

qpp  
0.1

Generated by Doxygen 1.8.7

Thu Oct 23 2014 12:04:37



# 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 . . . . .	14
5.1.1.1	absm . . . . .	14
5.1.1.2	adjoint . . . . .	15
5.1.1.3	anticomm . . . . .	15
5.1.1.4	channel . . . . .	16
5.1.1.5	channel . . . . .	17
5.1.1.6	choi . . . . .	17
5.1.1.7	choi2kraus . . . . .	18
5.1.1.8	comm . . . . .	18
5.1.1.9	compperm . . . . .	19
5.1.1.10	conjugate . . . . .	20
5.1.1.11	cosm . . . . .	20
5.1.1.12	cwise . . . . .	21
5.1.1.13	det . . . . .	21
5.1.1.14	disp . . . . .	22
5.1.1.15	disp . . . . .	22
5.1.1.16	disp . . . . .	22
5.1.1.17	disp . . . . .	22
5.1.1.18	displn . . . . .	23

5.1.1.19	displn	23
5.1.1.20	displn	23
5.1.1.21	displn	24
5.1.1.22	entanglement	24
5.1.1.23	evals	24
5.1.1.24	evects	25
5.1.1.25	expandout	25
5.1.1.26	expm	26
5.1.1.27	funm	27
5.1.1.28	gconcurrency	28
5.1.1.29	grams	28
5.1.1.30	grams	28
5.1.1.31	grams	29
5.1.1.32	hevals	29
5.1.1.33	hevects	30
5.1.1.34	inverse	30
5.1.1.35	invperm	31
5.1.1.36	kron	31
5.1.1.37	kron	32
5.1.1.38	kron	32
5.1.1.39	kron	33
5.1.1.40	kronpow	33
5.1.1.41	load	34
5.1.1.42	loadMATLABmatrix	34
5.1.1.43	loadMATLABmatrix	34
5.1.1.44	loadMATLABmatrix	34
5.1.1.45	logdet	34
5.1.1.46	logm	35
5.1.1.47	mket	35
5.1.1.48	mket	36
5.1.1.49	mket	36
5.1.1.50	multiidx2n	37
5.1.1.51	n2multiidx	37
5.1.1.52	norm	38
5.1.1.53	operator""_i	38
5.1.1.54	operator""_i	39
5.1.1.55	powm	39
5.1.1.56	prj	39
5.1.1.57	ptrace	40
5.1.1.58	ptrace1	41

5.1.1.59	<a href="#">ptrace2</a>	42
5.1.1.60	<a href="#">ptranspose</a>	43
5.1.1.61	<a href="#">qmutualinfo</a>	45
5.1.1.62	<a href="#">rand</a>	45
5.1.1.63	<a href="#">rand</a>	45
5.1.1.64	<a href="#">rand</a>	46
5.1.1.65	<a href="#">rand</a>	46
5.1.1.66	<a href="#">randH</a>	46
5.1.1.67	<a href="#">randint</a>	46
5.1.1.68	<a href="#">randket</a>	47
5.1.1.69	<a href="#">randkraus</a>	47
5.1.1.70	<a href="#">randn</a>	47
5.1.1.71	<a href="#">randn</a>	47
5.1.1.72	<a href="#">randn</a>	48
5.1.1.73	<a href="#">randn</a>	48
5.1.1.74	<a href="#">randperm</a>	48
5.1.1.75	<a href="#">randrho</a>	49
5.1.1.76	<a href="#">randU</a>	49
5.1.1.77	<a href="#">randV</a>	49
5.1.1.78	<a href="#">renyi</a>	49
5.1.1.79	<a href="#">renyi_inf</a>	50
5.1.1.80	<a href="#">reshape</a>	50
5.1.1.81	<a href="#">save</a>	50
5.1.1.82	<a href="#">saveMATLABmatrix</a>	50
5.1.1.83	<a href="#">saveMATLABmatrix</a>	51
5.1.1.84	<a href="#">saveMATLABmatrix</a>	51
5.1.1.85	<a href="#">schmidtcoeff</a>	51
5.1.1.86	<a href="#">schmidtprob</a>	52
5.1.1.87	<a href="#">schmidtU</a>	52
5.1.1.88	<a href="#">schmidtV</a>	53
5.1.1.89	<a href="#">shannon</a>	53
5.1.1.90	<a href="#">sinm</a>	53
5.1.1.91	<a href="#">spectralpowm</a>	54
5.1.1.92	<a href="#">sqrtm</a>	55
5.1.1.93	<a href="#">sum</a>	56
5.1.1.94	<a href="#">super</a>	56
5.1.1.95	<a href="#">syspermute</a>	57
5.1.1.96	<a href="#">trace</a>	58
5.1.1.97	<a href="#">transpose</a>	59
5.1.1.98	<a href="#">tsallis</a>	60

5.1.2	Variable Documentation	60
5.1.2.1	gt	60
5.1.2.2	rdevs	60
5.1.2.3	st	60
5.2	qpp::ct Namespace Reference	60
5.2.1	Function Documentation	61
5.2.1.1	omega	61
5.2.2	Variable Documentation	61
5.2.2.1	chop	61
5.2.2.2	ee	61
5.2.2.3	eps	61
5.2.2.4	maxn	61
5.2.2.5	pi	61
5.3	qpp::internal Namespace Reference	62
5.3.1	Function Documentation	62
5.3.1.1	_check_col_vector	62
5.3.1.2	_check_dims	62
5.3.1.3	_check_dims_match_cvect	62
5.3.1.4	_check_dims_match_mat	62
5.3.1.5	_check_dims_match_rvect	62
5.3.1.6	_check_eq_dims	62
5.3.1.7	_check_nonzero_size	63
5.3.1.8	_check_perm	63
5.3.1.9	_check_row_vector	63
5.3.1.10	_check_square_mat	63
5.3.1.11	_check_subsys_match_dims	63
5.3.1.12	_check_vector	63
5.3.1.13	_kron2	63
5.3.1.14	_multiidx2n	63
5.3.1.15	_n2multiidx	63
5.3.1.16	variadic_vector_emplace	63
5.3.1.17	variadic_vector_emplace	63
5.4	qpp::types Namespace Reference	63
5.4.1	Typedef Documentation	64
5.4.1.1	bra	64
5.4.1.2	cmat	64
5.4.1.3	cplx	64
5.4.1.4	dmat	64
5.4.1.5	DynMat	64
5.4.1.6	ket	64

<b>6</b>	<b>Class Documentation</b>	<b>65</b>
6.1	qpp::DiscreteDistribution Class Reference	65
6.1.1	Constructor & Destructor Documentation	65
6.1.1.1	DiscreteDistribution	65
6.1.1.2	DiscreteDistribution	65
6.1.1.3	DiscreteDistribution	65
6.1.2	Member Function Documentation	65
6.1.2.1	probabilities	65
6.1.2.2	sample	66
6.1.3	Member Data Documentation	66
6.1.3.1	_d	66
6.2	qpp::DiscreteDistributionAbsSquare Class Reference	66
6.2.1	Constructor & Destructor Documentation	67
6.2.1.1	DiscreteDistributionAbsSquare	67
6.2.1.2	DiscreteDistributionAbsSquare	67
6.2.1.3	DiscreteDistributionAbsSquare	67
6.2.1.4	DiscreteDistributionAbsSquare	67
6.2.2	Member Function Documentation	67
6.2.2.1	cplx2weights	67
6.2.2.2	probabilities	67
6.2.2.3	sample	67
6.2.3	Member Data Documentation	67
6.2.3.1	_d	67
6.3	qpp::Exception Class Reference	67
6.3.1	Member Enumeration Documentation	69
6.3.1.1	Type	69
6.3.2	Constructor & Destructor Documentation	70
6.3.2.1	Exception	70
6.3.2.2	Exception	70
6.3.3	Member Function Documentation	70
6.3.3.1	_construct_exception_msg	70
6.3.3.2	what	70
6.3.4	Member Data Documentation	70
6.3.4.1	_custom	70
6.3.4.2	_msg	70
6.3.4.3	_type	70
6.3.4.4	_where	70
6.4	qpp::Gates Class Reference	70
6.4.1	Constructor & Destructor Documentation	72
6.4.1.1	Gates	72

6.4.2	Member Function Documentation	72
6.4.2.1	apply	73
6.4.2.2	applyCTRL	73
6.4.2.3	CTRL	74
6.4.2.4	Fd	74
6.4.2.5	Id	74
6.4.2.6	Rn	74
6.4.2.7	Xd	75
6.4.2.8	Zd	75
6.4.3	Friends And Related Function Documentation	75
6.4.3.1	Singleton< const Gates >	75
6.4.4	Member Data Documentation	75
6.4.4.1	CNOTab	75
6.4.4.2	CNOTba	75
6.4.4.3	CZ	75
6.4.4.4	FRED	75
6.4.4.5	H	75
6.4.4.6	Id2	75
6.4.4.7	S	75
6.4.4.8	SWAP	75
6.4.4.9	T	75
6.4.4.10	TOF	76
6.4.4.11	X	76
6.4.4.12	Y	76
6.4.4.13	Z	76
6.5	qpp::NormalDistribution Class Reference	76
6.5.1	Constructor & Destructor Documentation	76
6.5.1.1	NormalDistribution	76
6.5.2	Member Function Documentation	76
6.5.2.1	sample	76
6.5.3	Member Data Documentation	76
6.5.3.1	_d	76
6.6	qpp::Qudit Class Reference	77
6.6.1	Constructor & Destructor Documentation	77
6.6.1.1	Qudit	77
6.6.2	Member Function Documentation	77
6.6.2.1	getD	77
6.6.2.2	getRho	77
6.6.2.3	measure	78
6.6.2.4	measure	78



6.6.3	Member Data Documentation . . . . .	78
6.6.3.1	_D . . . . .	78
6.6.3.2	_rho . . . . .	78
6.7	qpp::RandomDevices Class Reference . . . . .	79
6.7.1	Constructor & Destructor Documentation . . . . .	80
6.7.1.1	RandomDevices . . . . .	80
6.7.2	Friends And Related Function Documentation . . . . .	80
6.7.2.1	Singleton< RandomDevices > . . . . .	80
6.7.3	Member Data Documentation . . . . .	80
6.7.3.1	_rd . . . . .	80
6.7.3.2	_rng . . . . .	80
6.8	qpp::Singleton< T > Class Template Reference . . . . .	80
6.8.1	Constructor & Destructor Documentation . . . . .	81
6.8.1.1	Singleton . . . . .	81
6.8.1.2	~Singleton . . . . .	81
6.8.1.3	Singleton . . . . .	81
6.8.2	Member Function Documentation . . . . .	81
6.8.2.1	get_instance . . . . .	81
6.8.2.2	operator= . . . . .	81
6.9	qpp::States Class Reference . . . . .	81
6.9.1	Constructor & Destructor Documentation . . . . .	83
6.9.1.1	States . . . . .	83
6.9.2	Friends And Related Function Documentation . . . . .	83
6.9.2.1	Singleton< const States > . . . . .	83
6.9.3	Member Data Documentation . . . . .	83
6.9.3.1	b00 . . . . .	83
6.9.3.2	b01 . . . . .	83
6.9.3.3	b10 . . . . .	83
6.9.3.4	b11 . . . . .	83
6.9.3.5	GHZ . . . . .	83
6.9.3.6	pb00 . . . . .	83
6.9.3.7	pb01 . . . . .	83
6.9.3.8	pb10 . . . . .	83
6.9.3.9	pb11 . . . . .	83
6.9.3.10	pGHZ . . . . .	83
6.9.3.11	pW . . . . .	83
6.9.3.12	px0 . . . . .	83
6.9.3.13	px1 . . . . .	83
6.9.3.14	py0 . . . . .	83
6.9.3.15	py1 . . . . .	83

6.9.3.16	pz0 . . . . .	83
6.9.3.17	pz1 . . . . .	83
6.9.3.18	W . . . . .	83
6.9.3.19	x0 . . . . .	83
6.9.3.20	x1 . . . . .	83
6.9.3.21	y0 . . . . .	83
6.9.3.22	y1 . . . . .	83
6.9.3.23	z0 . . . . .	84
6.9.3.24	z1 . . . . .	84
6.10	qpp::Timer Class Reference . . . . .	84
6.10.1	Constructor & Destructor Documentation . . . . .	84
6.10.1.1	Timer . . . . .	84
6.10.2	Member Function Documentation . . . . .	84
6.10.2.1	seconds . . . . .	84
6.10.2.2	tic . . . . .	84
6.10.2.3	toc . . . . .	84
6.10.3	Friends And Related Function Documentation . . . . .	84
6.10.3.1	operator<< . . . . .	84
6.10.4	Member Data Documentation . . . . .	84
6.10.4.1	_end . . . . .	84
6.10.4.2	_start . . . . .	84
6.11	qpp::UniformIntDistribution Class Reference . . . . .	85
6.11.1	Constructor & Destructor Documentation . . . . .	85
6.11.1.1	UniformIntDistribution . . . . .	85
6.11.2	Member Function Documentation . . . . .	85
6.11.2.1	sample . . . . .	85
6.11.3	Member Data Documentation . . . . .	85
6.11.3.1	_d . . . . .	85
6.12	qpp::UniformRealDistribution Class Reference . . . . .	85
6.12.1	Constructor & Destructor Documentation . . . . .	86
6.12.1.1	UniformRealDistribution . . . . .	86
6.12.2	Member Function Documentation . . . . .	86
6.12.2.1	sample . . . . .	86
6.12.3	Member Data Documentation . . . . .	86
6.12.3.1	_d . . . . .	86
<b>7</b>	<b>File Documentation</b>	<b>87</b>
7.1	include/channels.h File Reference . . . . .	87
7.2	include/classes/exception.h File Reference . . . . .	88
7.3	include/classes/gates.h File Reference . . . . .	88

7.4	include/classes/qudit.h File Reference	89
7.5	include/classes/randevs.h File Reference	89
7.6	include/classes/singleton.h File Reference	90
7.6.1	Macro Definition Documentation	90
7.6.1.1	CLASS_CONST_SINGLETON	90
7.6.1.2	CLASS_SINGLETON	90
7.7	include/classes/stat.h File Reference	91
7.8	include/classes/states.h File Reference	91
7.9	include/classes/timer.h File Reference	92
7.10	include/constants.h File Reference	92
7.11	include/entanglement.h File Reference	93
7.12	include/entropies.h File Reference	94
7.13	include/functions.h File Reference	95
7.14	include/internal.h File Reference	98
7.15	include/io.h File Reference	100
7.16	include/matlab.h File Reference	100
7.17	include/qpp.h File Reference	102
7.18	include/random.h File Reference	103
7.19	include/types.h File Reference	104
	<b>Index</b>	<b>105</b>



