qpp_prace2[qpp::prace2]
    main --> qpp_transpose[qpp::transpose]
    main --> qpp_timer_seconds[qpp::Timer::seconds]
    main --> qpp_gt_init_gates[qpp::gt::_init_gates]

    qpp_init --> qpp_gt_init_gates
    qpp_stat --> qpp_gt_init_gates
    qpp_timer_toc --> qpp_systemmute[qpp::systemmute]
    qpp_timer_tic --> qpp_systemmute
    qpp_prace --> qpp_systemmute
    qpp_prace2 --> qpp_systemmute
    qpp_transpose --> qpp_systemmute
    qpp_timer_seconds --> qpp_systemmute

    qpp_systemmute --> qpp_internal_check_perm[qpp::internal::_check_perm]
    qpp_systemmute --> qpp_internal_systemmute_worker[qpp::internal::_systemmute_worker]
    qpp_systemmute --> qpp_internal_check_dims_match_mat[qpp::internal::_check_dims_match_mat]
    qpp_systemmute --> qpp_internal_check_square_mat[qpp::internal::_check_square_mat]
    qpp_systemmute --> qpp_internal_check_dims[qpp::internal::_check_dims]
    qpp_systemmute --> qpp_internal_check_nonzero_size[qpp::internal::_check_nonzero_size]
    qpp_systemmute --> qpp_internal_transpose_worker[qpp::internal::_transpose_worker]
    qpp_systemmute --> qpp_internal_check_subsys[qpp::internal::_check_subsys]

    qpp_internal_check_perm --> qpp_internal_n2multidx[qpp::internal::_n2multidx]
    qpp_internal_systemmute_worker --> qpp_internal_n2multidx
    qpp_internal_check_dims_match_mat --> qpp_internal_n2multidx
    qpp_internal_check_square_mat --> qpp_internal_n2multidx
    qpp_internal_check_dims --> qpp_internal_n2multidx
    qpp_internal_check_nonzero_size --> qpp_internal_n2multidx
    qpp_internal_transpose_worker --> qpp_internal_multidx2n[qpp::internal::_multidx2n]
    qpp_internal_check_subsys --> qpp_internal_multidx2n
    qpp_internal_n2multidx --> qpp_internal_multidx2n
  
```