# 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>	60
<a href="#">qpp::internal</a>	62
<a href="#">qpp::types</a>	63



## Chapter 2

# Hierarchical Index

### 2.1 Class Hierarchy

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

qpp::DiscreteDistribution . . . . .	65
qpp::DiscreteDistributionAbsSquare . . . . .	66
exception	
qpp::Exception . . . . .	67
qpp::NormalDistribution . . . . .	76
qpp::Qudit . . . . .	77
qpp::Singleton< T > . . . . .	80
qpp::Gates . . . . .	70
qpp::RandomDevices . . . . .	79
qpp::Singleton< const Gates > . . . . .	80
qpp::Singleton< const States > . . . . .	80
qpp::States . . . . .	81
qpp::Singleton< RandomDevices > . . . . .	80
qpp::Timer . . . . .	84
qpp::UniformIntDistribution . . . . .	85
qpp::UniformRealDistribution . . . . .	85





## Chapter 3

# Class Index

### 3.1 Class List

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

<a href="#">qpp::DiscreteDistribution</a>	65
<a href="#">qpp::DiscreteDistributionAbsSquare</a>	66
<a href="#">qpp::Exception</a>	67
<a href="#">qpp::Gates</a>	70
<a href="#">qpp::NormalDistribution</a>	76
<a href="#">qpp::Qudit</a>	77
<a href="#">qpp::RandomDevices</a>	79
<a href="#">qpp::Singleton&lt; T &gt;</a>	80
<a href="#">qpp::States</a>	81
<a href="#">qpp::Timer</a>	84
<a href="#">qpp::UniformIntDistribution</a>	85
<a href="#">qpp::UniformRealDistribution</a>	85



## Chapter 4

# File Index

### 4.1 File List

Here is a list of all files with brief descriptions:

include/channels.h	87
include/constants.h	92
include/entanglement.h	93
include/entropies.h	94
include/functions.h	95
include/internal.h	98
include/io.h	100
include/matlab.h	100
include/qpp.h	102
include/random.h	103
include/types.h	104
include/classes/exception.h	88
include/classes/gates.h	88
include/classes/qudit.h	89
include/classes/randevs.h	89
include/classes/singleton.h	90
include/classes/stat.h	91
include/classes/states.h	91
include/classes/timer.h	92



## Chapter 5

# Namespace Documentation

### 5.1 qpp Namespace Reference

#### Namespaces

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

#### Classes

- class [DiscreteDistribution](#)
- class [DiscreteDistributionAbsSquare](#)
- class [Exception](#)
- class [Gates](#)
- class [NormalDistribution](#)
- class [Qudit](#)
- class [RandomDevices](#)
- class [Singleton](#)
- class [States](#)
- class [Timer](#)
- class [UniformIntDistribution](#)
- class [UniformRealDistribution](#)

#### Functions

- [types::cmat super](#) (const std::vector< [types::cmat](#) > &Ks)
- [types::cmat choi](#) (const std::vector< [types::cmat](#) > &Ks)
- std::vector< [types::cmat](#) > [choi2kraus](#) (const [types::cmat](#) &A)
- template<typename Derived >  
[types::cmat channel](#) (const Eigen::MatrixBase< Derived > &rho, const std::vector< [types::cmat](#) > &Ks)
- template<typename Derived >  
[types::cmat channel](#) (const Eigen::MatrixBase< Derived > &rho, const std::vector< [types::cmat](#) > &Ks,  
const std::vector< std::size\_t > &subsys, const std::vector< std::size\_t > &dims)
- constexpr std::complex< double > [operator""\\_i](#) (unsigned long long int x)  
*User-defined literal for complex  $i = \sqrt{-1}$  (integer overload)*
- constexpr std::complex< double > [operator""\\_i](#) (long double x)  
*User-defined literal for complex  $i = \sqrt{-1}$  (real overload)*

- `template<typename Derived >`  
`types::cmat schmidtcoeff` (const Eigen::MatrixBase< Derived > &A, const std::vector< std::size\_t > &dims)
- `template<typename Derived >`  
`types::cmat schmidtU` (const Eigen::MatrixBase< Derived > &A, const std::vector< std::size\_t > &dims)
- `template<typename Derived >`  
`types::cmat schmidtV` (const Eigen::MatrixBase< Derived > &A, const std::vector< std::size\_t > &dims)
- `template<typename Derived >`  
`types::cmat schmidtprob` (const Eigen::MatrixBase< Derived > &A, const std::vector< std::size\_t > &dims)
- `template<typename Derived >`  
`double entanglement` (const Eigen::MatrixBase< Derived > &A, const std::vector< std::size\_t > &dims)
- `template<typename Derived >`  
`double gconcurrency` (const Eigen::MatrixBase< Derived > &A)
- `template<typename Derived >`  
`double shannon` (const Eigen::MatrixBase< Derived > &A)
- `template<typename Derived >`  
`double renyi` (const double alpha, const Eigen::MatrixBase< Derived > &A)
- `template<typename Derived >`  
`double renyi_inf` (const Eigen::MatrixBase< Derived > &A)
- `template<typename Derived >`  
`double tsallis` (const double alpha, const Eigen::MatrixBase< Derived > &A)
- `template<typename Derived >`  
`double qmutualinfo` (const Eigen::MatrixBase< Derived > &A, const std::vector< std::size\_t > &subs, const std::vector< std::size\_t > &dims)
- `template<typename Derived >`  
`types::DynMat< typename`  
`Derived::Scalar > transpose` (const Eigen::MatrixBase< Derived > &A)  
*Transpose.*
- `template<typename Derived >`  
`types::DynMat< typename`  
`Derived::Scalar > conjugate` (const Eigen::MatrixBase< Derived > &A)  
*Complex conjugate.*
- `template<typename Derived >`  
`types::DynMat< typename`  
`Derived::Scalar > adjoint` (const Eigen::MatrixBase< Derived > &A)  
*Adjoint.*
- `template<typename Derived >`  
`types::DynMat< typename`  
`Derived::Scalar > inverse` (const Eigen::MatrixBase< Derived > &A)  
*Inverse.*
- `template<typename Derived >`  
`Derived::Scalar trace` (const Eigen::MatrixBase< Derived > &A)  
*Trace.*
- `template<typename Derived >`  
`Derived::Scalar det` (const Eigen::MatrixBase< Derived > &A)  
*Determinant.*
- `template<typename Derived >`  
`Derived::Scalar logdet` (const Eigen::MatrixBase< Derived > &A)  
*Logarithm of the determinant.*
- `template<typename Derived >`  
`Derived::Scalar sum` (const Eigen::MatrixBase< Derived > &A)  
*Element-wise sum.*
- `template<typename Derived >`  
`double norm` (const Eigen::MatrixBase< Derived > &A)  
*Trace norm.*

- `template<typename Derived >`  
`types::cmat evals` (const Eigen::MatrixBase< Derived > &A)  
*Eigenvalues.*
- `template<typename Derived >`  
`types::cmat evects` (const Eigen::MatrixBase< Derived > &A)  
*Eigenvectors.*
- `template<typename Derived >`  
`types::dmat hevals` (const Eigen::MatrixBase< Derived > &A)  
*Hermitian eigenvalues.*
- `template<typename Derived >`  
`types::cmat hevects` (const Eigen::MatrixBase< Derived > &A)  
*Hermitian eigenvectors.*
- `template<typename Derived >`  
`types::cmat funm` (const Eigen::MatrixBase< Derived > &A, `types::cplx`(\*f)(const `types::cplx` &))  
*Functional calculus  $f(A)$*
- `template<typename Derived >`  
`types::cmat sqrtm` (const Eigen::MatrixBase< Derived > &A)  
*Matrix square root.*
- `template<typename Derived >`  
`types::cmat absm` (const Eigen::MatrixBase< Derived > &A)  
*Matrix absolut value.*
- `template<typename Derived >`  
`types::cmat expm` (const Eigen::MatrixBase< Derived > &A)  
*Matrix exponential.*
- `template<typename Derived >`  
`types::cmat logm` (const Eigen::MatrixBase< Derived > &A)  
*Matrix logarithm.*
- `template<typename Derived >`  
`types::cmat sinm` (const Eigen::MatrixBase< Derived > &A)  
*Matrix sin.*
- `template<typename Derived >`  
`types::cmat cosm` (const Eigen::MatrixBase< Derived > &A)  
*Matrix cos.*
- `template<typename Derived >`  
`types::cmat spectralpowm` (const Eigen::MatrixBase< Derived > &A, const `types::cplx` z)  
*Matrix power.*
- `template<typename Derived >`  
`types::DynMat< typename`  
`Derived::Scalar > powm` (const Eigen::MatrixBase< Derived > &A, `std::size_t` n)  
*Matrix power.*
- `template<typename OutputScalar , typename Derived >`  
`types::DynMat< OutputScalar > cwise` (const Eigen::MatrixBase< Derived > &A, `OutputScalar`(\*f)(const `typename Derived::Scalar` &))  
*Functor.*
- `template<typename T >`  
`types::DynMat< typename T::Scalar > kron` (const T &head)  
*Kronecker product (variadic overload)*
- `template<typename T , typename... Args>`  
`types::DynMat< typename T::Scalar > kron` (const T &head, const Args &...tail)  
*Kronecker product (variadic overload)*
- `template<typename Derived >`  
`types::DynMat< typename`  
`Derived::Scalar > kron` (const `std::vector< Derived >` &As)

- Kronecker product (std::vector overload)*

  - template<typename Derived >  
[types::DynMat](#)< typename  
 Derived::Scalar > [kron](#) (const std::initializer\_list< Derived > &As)

*Kronecker product (std::initializer\_list overload)*

  - template<typename Derived >  
[types::DynMat](#)< typename  
 Derived::Scalar > [kronpow](#) (const Eigen::MatrixBase< Derived > &A, std::size\_t n)

*Kronecker power.*

  - template<typename Derived >  
[types::DynMat](#)< typename  
 Derived::Scalar > [reshape](#) (const Eigen::MatrixBase< Derived > &A, std::size\_t rows, std::size\_t cols)

*Reshape.*

  - template<typename Derived >  
[types::DynMat](#)< typename  
 Derived::Scalar > [syspermute](#) (const Eigen::MatrixBase< Derived > &A, const std::vector< std::size\_t > &perm, const std::vector< std::size\_t > &dims)

*System permutation.*

  - template<typename Derived >  
[types::DynMat](#)< typename  
 Derived::Scalar > [ptrace1](#) (const Eigen::MatrixBase< Derived > &A, const std::vector< std::size\_t > &dims)

*Partial trace.*

  - template<typename Derived >  
[types::DynMat](#)< typename  
 Derived::Scalar > [ptrace2](#) (const Eigen::MatrixBase< Derived > &A, const std::vector< std::size\_t > &dims)

*Partial trace.*

  - template<typename Derived >  
[types::DynMat](#)< typename  
 Derived::Scalar > [ptrace](#) (const Eigen::MatrixBase< Derived > &A, const std::vector< std::size\_t > &subsys, const std::vector< std::size\_t > &dims)

*Partial trace.*

  - template<typename Derived >  
[types::DynMat](#)< typename  
 Derived::Scalar > [ptranspose](#) (const Eigen::MatrixBase< Derived > &A, const std::vector< std::size\_t > &subsys, const std::vector< std::size\_t > &dims)

*Partial transpose.*

  - template<typename Derived1 , typename Derived2 >  
[types::DynMat](#)< typename  
 Derived1::Scalar > [comm](#) (const Eigen::MatrixBase< Derived1 > &A, const Eigen::MatrixBase< Derived2 > &B)

*Commutator.*

  - template<typename Derived1 , typename Derived2 >  
[types::DynMat](#)< typename  
 Derived1::Scalar > [anticomm](#) (const Eigen::MatrixBase< Derived1 > &A, const Eigen::MatrixBase< Derived2 > &B)

*Anti-commutator.*

  - template<typename Derived >  
[types::DynMat](#)< typename  
 Derived::Scalar > [prj](#) (const Eigen::MatrixBase< Derived > &V)

*Projector.*

  - template<typename Derived >  
[types::DynMat](#)< typename  
 Derived::Scalar > [expandout](#) (const Eigen::MatrixBase< Derived > &A, std::size\_t pos, const std::vector< std::size\_t > &dims)

*Expand out.*



- `template<typename Derived >`  
`types::DynMat< typename`  
`Derived::Scalar > grams` (const std::vector< Derived > &Vs)  
*Gram-Schmidt orthogonalization (std::vector overload)*
- `template<typename Derived >`  
`types::DynMat< typename`  
`Derived::Scalar > grams` (const std::initializer\_list< Derived > &Vs)  
*Gram-Schmidt orthogonalization (std::initializer\_list overload)*
- `template<typename Derived >`  
`types::DynMat< typename`  
`Derived::Scalar > grams` (const Eigen::MatrixBase< Derived > &A)  
*Gram-Schmidt orthogonalization (Eigen expression (matrix) overload)*
- `std::vector< std::size_t > n2multiidx` (std::size\_t n, const std::vector< std::size\_t > &dims)  
*Non-negative integer index to multi-index.*
- `std::size_t multiidx2n` (const std::vector< std::size\_t > &midx, const std::vector< std::size\_t > &dims)  
*Multi-index to non-negative integer index.*
- `types::ket mket` (const std::vector< std::size\_t > &mask)  
*Multi-partite qubit ket.*
- `types::ket mket` (const std::vector< std::size\_t > &mask, const std::vector< std::size\_t > &dims)  
*Multi-partite qudit ket (different dimensions overload)*
- `types::ket mket` (const std::vector< std::size\_t > &mask, std::size\_t d)  
*Multi-partite qudit ket (same dimensions overload)*
- `std::vector< std::size_t > invperm` (const std::vector< std::size\_t > &perm)  
*Inverse permutation.*
- `std::vector< std::size_t > compperm` (const std::vector< std::size\_t > &perm, const std::vector< std::size\_t > &sigma)  
*Compose permutations.*
- `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 std::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 std::size\_t n, const std::string &separator, const std::string &start="[" , const std::string &end="]",  
std::ostream &os=std::cout)
- `template<typename Derived >`  
`void disp` (const Eigen::MatrixBase< Derived > &A, double chop=ct::chop, std::ostream &os=std::cout)
- `template<typename Derived >`  
`void displn` (const Eigen::MatrixBase< Derived > &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 Derived >`  
`void save` (const Eigen::MatrixBase< Derived > &A, const std::string &fname)
- `template<typename Derived >`  
`types::DynMat< typename`  
`Derived::Scalar > load` (const std::string &fname)
- `template<typename Derived >`  
`Derived loadMATLABmatrix` (const std::string &mat\_file, const std::string &var\_name)
- `template<>`  
`types::dmat loadMATLABmatrix` (const std::string &mat\_file, const std::string &var\_name)

- `template<>`  
`types::cmat loadMATLABmatrix` (const std::string &mat\_file, const std::string &var\_name)
- `template<typename Derived >`  
void `saveMATLABmatrix` (const Eigen::MatrixBase< Derived > &A, const std::string &mat\_file, const std::string &var\_name, const std::string &mode)
- `template<>`  
void `saveMATLABmatrix` (const Eigen::MatrixBase< types::dmat > &A, const std::string &mat\_file, const std::string &var\_name, const std::string &mode)
- `template<>`  
void `saveMATLABmatrix` (const Eigen::MatrixBase< typename types::cmat > &A, const std::string &mat\_file, const std::string &var\_name, const std::string &mode)
- `template<typename Derived >`  
Derived `rand` (std::size\_t rows, std::size\_t cols, double a=0, double b=1)
- `template<>`  
types::dmat `rand` (std::size\_t rows, std::size\_t cols, double a, double b)
- `template<>`  
types::cmat `rand` (std::size\_t rows, std::size\_t cols, double a, double b)
- double `rand` (double a=0, double b=1)
- long long `randint` (long long a, long long b)
- `template<typename Derived >`  
Derived `randn` (std::size\_t rows, std::size\_t cols, double mean=0, double sigma=1)
- `template<>`  
types::dmat `randn` (std::size\_t rows, std::size\_t cols, double mean, double sigma)
- `template<>`  
types::cmat `randn` (std::size\_t rows, std::size\_t cols, double mean, double sigma)
- double `randn` (double mean=0, double sigma=1)
- types::cmat `randU` (std::size\_t D)
- types::cmat `randV` (std::size\_t Din, std::size\_t Dout)
- std::vector< types::cmat > `randkraus` (std::size\_t n, std::size\_t D)
- types::cmat `randH` (std::size\_t D)
- types::ket `randket` (std::size\_t D)
- types::cmat `randrho` (std::size\_t D)
- std::vector< std::size\_t > `randperm` (std::size\_t n)

## Variables

- `RandomDevices` & `rdevs` = `RandomDevices::get_instance()`  
*qpp::RandomDevices Singleton*
- const `Gates` & `gt` = `Gates::get_instance()`  
*qpp::Gates const Singleton*
- const `States` & `st` = `States::get_instance()`  
*qpp::States const Singleton*

## 5.1.1 Function Documentation

### 5.1.1.1 `template<typename Derived > types::cmat qpp::absm ( const Eigen::MatrixBase< Derived > & A )`

Matrix absolut value.

#### Parameters

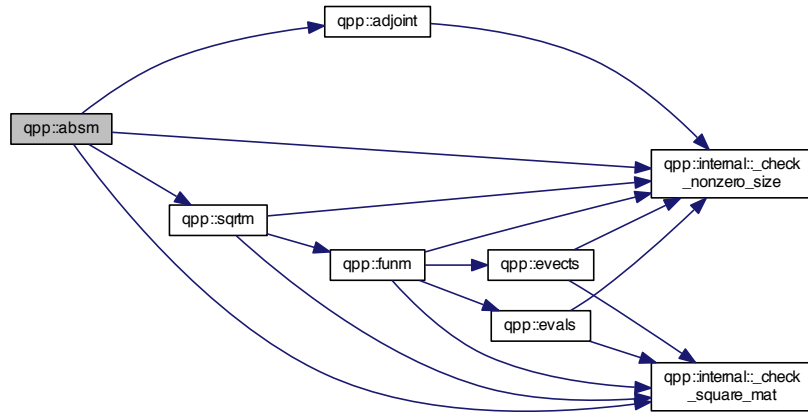
---

A	Eigen expression
---	------------------

**Returns**

Matrix absolut value of  $A$ , as a dynamic matrix over the complex field

Here is the call graph for this function:



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

Adjoint.

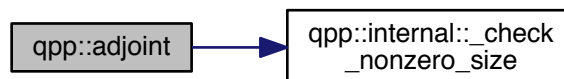
**Parameters**

A	Eigen expression
---	------------------

**Returns**

Adjoint (Hermitian conjugate) of  $A$ , as a dynamic matrix over the same scalar field

Here is the call graph for this function:



5.1.1.3 `template<typename Derived1 , typename Derived2 > types::DynMat<typename Derived1::Scalar> qpp::anticomm ( const Eigen::MatrixBase< Derived1 > & A, const Eigen::MatrixBase< Derived2 > & B )`

Anti-commutator.

Anti-commutator  $\{A, B\} = AB + BA$

Both  $A$  and  $B$  must be Eigen expressions over the same scalar field

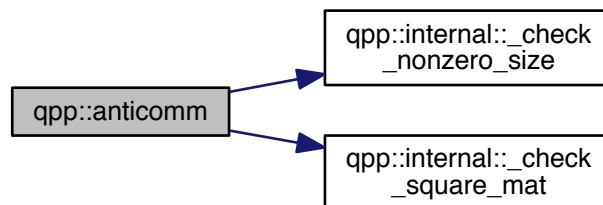
#### Parameters

$A$	Eigen expression
$B$	Eigen expression

#### Returns

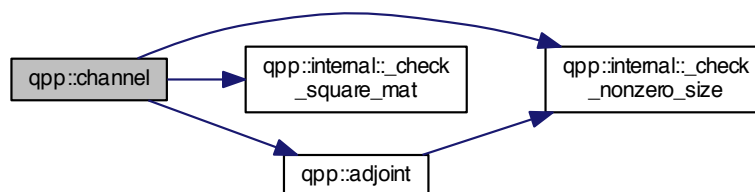
Anti-commutator  $AB + BA$ , as a dynamic matrix over the same scalar field

Here is the call graph for this function:



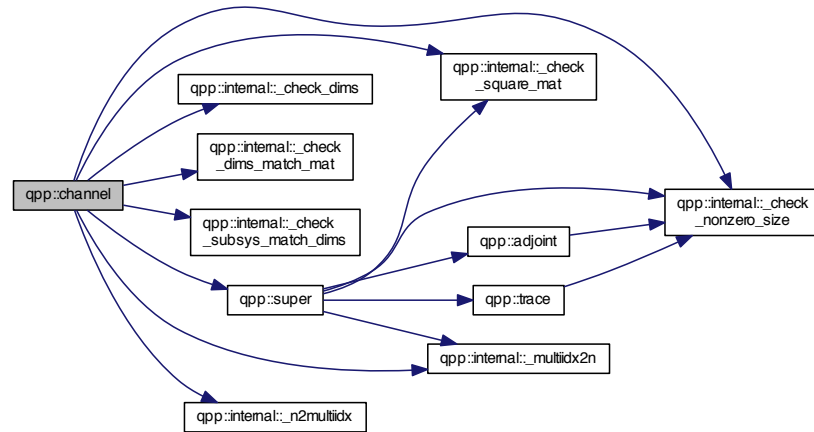
5.1.1.4 `template<typename Derived> types::cmat qpp::channel ( const Eigen::MatrixBase< Derived> & rho, const std::vector< types::cmat> & Ks )`

Here is the call graph for this function:



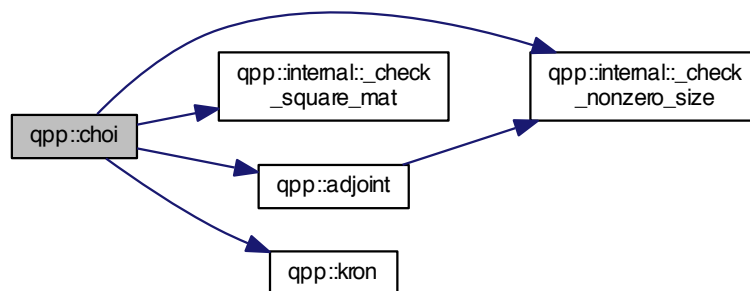
5.1.1.5 `template<typename Derived > types::cmat qpp::channel ( const Eigen::MatrixBase< Derived > & rho, const std::vector< types::cmat > & Ks, const std::vector< std::size_t > & subsys, const std::vector< std::size_t > & dims )`

Here is the call graph for this function:



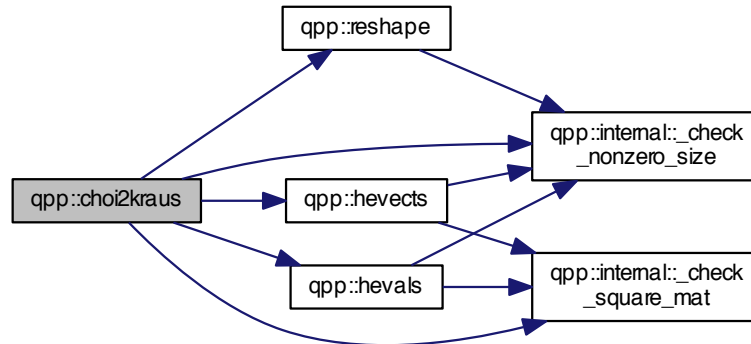
5.1.1.6 `types::cmat qpp::choi ( const std::vector< types::cmat > & Ks )`

Here is the call graph for this function:



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

Here is the call graph for this function:



#### 5.1.1.8 `template<typename Derived1 , typename Derived2 > types::DynMat<typename Derived1::Scalar> qpp::comm ( const Eigen::MatrixBase< Derived1 > & A, const Eigen::MatrixBase< Derived2 > & B )`

Commutator.

Commutator  $[A, B] = AB - BA$

Both  $A$  and  $B$  must be Eigen expressions over the same scalar field

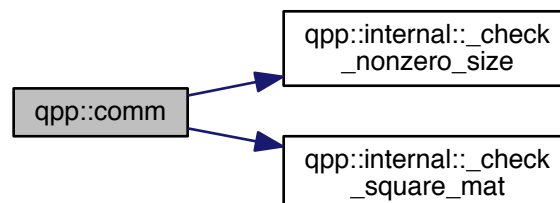
Parameters

$A$	Eigen expression
$B$	Eigen expression

Returns

Commutator  $AB - BA$ , as a dynamic matrix over the same scalar field

Here is the call graph for this function:



5.1.1.9 `std::vector<std::size_t> qpp::compperm ( const std::vector< std::size_t > & perm, const std::vector< std::size_t > & sigma )`

Compose permutations.

## Parameters

<i>perm</i>	Permutation
<i>sigma</i>	Permutation

## Returns

Composition of the permutations  $perm \circ sigma = perm(sigma)$

Here is the call graph for this function:



5.1.1.10 `template<typename Derived> types::DynMat<typename Derived::Scalar> qpp::conjugate ( const Eigen::MatrixBase< Derived> & A )`

Complex conjugate.

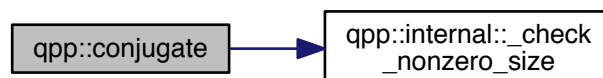
## Parameters

<i>A</i>	Eigen expression
----------	------------------

## Returns

Complex conjugate of *A*, as a dynamic matrix over the same scalar field

Here is the call graph for this function:



5.1.1.11 `template<typename Derived> types::cmat qpp::cosm ( const Eigen::MatrixBase< Derived> & A )`

Matrix cos.



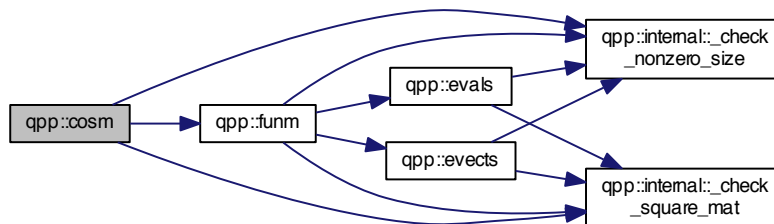
## Parameters

$A$	Eigen expression
-----	------------------

## Returns

Matrix cosine of  $A$ , as a dynamic matrix over the complex field

Here is the call graph for this function:



**5.1.1.12** `template<typename OutputScalar , typename Derived > types::DynMat<OutputScalar> qpp::cwise ( const Eigen::MatrixBase< Derived > & A, OutputScalar*)(const typename Derived::Scalar &) f )`

Functor.

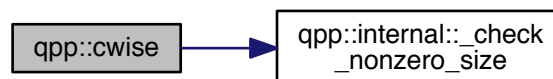
## Parameters

$A$	Eigen expression
$f$	Pointer-to-function from scalars of $A$ to <i>OutputScalar</i>

## Returns

Component-wise  $f(A)$ , as a dynamic matrix over the *OutputScalar* scalar field

Here is the call graph for this function:



**5.1.1.13** `template<typename Derived > Derived::Scalar qpp::det ( const Eigen::MatrixBase< Derived > & A )`

Determinant.

## Parameters

$A$	Eigen expression
-----	------------------

## Returns

Determinant of  $A$ , as a dynamic matrix over the same scalar field  
 Returns  $\pm\infty$  when the determinant overflows/underflows

Here is the call graph for this function:



5.1.1.14 `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 )`

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

5.1.1.16 `template<typename Derived> void qpp::disp ( const Eigen::MatrixBase< Derived > & A, double chop = ct::chop, std::ostream & os = std::cout )`

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

Here is the call graph for this function:



5.1.1.18 `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:



5.1.1.19 `template<typename T> void qpp::displn ( const T * x, const std::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:



5.1.1.20 `template<typename Derived> void qpp::displn ( const Eigen::MatrixBase< Derived > & A, double chop = ct::chop, std::ostream & os = std::cout )`

Here is the call graph for this function:



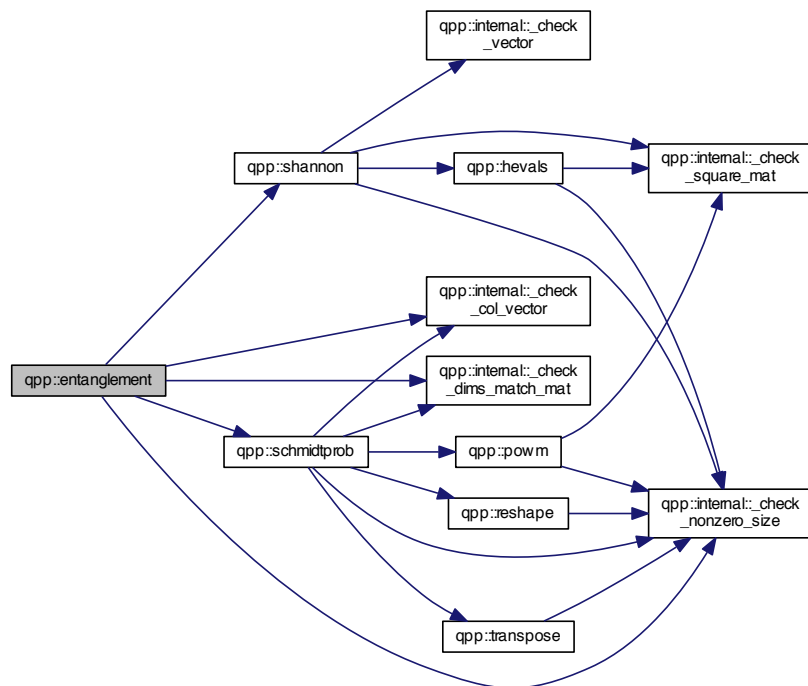
5.1.1.21 `void qpp::displn ( const types::cplx c, double chop = ct:::chop, std::ostream & os = std:::cout )`

Here is the call graph for this function:



5.1.1.22 `template<typename Derived> double qpp::entanglement ( const Eigen::MatrixBase< Derived> & A, const std::vector< std::size_t> & dims )`

Here is the call graph for this function:



5.1.1.23 `template<typename Derived> types::cmat qpp::evals ( const Eigen::MatrixBase< Derived> & A )`

Eigenvalues.

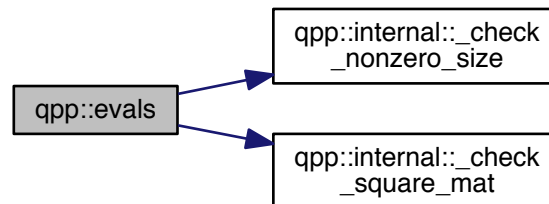
## Parameters

$A$	Eigen expression
-----	------------------

## Returns

Eigenvalues of  $A$ , as a diagonal dynamic matrix over the complex field, with eigenvalues on the diagonal

Here is the call graph for this function:



5.1.1.24 `template<typename Derived> types::cmat qpp::evecs ( const Eigen::MatrixBase< Derived> & A )`

Eigenvectors.

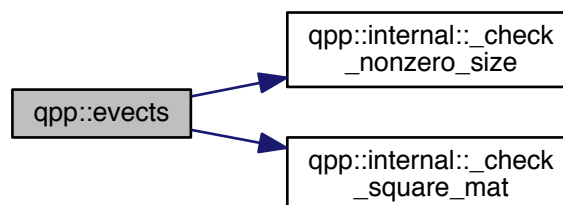
## Parameters

$A$	Eigen expression
-----	------------------

## Returns

Eigenvectors of  $A$ , as columns of a dynamic matrix over the complex field

Here is the call graph for this function:



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

Expand out.

Expand out  $A$  as a matrix in a multi-partite system  
 Faster than using `qpp::kron(I, I, ..., I, A, I, ..., I)`

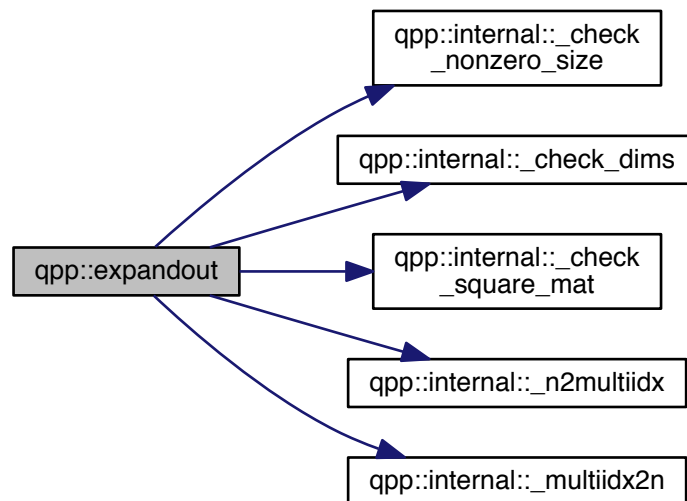
#### Parameters

$A$	Eigen expression
$pos$	Position
$dims$	Dimensions of the multi-partite system

#### Returns

Tensor product  $I \otimes \dots \otimes I \otimes A \otimes I \otimes \dots \otimes I$ , with  $A$  on position  $pos$ , as a dynamic matrix over the same scalar field

Here is the call graph for this function:



5.1.1.26 `template<typename Derived> types::cmat qpp::expm ( const Eigen::MatrixBase< Derived> & A )`

Matrix exponential.

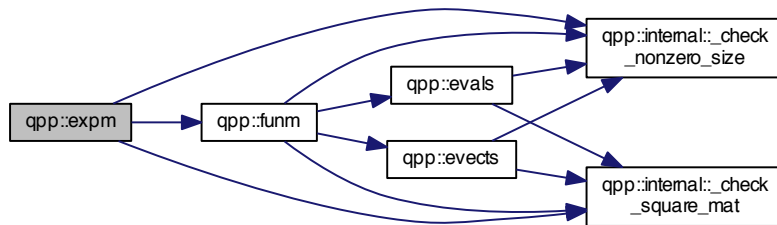
#### Parameters

$A$	Eigen expression
-----	------------------

## Returns

Matrix exponential of  $A$ , as a dynamic matrix over the complex field

Here is the call graph for this function:



5.1.1.27 `template<typename Derived > types::cmat qpp::funm ( const Eigen::MatrixBase< Derived > & A, types::cplx*)(const types::cplx &) f )`

Functional calculus  $f(A)$

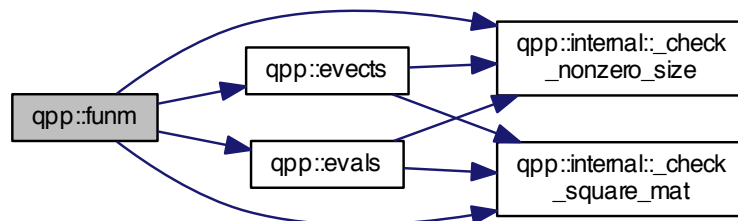
## Parameters

$A$	Eigen expression
$f$	Pointer-to-function from complex to complex

## Returns

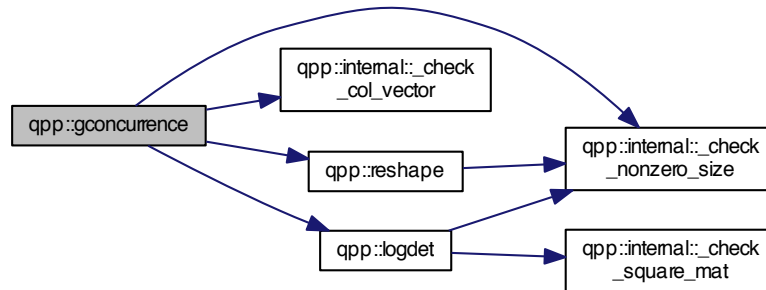
$f(A)$ , as a dynamic matrix over the complex field

Here is the call graph for this function:



5.1.1.28 `template<typename Derived> double qpp::gconcurrency ( const Eigen::MatrixBase< Derived> & A )`

Here is the call graph for this function:



5.1.1.29 `template<typename Derived> types::DynMat<typename Derived::Scalar> qpp::grams ( const std::vector< Derived> & Vs )`

Gram-Schmidt orthogonalization (std::vector overload)

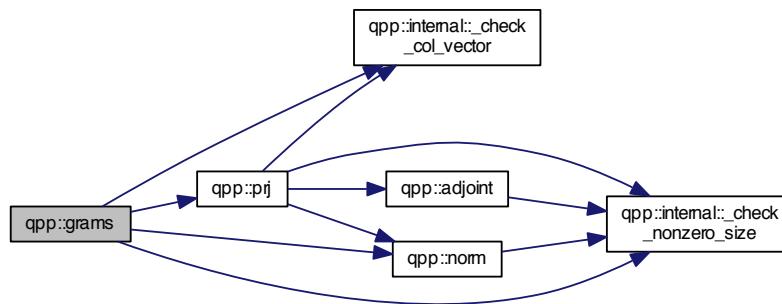
Parameters

<code>Vs</code>	std::vector of Eigen expressions as column vectors
-----------------	--

Returns

Gram-Schmidt vectors of `Vs` as columns of a dynamic matrix over the same scalar field

Here is the call graph for this function:



5.1.1.30 `template<typename Derived> types::DynMat<typename Derived::Scalar> qpp::grams ( const std::initializer_list< Derived> & Vs )`

Gram-Schmidt orthogonalization (std::initializer\_list overload)



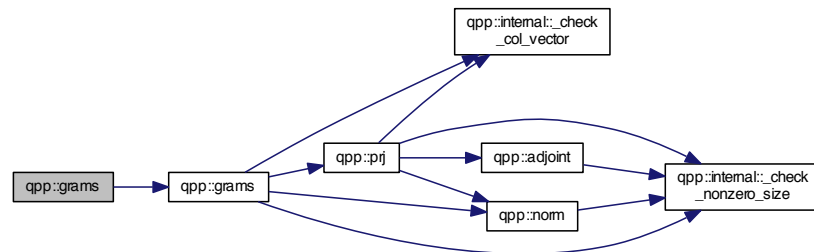
## Parameters

<i>Vs</i>	<code>std::initializer_list</code> of Eigen expressions as column vectors
-----------	---

## Returns

Gram-Schmidt vectors of *Vs* as columns of a dynamic matrix over the same scalar field

Here is the call graph for this function:



**5.1.1.31** `template<typename Derived> types::DynMat<typename Derived::Scalar> qpp::grams ( const Eigen::MatrixBase< Derived> & A )`

Gram-Schmidt orthogonalization (Eigen expression (matrix) overload)

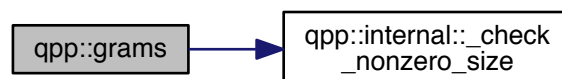
## Parameters

<i>A</i>	Eigen expression, the input vectors are the columns of <i>A</i>
----------	---

## Returns

Gram-Schmidt vectors of the columns of *A*, as columns of a dynamic matrix over the same scalar field

Here is the call graph for this function:



**5.1.1.32** `template<typename Derived> types::dmat qpp::hevals ( const Eigen::MatrixBase< Derived> & A )`

Hermitian eigenvalues.

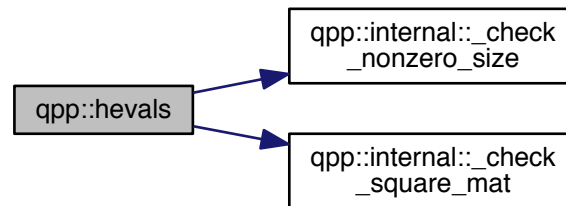
## Parameters

$A$	Eigen expression
-----	------------------

## Returns

Eigenvalues of Hermitian  $A$ , as a diagonal dynamic matrix over the real field, with eigenvalues on the diagonal

Here is the call graph for this function:



**5.1.1.33** `template<typename Derived> types::cmat qpp::hevects ( const Eigen::MatrixBase< Derived> & A )`

Hermitian eigenvectors.

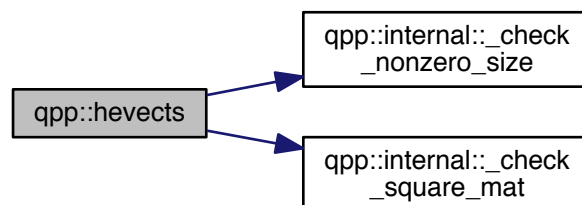
## Parameters

$A$	Eigen expression
-----	------------------

## Returns

Eigenvectors of Hermitian  $A$ , as columns of a dynamic matrix over the complex field

Here is the call graph for this function:



**5.1.1.34** `template<typename Derived> types::DynMat<typename Derived::Scalar> qpp::inverse ( const Eigen::MatrixBase< Derived> & A )`

Inverse.

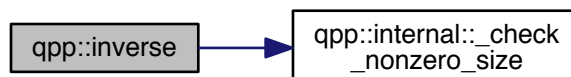
## Parameters

$A$	Eigen expression
-----	------------------

## Returns

Inverse of  $A$ , as a dynamic matrix over the same scalar field

Here is the call graph for this function:



5.1.1.35 `std::vector<std::size_t> qpp::invperm ( const std::vector< std::size_t > & perm )`

Inverse permutation.

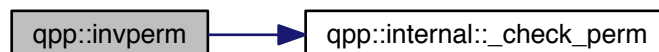
## Parameters

<i>perm</i>	Permutation
-------------	-------------

## Returns

Inverse of the permutation *perm*

Here is the call graph for this function:



5.1.1.36 `template<typename T> types::DynMat<typename T::Scalar> qpp::kron ( const T & head )`

Kronecker product (variadic overload)

Used to stop the recursion for the variadic template version of [qpp::kron\(\)](#)

## Parameters

<i>head</i>	Eigen expression
-------------	------------------

**Returns**

Its argument *head*

**5.1.1.37** `template<typename T , typename... Args> types::DynMat<typename T::Scalar> qpp::kron ( const T & head, const Args &... tail )`

Kronecker product (variadic overload)

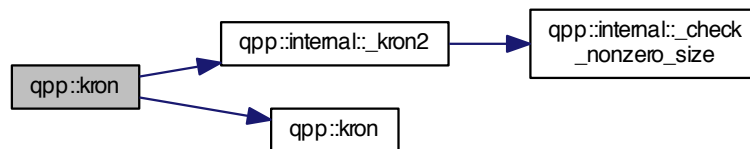
**Parameters**

<i>head</i>	Eigen expression
<i>tail</i>	Variadic Eigen expression (zero or more parameters)

**Returns**

Kronecker product of all input parameters, evaluated from left to right, as a dynamic matrix over the same scalar field

Here is the call graph for this function:



**5.1.1.38** `template<typename Derived > types::DynMat<typename Derived::Scalar> qpp::kron ( const std::vector< Derived > & As )`

Kronecker product (std::vector overload)

**Parameters**

<i>As</i>	std::vector of Eigen expressions
-----------	----------------------------------

**Returns**

Kronecker product of all elements in  $As$ , evaluated from left to right, as a dynamic matrix over the same scalar field

Here is the call graph for this function:



**5.1.1.39** `template<typename Derived> types::DynMat<typename Derived::Scalar> qpp::kron ( const std::initializer_list<Derived> & As )`

Kronecker product (std::initializer\_list overload)

**Parameters**

$As$	std::initializer_list of Eigen expressions, such as $\{A1, A2, \dots, Ak\}$
------	---

**Returns**

Kronecker product of all elements in  $As$ , evaluated from left to right, as a dynamic matrix over the same scalar field

Here is the call graph for this function:



**5.1.1.40** `template<typename Derived> types::DynMat<typename Derived::Scalar> qpp::kronpow ( const Eigen::MatrixBase<Derived> & A, std::size_t n )`

Kronecker power.

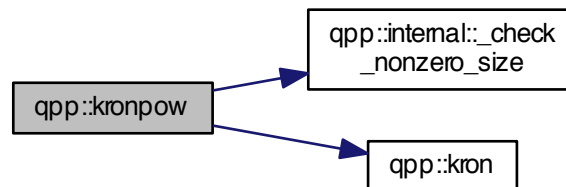
**Parameters**

$A$	Eigen expression
$n$	Non-negative integer

**Returns**

Kronecker product of  $A$  with itself  $n$  times  $A^{\otimes n}$ , as a dynamic matrix over the same scalar field

Here is the call graph for this function:



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

5.1.1.42 `template<typename Derived > Derived qpp::loadMATLABmatrix ( const std::string & mat_file, const std::string & var_name )`

5.1.1.43 `template<> types::dmat qpp::loadMATLABmatrix ( const std::string & mat_file, const std::string & var_name )`

5.1.1.44 `template<> types::cmat qpp::loadMATLABmatrix ( const std::string & mat_file, const std::string & var_name )`

5.1.1.45 `template<typename Derived > Derived::Scalar qpp::logdet ( const Eigen::MatrixBase< Derived > & A )`

Logarithm of the determinant.

Especially useful when the determinant overflows/underflows

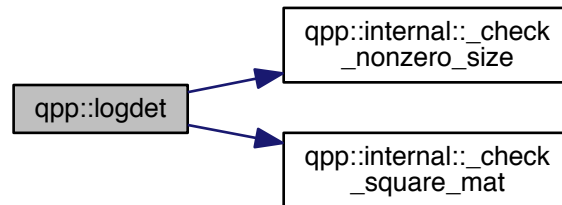
**Parameters**

$A$	Eigen expression
-----	------------------

## Returns

Logarithm of the determinant of  $A$ , as a dynamic matrix over the same scalar field

Here is the call graph for this function:



5.1.1.46 `template<typename Derived> types::cmat qpp::logm ( const Eigen::MatrixBase< Derived > & A )`

Matrix logarithm.

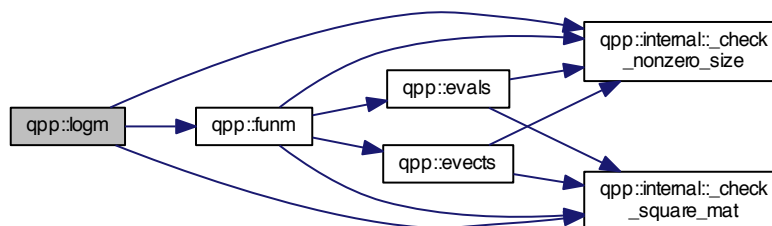
## Parameters

$A$	Eigen expression
-----	------------------

## Returns

Matrix logarithm of  $A$ , as a dynamic matrix over the complex field

Here is the call graph for this function:



5.1.1.47 `types::ket qpp::mket ( const std::vector< std::size_t > & mask )`

Multi-partite qubit ket.

Constructs the multi-partite qubit ket  $|\text{mask}\rangle$ , where  $\text{mask}$  is a `std::vector` of 0's and 1's

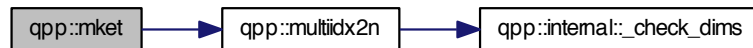
## Parameters

<i>mask</i>	std::vector of 0's and 1's
-------------	----------------------------

## Returns

Multi-partite qubit state vector, as a dynamic column vector over the complex field

Here is the call graph for this function:



#### 5.1.1.48 types::ket qpp::mket ( const std::vector< std::size\_t > & *mask*, const std::vector< std::size\_t > & *dims* )

Multi-partite qudit ket (different dimensions overload)

Constructs the multi-partite qudit ket  $|\text{mask}\rangle$ , where *mask* is a std::vector of non-negative integers  
Each element in *mask* has to be smaller than the corresponding element in *dims*

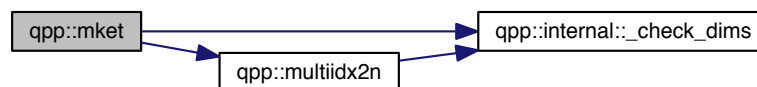
## Parameters

<i>mask</i>	std::vector of non-negative integers
-------------	--------------------------------------

## Returns

Multi-partite qudit state vector, as a dynamic column vector over the complex field

Here is the call graph for this function:



#### 5.1.1.49 types::ket qpp::mket ( const std::vector< std::size\_t > & *mask*, std::size\_t *d* )

Multi-partite qudit ket (same dimensions overload)

Constructs the multi-partite qudit ket  $|\text{mask}\rangle$  in a multi-partite system, all subsystem having equal dimension *d*  
*mask* is a std::vector of non-negative integers, and each element in *mask* has to be strictly smaller than *d*



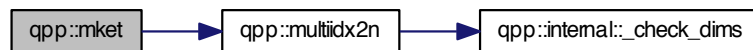
## Parameters

<i>mask</i>	std::vector of non-negative integers
<i>d</i>	Subsystems' dimension

## Returns

Multi-partite qudit state vector, as a dynamic column vector over the complex field

Here is the call graph for this function:



5.1.1.50 `std::size_t qpp::multiidx2n ( const std::vector< std::size_t > & midx, const std::vector< std::size_t > & dims )`

Multi-index to non-negative integer index.

Uses standard lexicographical order, i.e. 00...0, 00...1 etc.

## Parameters

<i>midx</i>	Multi-index
<i>dims</i>	Dimensions of the multi-partite system

## Returns

Non-negative integer index

Here is the call graph for this function:



5.1.1.51 `std::vector<std::size_t> qpp::n2multiidx ( std::size_t n, const std::vector< std::size_t > & dims )`

Non-negative integer index to multi-index.

Uses standard lexicographical order, i.e. 00...0, 00...1 etc.

## Parameters

$n$	Non-negative integer index
$dims$	Dimensions of the multi-partite system

## Returns

Multi-index of the same size as  $dims$

Here is the call graph for this function:



#### 5.1.1.52 `template<typename Derived> double qpp::norm ( const Eigen::MatrixBase< Derived > & A )`

Trace norm.

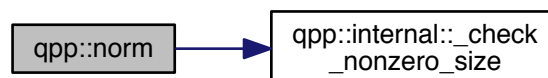
## Parameters

$A$	Eigen expression
-----	------------------

## Returns

Trace norm (Frobenius norm) of  $A$ , as a real number

Here is the call graph for this function:



#### 5.1.1.53 `constexpr std::complex<double> qpp::operator""_i ( unsigned long long int x )`

User-defined literal for complex  $i = \sqrt{-1}$  (integer overload)

Example:

```
auto z = 4_i; // type of z is std::complex<double>
```

5.1.1.54 `constexpr std::complex<double> qpp::operator""_i ( long double x )`

User-defined literal for complex  $i = \sqrt{-1}$  (real overload)

Example:

```
auto z = 4.5_i; // type of z is std::complex<double>
```

5.1.1.55 `template<typename Derived > types::DynMat<typename Derived::Scalar> qpp::powm ( const Eigen::MatrixBase< Derived > & A, std::size_t n )`

Matrix power.

Explicitly multiplies the matrix  $A$  with itself  $n$  times

By convention  $A^0 = I$

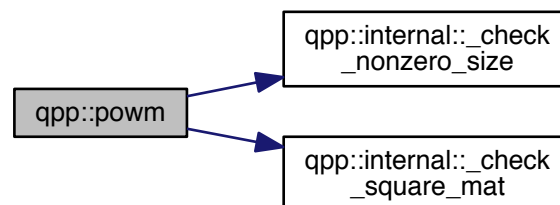
Parameters

$A$	Eigen expression
$n$	Non-negative integer

Returns

Matrix power  $A^n$ , as a dynamic matrix over the same scalar field

Here is the call graph for this function:

5.1.1.56 `template<typename Derived > types::DynMat<typename Derived::Scalar> qpp::prj ( const Eigen::MatrixBase< Derived > & V )`

Projector.

Normalized projector onto state vector

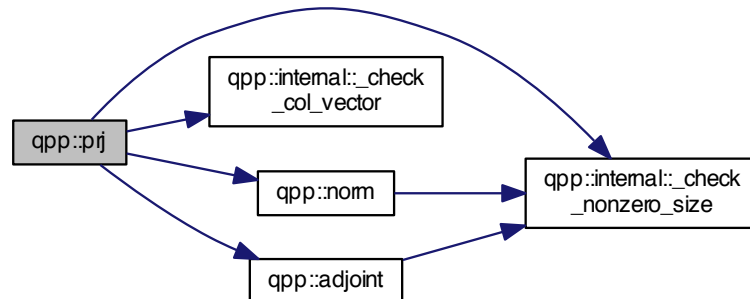
Parameters

$V$	Eigen expression
-----	------------------

Returns

Projector onto the state vector  $V$ , or the matrix *Zero* if  $V$  has norm zero (i.e. smaller than [qpp::ct::eps](#)), as a dynamic matrix over the same scalar field

Here is the call graph for this function:



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

Partial trace.

Partial trace of the multi-partite density matrix over a list of subsystems

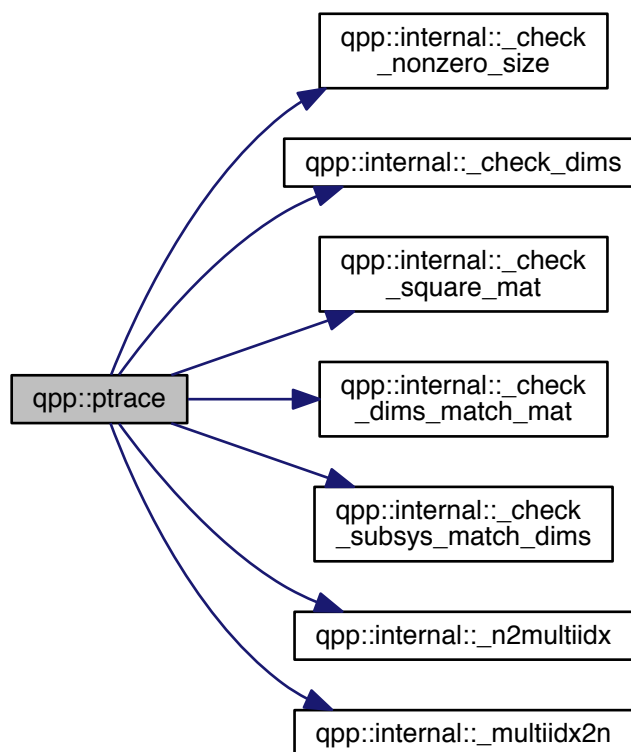
Parameters

<i>A</i>	Eigen expression
<i>subsys</i>	Subsystems' indexes
<i>dims</i>	Dimensions of the multi-partite system

Returns

Partial trace  $Tr_{subsys}(\cdot)$  over the subsystems *subsys* in a multi-partite system, as a dynamic matrix over the same scalar field

Here is the call graph for this function:



5.1.1.58 `template<typename Derived> types::DynMat<typename Derived::Scalar> qpp::ptrace1 ( const Eigen::MatrixBase< Derived> & A, const std::vector< std::size_t> & dims )`

Partial trace.

Partial trace of density matrix over the first subsystem in a bi-partite system

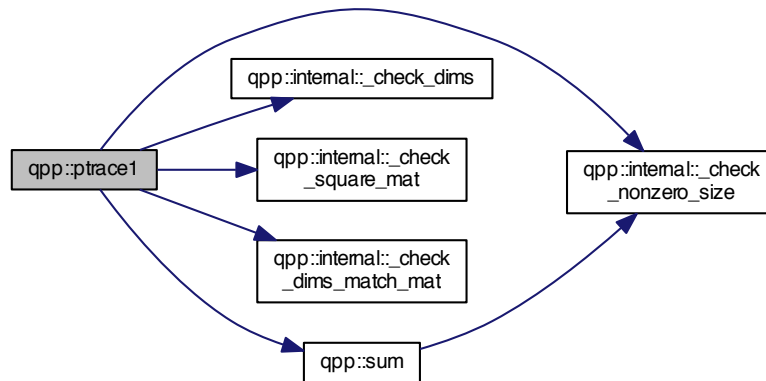
Parameters

<i>A</i>	Eigen expression
<i>dims</i>	Dimensions of bi-partite system (must be a <code>std::vector</code> with 2 elements)

**Returns**

Partial trace  $Tr_A(\cdot)$  over the first subsystem  $A$  in a bi-partite system  $A \otimes B$ , as a dynamic matrix over the same scalar field

Here is the call graph for this function:



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

Partial trace.

Partial trace of density matrix over the second subsystem in a bi-partite system

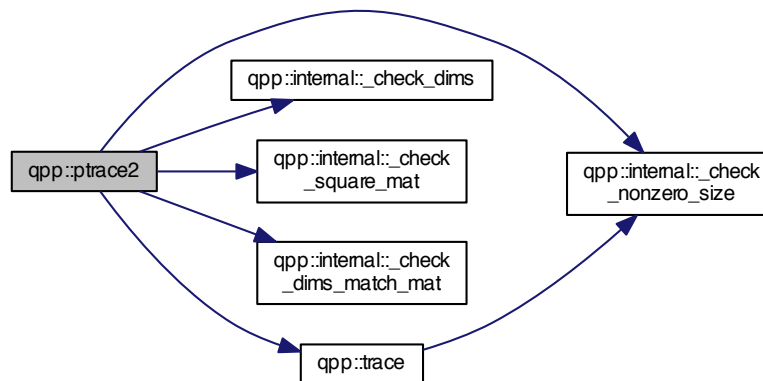
**Parameters**

<i>A</i>	Eigen expression
<i>dims</i>	Dimensions of bi-partite system (must be a <code>std::vector</code> with 2 elements)

## Returns

Partial trace  $Tr_B(\cdot)$  over the second subsystem  $B$  in a bi-partite system  $A \otimes B$ , as a dynamic matrix over the same scalar field

Here is the call graph for this function:



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

Partial transpose.

Partial transpose of the multi-partite density matrix over a list of subsystems

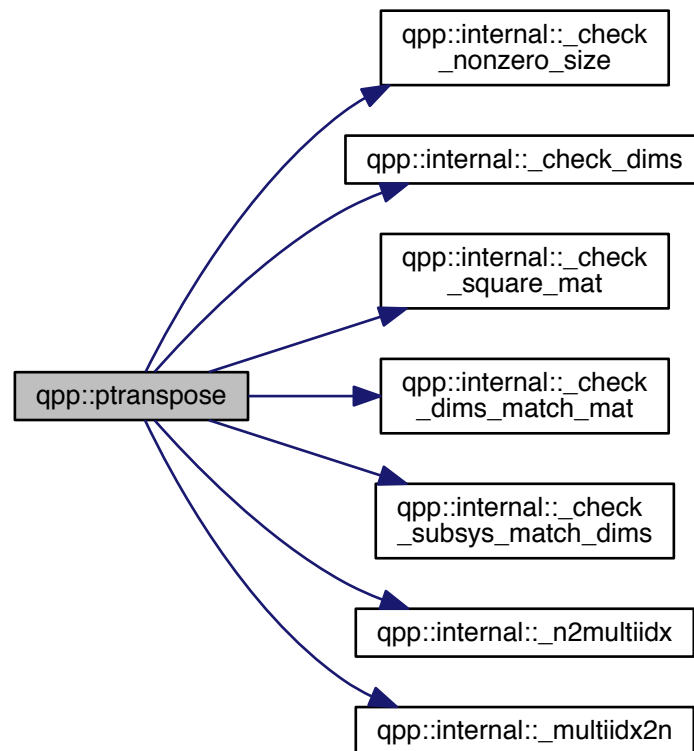
## Parameters

<i>A</i>	Eigen expression
<i>subsys</i>	Subsystems' indexes
<i>dims</i>	Dimensions of the multi-partite system

**Returns**

Partial transpose  $(\cdot)^{T_{\text{subsys}}}$  over the subsystems *subsys* in a multi-partite system, as a dynamic matrix over the same scalar field

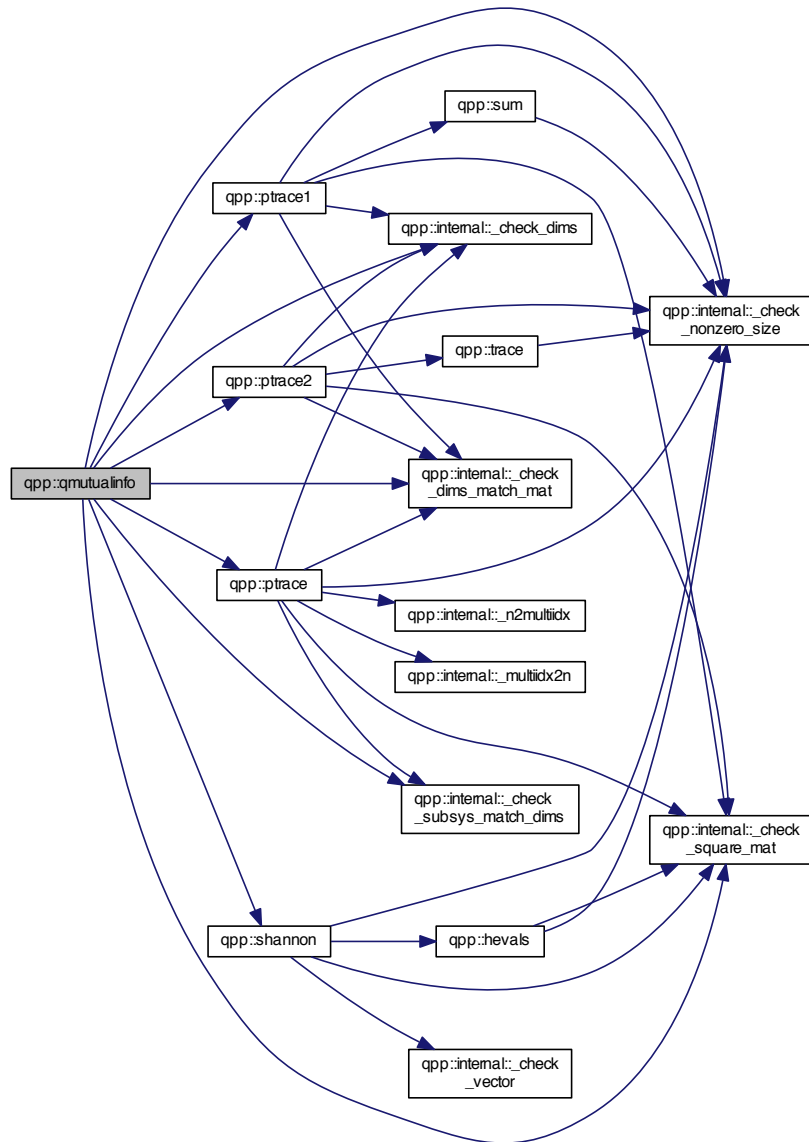
Here is the call graph for this function:





5.1.1.61 `template<typename Derived> double qpp::qmutualinfo ( const Eigen::MatrixBase< Derived> & A, const std::vector< std::size_t> & subsys, const std::vector< std::size_t> & dims )`

Here is the call graph for this function:



5.1.1.62 `template<typename Derived> Derived qpp::rand ( std::size_t rows, std::size_t cols, double a = 0, double b = 1 )`

5.1.1.63 `template<> types::dmat qpp::rand ( std::size_t rows, std::size_t cols, double a, double b )`

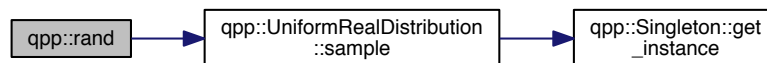
5.1.1.64 `template<> types::cmat qpp::rand ( std::size_t rows, std::size_t cols, double a, double b )`

Here is the call graph for this function:



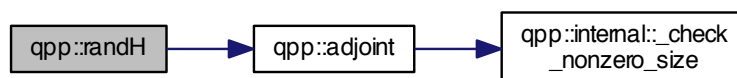
5.1.1.65 `double qpp::rand ( double a = 0, double b = 1 )`

Here is the call graph for this function:



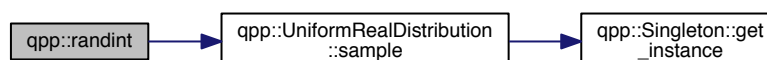
5.1.1.66 `types::cmat qpp::randH ( std::size_t D )`

Here is the call graph for this function:



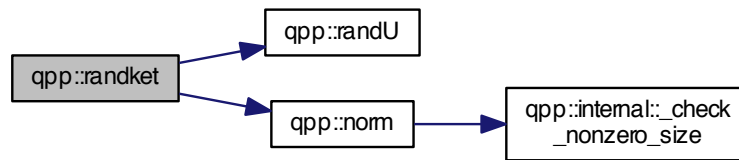
5.1.1.67 `long long qpp::randint ( long long a, long long b )`

Here is the call graph for this function:

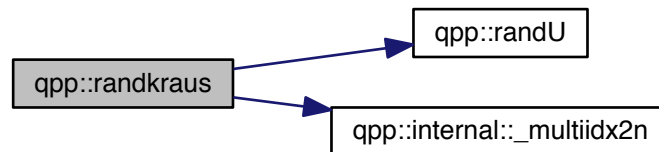


5.1.1.68 `types::ket qpp::randket ( std::size_t D )`

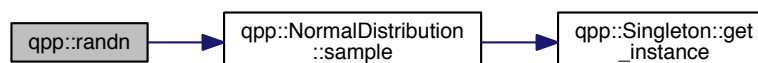
Here is the call graph for this function:

5.1.1.69 `std::vector<types::cmat> qpp::randkraus ( std::size_t n, std::size_t D )`

Here is the call graph for this function:

5.1.1.70 `template<typename Derived> Derived qpp::randn ( std::size_t rows, std::size_t cols, double mean = 0, double sigma = 1 )`5.1.1.71 `template<> types::dmat qpp::randn ( std::size_t rows, std::size_t cols, double mean, double sigma )`

Here is the call graph for this function:



5.1.1.72 `template<> types::cmat qpp::randn ( std::size_t rows, std::size_t cols, double mean, double sigma )`

Here is the call graph for this function:



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

Here is the call graph for this function:



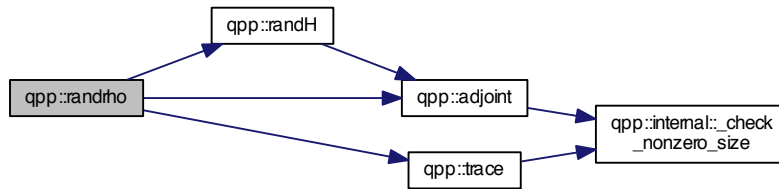
5.1.1.74 `std::vector<std::size_t> qpp::randperm ( std::size_t n )`

Here is the call graph for this function:



5.1.1.75 `types::cmat qpp::randrho ( std::size_t D )`

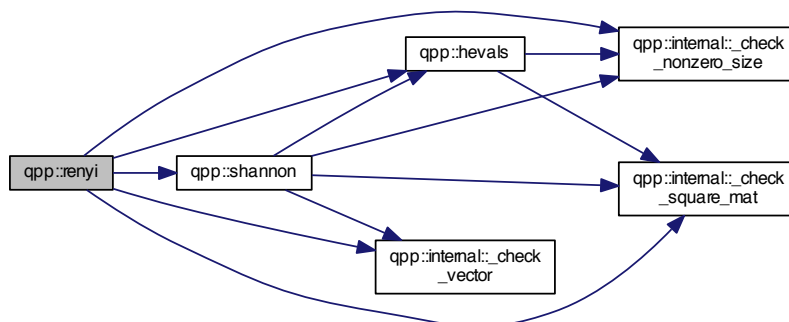
Here is the call graph for this function:

5.1.1.76 `types::cmat qpp::randU ( std::size_t D )`5.1.1.77 `types::cmat qpp::randV ( std::size_t Din, std::size_t Dout )`

Here is the call graph for this function:

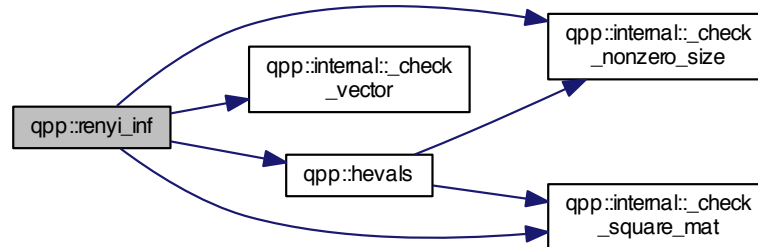
5.1.1.78 `template<typename Derived> double qpp::renyi ( const double alpha, const Eigen::MatrixBase< Derived > & A )`

Here is the call graph for this function:



5.1.1.79 `template<typename Derived> double qpp::renyi_inf ( const Eigen::MatrixBase< Derived> & A )`

Here is the call graph for this function:



5.1.1.80 `template<typename Derived> types::DynMat<typename Derived::Scalar> qpp::reshape ( const Eigen::MatrixBase< Derived> & A, std::size_t rows, std::size_t cols )`

Reshape.

Uses column-major order when reshaping (same as MATLAB)

Parameters

<i>A</i>	Eigen expression
<i>rows</i>	Number of rows of the reshaped matrix
<i>cols</i>	Number of columns of the reshaped matrix

Returns

Reshaped matrix with *rows* rows and *cols* columns, as a dynamic matrix over the same scalar field

Here is the call graph for this function:

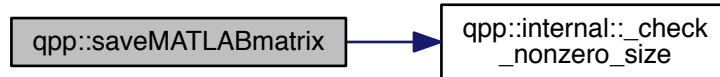


5.1.1.81 `template<typename Derived> void qpp::save ( const Eigen::MatrixBase< Derived> & A, const std::string & fname )`

5.1.1.82 `template<typename Derived> void qpp::saveMATLABmatrix ( const Eigen::MatrixBase< Derived> & A, const std::string & mat_file, const std::string & var_name, const std::string & mode )`

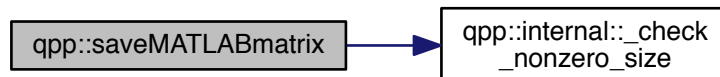
5.1.1.83 `template<> void qpp::saveMATLABmatrix ( const Eigen::MatrixBase< types::dmat > & A, const std::string & mat_file, const std::string & var_name, const std::string & mode )`

Here is the call graph for this function:



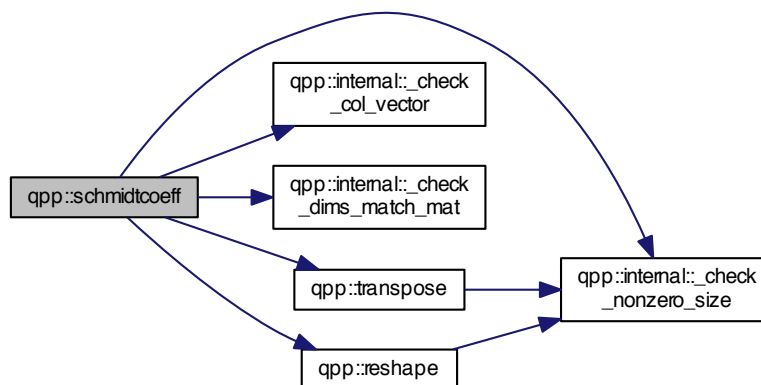
5.1.1.84 `template<> void qpp::saveMATLABmatrix ( const Eigen::MatrixBase< typename types::cmat > & A, const std::string & mat_file, const std::string & var_name, const std::string & mode )`

Here is the call graph for this function:



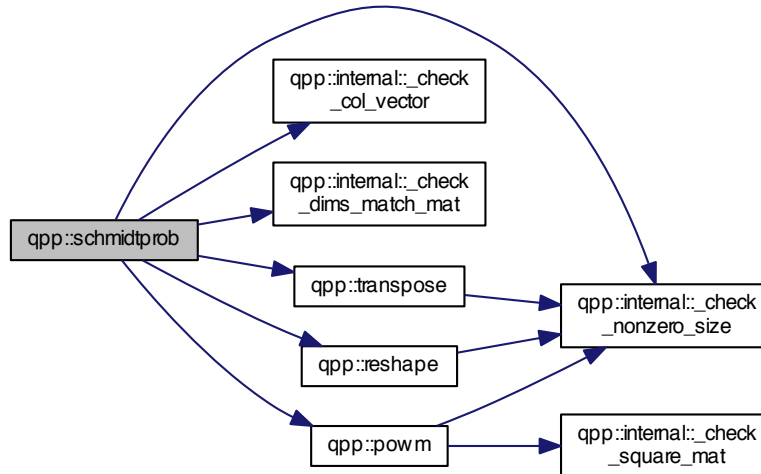
5.1.1.85 `template<typename Derived> types::cmat qpp::schmidtcoeff ( const Eigen::MatrixBase< Derived > & A, const std::vector< std::size_t > & dims )`

Here is the call graph for this function:



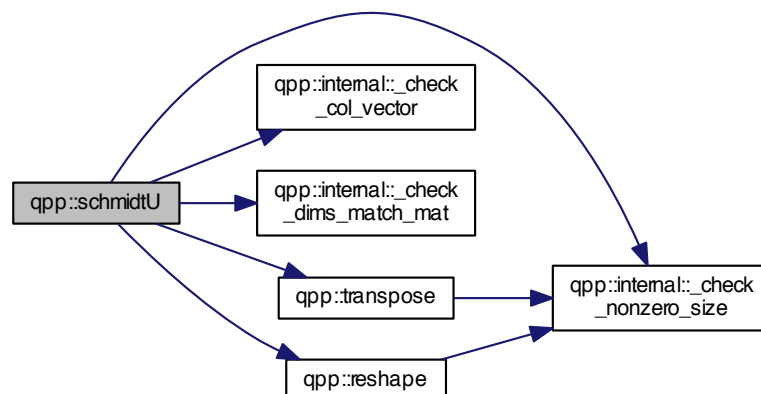
5.1.1.86 `template<typename Derived> types::cmat qpp::schmidtprob ( const Eigen::MatrixBase< Derived> & A, const std::vector< std::size_t> & dims )`

Here is the call graph for this function:



5.1.1.87 `template<typename Derived> types::cmat qpp::schmidtU ( const Eigen::MatrixBase< Derived> & A, const std::vector< std::size_t> & dims )`

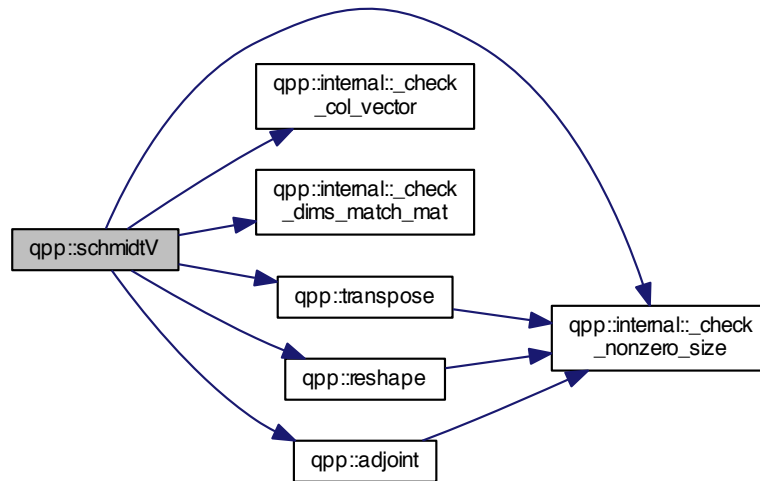
Here is the call graph for this function:





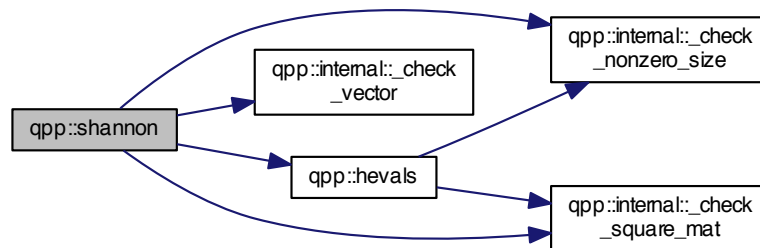
5.1.1.88 `template<typename Derived> types::cmat qpp::schmidtV ( const Eigen::MatrixBase< Derived> & A, const std::vector< std::size_t> & dims )`

Here is the call graph for this function:



5.1.1.89 `template<typename Derived> double qpp::shannon ( const Eigen::MatrixBase< Derived> & A )`

Here is the call graph for this function:



5.1.1.90 `template<typename Derived> types::cmat qpp::sinm ( const Eigen::MatrixBase< Derived> & A )`

Matrix sin.

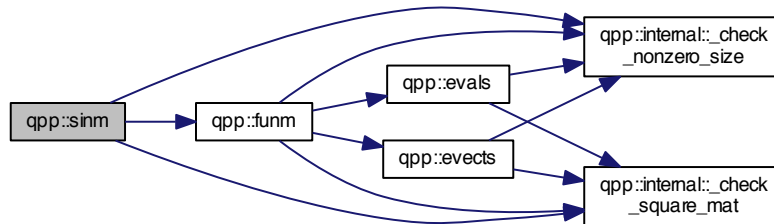
## Parameters

$A$	Eigen expression
-----	------------------

## Returns

Matrix sine of  $A$ , as a dynamic matrix over the complex field

Here is the call graph for this function:



5.1.1.91 `template<typename Derived> types::cmat qpp::spectralpowm ( const Eigen::MatrixBase< Derived> & A, const types::cplx z )`

Matrix power.

Uses the spectral decomposition of  $A$  to compute the matrix power  
By convention  $A^0 = I$

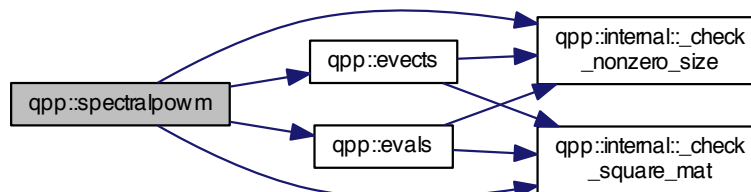
## Parameters

$A$	Eigen expression
$z$	Complex number

## Returns

Matrix power  $A^z$ , as a dynamic matrix over the complex field

Here is the call graph for this function:



5.1.1.92 `template<typename Derived> types::cmat qpp::sqrtm ( const Eigen::MatrixBase< Derived > & A )`

Matrix square root.

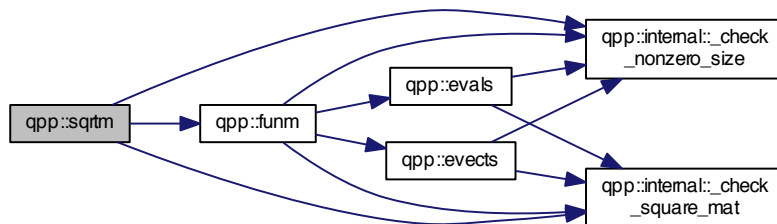
## Parameters

$A$	Eigen expression
-----	------------------

## Returns

Matrix square root of  $A$ , as a dynamic matrix over the complex field

Here is the call graph for this function:



**5.1.1.93** `template<typename Derived > Derived::Scalar qpp::sum ( const Eigen::MatrixBase< Derived > & A )`

Element-wise sum.

## Parameters

$A$	Eigen expression
-----	------------------

## Returns

Element-wise sum of  $A$ , as a dynamic matrix over the same scalar field

Here is the call graph for this function:



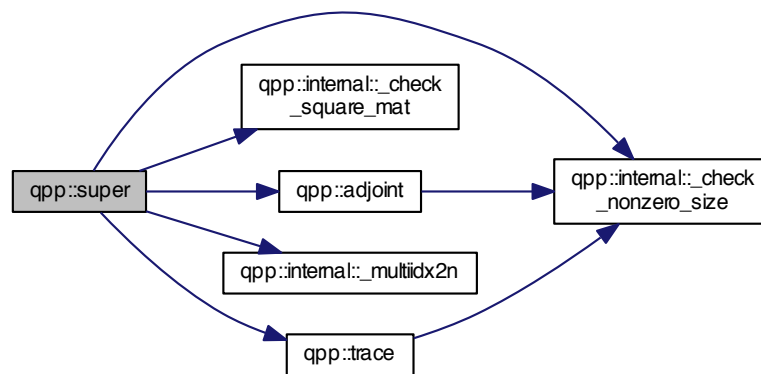
**5.1.1.94** `types::cmat qpp::super ( const std::vector< types::cmat > & Ks )`

## Parameters

<i>Ks</i>	
-----------	--

## Returns

Here is the call graph for this function:



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

System permutation.

Permutes the subsystems in a state vector or density matrix

The qubit *perm*[*i*] is permuted to the location *i*

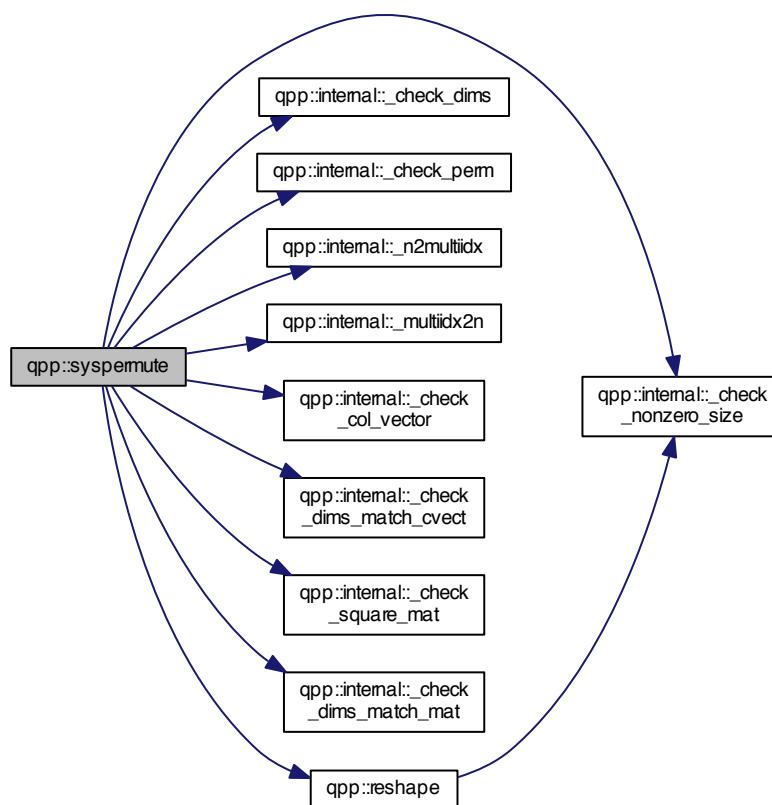
## Parameters

<i>A</i>	Eigen expression
<i>perm</i>	Permutation
<i>dims</i>	Subsystems' dimensions

## Returns

Permuted system, as a dynamic matrix over the same scalar field

Here is the call graph for this function:



5.1.1.96 `template<typename Derived> Derived::Scalar qpp::trace ( const Eigen::MatrixBase< Derived> & A )`

Trace.

## Parameters

A	Eigen expression
---	------------------

**Returns**

Trace of  $A$ , as a dynamic matrix over the same scalar field

Here is the call graph for this function:



**5.1.1.97** `template<typename Derived > types::DynMat<typename Derived::Scalar> qpp::transpose ( const Eigen::MatrixBase< Derived > & A )`

Transpose.

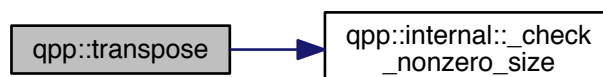
**Parameters**

$A$	Eigen expression
-----	------------------

**Returns**

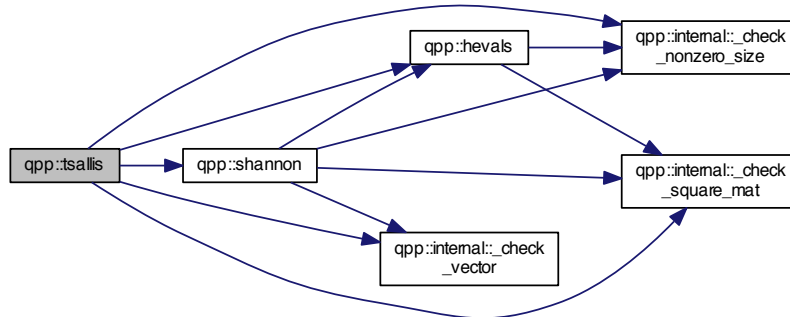
Transpose of  $A$ , as a dynamic matrix over the same scalar field

Here is the call graph for this function:



5.1.1.98 `template<typename Derived> double qpp::tsallis ( const double alpha, const Eigen::MatrixBase< Derived > & A )`

Here is the call graph for this function:



## 5.1.2 Variable Documentation

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

[qpp::Gates](#) const [Singleton](#)

Initializes the gates, see the class [qpp::Gates](#)

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

[qpp::RandomDevices](#) [Singleton](#)

Initializes the random devices, see the class [qpp::RandomDevices](#)

5.1.2.3 `const States& qpp::st = States::get_instance()`

[qpp::States](#) const [Singleton](#)

Initializes the states, see the class [qpp::States](#)

## 5.2 qpp::ct Namespace Reference

### Functions

- `std::complex< double > omega (std::size_t D)`  
*D-th root of unity.*

### Variables

- `constexpr double chop = 1e-10`  
*Used in [qpp::disp\(\)](#) and [qpp::displn\(\)](#) for setting to zero numbers that have their absolute value smaller than [qpp::ct::chop](#).*
- `constexpr double eps = 1e-12`  
*Used to decide whether a number or expression in double precision is zero or not.*



- constexpr std::size\_t `maxn` = 64  
*Maximum number of qubits.*
- constexpr double `pi` = 3.141592653589793238462643383279502884  
 $\pi$
- constexpr double `ee` = 2.718281828459045235360287471352662497  
*Base of natural logarithm,  $e$ .*

## 5.2.1 Function Documentation

### 5.2.1.1 std::complex<double> qpp::ct::omega ( std::size\_t *D* )

D-th root of unity.

Parameters

<i>D</i>	Non-negative integer
----------	----------------------

Returns

D-th root of unity  $\exp(2\pi i/D)$

## 5.2.2 Variable Documentation

### 5.2.2.1 constexpr double qpp::ct::chop = 1e-10

Used in `qpp::disp()` and `qpp::displn()` for setting to zero numbers that have their absolute value smaller than `qpp::ct::chop`.

### 5.2.2.2 constexpr double qpp::ct::ee = 2.718281828459045235360287471352662497

Base of natural logarithm,  $e$ .

### 5.2.2.3 constexpr double qpp::ct::eps = 1e-12

Used to decide whether a number or expression in double precision is zero or not.

Example:

```
if(std::abs(x) < qpp::ct::eps) // x is zero
```

### 5.2.2.4 constexpr std::size\_t qpp::ct::maxn = 64

Maximum number of qubits.

Used internally to statically allocate arrays (for speed reasons)

### 5.2.2.5 constexpr double qpp::ct::pi = 3.141592653589793238462643383279502884

$\pi$

## 5.3 qpp::internal Namespace Reference

### Functions

- void [\\_n2multiidx](#) (std::size\_t n, std::size\_t numdims, const std::size\_t \*dims, std::size\_t \*result)
- std::size\_t [\\_multiidx2n](#) (const std::size\_t \*midx, std::size\_t numdims, const std::size\_t \*dims)
- template<typename Derived >  
bool [\\_check\\_square\\_mat](#) (const Eigen::MatrixBase< Derived > &A)
- template<typename Derived >  
bool [\\_check\\_vector](#) (const Eigen::MatrixBase< Derived > &A)
- template<typename Derived >  
bool [\\_check\\_row\\_vector](#) (const Eigen::MatrixBase< Derived > &A)
- template<typename Derived >  
bool [\\_check\\_col\\_vector](#) (const Eigen::MatrixBase< Derived > &A)
- template<typename T >  
bool [\\_check\\_nonzero\\_size](#) (const T &x)
- bool [\\_check\\_dims](#) (const std::vector< std::size\_t > &dims)
- template<typename Derived >  
bool [\\_check\\_dims\\_match\\_mat](#) (const std::vector< std::size\_t > &dims, const Eigen::MatrixBase< Derived > &A)
- template<typename Derived >  
bool [\\_check\\_dims\\_match\\_cvect](#) (const std::vector< std::size\_t > &dims, const Eigen::MatrixBase< Derived > &V)
- template<typename Derived >  
bool [\\_check\\_dims\\_match\\_rvect](#) (const std::vector< std::size\_t > &dims, const Eigen::MatrixBase< Derived > &V)
- bool [\\_check\\_eq\\_dims](#) (const std::vector< std::size\_t > &dims, std::size\_t dim)
- bool [\\_check\\_subsys\\_match\\_dims](#) (const std::vector< std::size\_t > &subsys, const std::vector< std::size\_t > &dims)
- bool [\\_check\\_perm](#) (const std::vector< std::size\_t > &perm)
- template<typename Derived1 , typename Derived2 >  
[types::DynMat](#)< typename  
Derived1::Scalar > [\\_kron2](#) (const Eigen::MatrixBase< Derived1 > &A, const Eigen::MatrixBase< Derived2 > &B)
- template<typename T >  
void [variadic\\_vector\\_emplace](#) (std::vector< T > &)
- template<typename T , typename First , typename... Args>  
void [variadic\\_vector\\_emplace](#) (std::vector< T > &v, First &&first, Args &&...args)

### 5.3.1 Function Documentation

5.3.1.1 template<typename Derived > bool qpp::internal::\_check\_col\_vector ( const Eigen::MatrixBase< Derived > & A )

5.3.1.2 bool qpp::internal::\_check\_dims ( const std::vector< std::size\_t > & dims )

5.3.1.3 template<typename Derived > bool qpp::internal::\_check\_dims\_match\_cvect ( const std::vector< std::size\_t > & dims, const Eigen::MatrixBase< Derived > & V )

5.3.1.4 template<typename Derived > bool qpp::internal::\_check\_dims\_match\_mat ( const std::vector< std::size\_t > & dims, const Eigen::MatrixBase< Derived > & A )

5.3.1.5 template<typename Derived > bool qpp::internal::\_check\_dims\_match\_rvect ( const std::vector< std::size\_t > & dims, const Eigen::MatrixBase< Derived > & V )

5.3.1.6 bool qpp::internal::\_check\_eq\_dims ( const std::vector< std::size\_t > & dims, std::size\_t dim )

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

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

5.3.1.9 `template<typename Derived > bool qpp::internal::_check_row_vector ( const Eigen::MatrixBase< Derived > & A )`

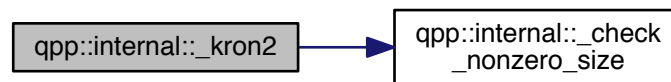
5.3.1.10 `template<typename Derived > bool qpp::internal::_check_square_mat ( const Eigen::MatrixBase< Derived > & A )`

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

5.3.1.12 `template<typename Derived > bool qpp::internal::_check_vector ( const Eigen::MatrixBase< Derived > & A )`

5.3.1.13 `template<typename Derived1 , typename Derived2 > types::DynMat<typename Derived1::Scalar>  
qpp::internal::_kron2 ( const Eigen::MatrixBase< Derived1 > & A, const Eigen::MatrixBase< Derived2 > & B )`

Here is the call graph for this function:



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

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

5.3.1.16 `template<typename T > void qpp::internal::variadic_vector_emplace ( std::vector< T > & )`

5.3.1.17 `template<typename T , typename First , typename... Args> void qpp::internal::variadic_vector_emplace ( std::vector< T > & v, First && first, Args &&... args )`

Here is the call graph for this function:



## 5.4 qpp::types Namespace Reference

## Typedefs

- using `cplx` = `std::complex< double >`  
*Complex number in double precision.*
- using `cmat` = `Eigen::MatrixXcd`  
*Complex (double precision) dynamic Eigen matrix.*
- using `dmat` = `Eigen::MatrixXd`  
*Real (double precision) dynamic Eigen matrix.*
- using `ket` = `Eigen::Matrix< cplx, Eigen::Dynamic, 1 >`  
*Complex (double precision) dynamic Eigen column matrix.*
- using `bra` = `Eigen::Matrix< cplx, 1, Eigen::Dynamic >`  
*Complex (double precision) dynamic Eigen row matrix.*
- template<typename Scalar >  
using `DynMat` = `Eigen::Matrix< Scalar, Eigen::Dynamic, Eigen::Dynamic >`  
*Dynamic Eigen matrix over the field specified by Scalar.*

### 5.4.1 Typedef Documentation

#### 5.4.1.1 using `qpp::types::bra` = `typedef Eigen::Matrix<cplx, 1, Eigen::Dynamic>`

Complex (double precision) dynamic Eigen row matrix.

#### 5.4.1.2 using `qpp::types::cmat` = `typedef Eigen::MatrixXcd`

Complex (double precision) dynamic Eigen matrix.

#### 5.4.1.3 using `qpp::types::cplx` = `typedef std::complex<double>`

Complex number in double precision.

#### 5.4.1.4 using `qpp::types::dmat` = `typedef Eigen::MatrixXd`

Real (double precision) dynamic Eigen matrix.

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

Dynamic Eigen matrix over the field specified by *Scalar*.

Example:

```
auto mat = DynMat<float>(2,3); // type of mat is Eigen::Matrix<float, Eigen::Dynamic, Eigen::Dynamic>
```

#### 5.4.1.6 using `qpp::types::ket` = `typedef Eigen::Matrix<cplx, Eigen::Dynamic, 1>`

Complex (double precision) dynamic Eigen column matrix.

## Chapter 6

# Class Documentation

### 6.1 qpp::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)
- `std::size_t sample` ()
- `std::vector< double > probabilities` () const

#### Protected Attributes

- `std::discrete_distribution`  
< std::size\_t > `_d`

#### 6.1.1 Constructor & Destructor Documentation

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

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

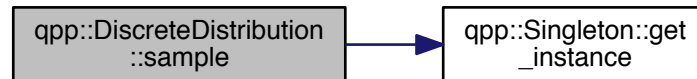
6.1.1.3 `qpp::DiscreteDistribution::DiscreteDistribution ( std::vector< double > weights )` [inline]

#### 6.1.2 Member Function Documentation

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

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

Here is the call graph for this function:



## 6.1.3 Member Data Documentation

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

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

- [include/classes/stat.h](#)

## 6.2 `qpp::DiscreteDistributionAbsSquare` Class Reference

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

### Public Member Functions

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

### Protected Member Functions

- `template<typename InputIterator >`  
`std::vector< double > cplx2weights (InputIterator first, InputIterator last) const`

### Protected Attributes

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

### 6.2.1 Constructor & Destructor Documentation

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

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

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

6.2.1.4 `template<typename Derived > qpp::DiscreteDistributionAbsSquare::DiscreteDistributionAbsSquare ( const Eigen::MatrixBase< Derived > & V ) [inline]`

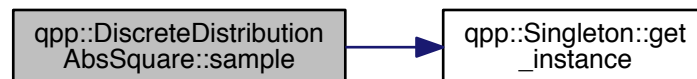
### 6.2.2 Member Function Documentation

6.2.2.1 `template<typename InputIterator > std::vector<double> qpp::DiscreteDistributionAbsSquare::cplx2weights ( InputIterator first, InputIterator last ) const [inline], [protected]`

6.2.2.2 `std::vector<double> qpp::DiscreteDistributionAbsSquare::probabilities ( ) const [inline]`

6.2.2.3 `std::size_t qpp::DiscreteDistributionAbsSquare::sample ( ) [inline]`

Here is the call graph for this function:



### 6.2.3 Member Data Documentation

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

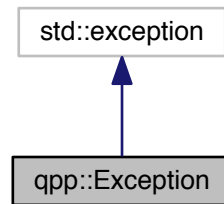
The documentation for this class was generated from the following file:

- [include/classes/stat.h](#)

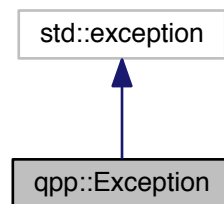
## 6.3 qpp::Exception Class Reference

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

Inheritance diagram for qpp::Exception:



Collaboration diagram for qpp::Exception:



## Public Types

- enum `Type` {  
`Type::UNKNOWN_EXCEPTION = 1`, `Type::ZERO_SIZE`, `Type::MATRIX_NOT_SQUARE`, `Type::MATRIX_NOT_CVECTOR`,  
`Type::MATRIX_NOT_RVECTOR`, `Type::MATRIX_NOT_VECTOR`, `Type::MATRIX_NOT_SQUARE_OR_CVECTOR`,  
`Type::MATRIX_NOT_SQUARE_OR_RVECTOR`,  
`Type::MATRIX_NOT_SQUARE_OR_VECTOR`, `Type::DIMS_INVALID`, `Type::DIMS_NOT_EQUAL`, `Type::DIMS_MISMATCH_MATRIX`,  
`Type::DIMS_MISMATCH_CVECTOR`, `Type::DIMS_MISMATCH_RVECTOR`, `Type::DIMS_MISMATCH_VECTOR`,  
`Type::SUBSYS_MISMATCH_DIMS`,  
`Type::PERM_INVALID`, `Type::NOT_QUBIT_GATE`, `Type::NOT_QUBIT_SUBSYS`, `Type::NOT_BIPARTITE`,  
`Type::OUT_OF_RANGE`, `Type::TYPE_MISMATCH`, `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



## 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***  
***MATRIX\_NOT\_SQUARE\_OR\_CVECTOR***  
***MATRIX\_NOT\_SQUARE\_OR\_RVECTOR***  
***MATRIX\_NOT\_SQUARE\_OR\_VECTOR***  
***DIMS\_INVALID***  
***DIMS\_NOT\_EQUAL***  
***DIMS\_MISMATCH\_MATRIX***  
***DIMS\_MISMATCH\_CVECTOR***  
***DIMS\_MISMATCH\_RVECTOR***  
***DIMS\_MISMATCH\_VECTOR***  
***SUBSYS\_MISMATCH\_DIMS***  
***PERM\_INVALID***  
***NOT\_QUBIT\_GATE***  
***NOT\_QUBIT\_SUBSYS***  
***NOT\_BIPARTITE***  
***OUT\_OF\_RANGE***  
***TYPE\_MISMATCH***  
***UNDEFINED\_TYPE***  
***CUSTOM\_EXCEPTION***

### 6.3.2 Constructor & Destructor Documentation

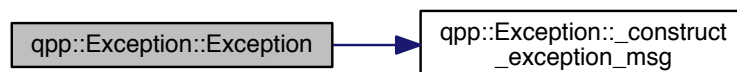
6.3.2.1 `qpp::Exception::Exception ( const std::string & where, const Type & type )` `[inline]`

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:



### 6.3.3 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]`

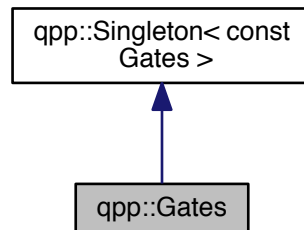
The documentation for this class was generated from the following file:

- [include/classes/exception.h](#)

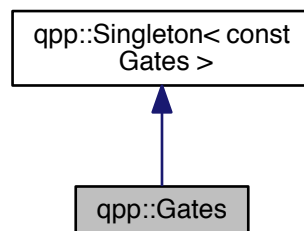
## 6.4 qpp::Gates Class Reference

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

Inheritance diagram for qpp::Gates:



Collaboration diagram for qpp::Gates:



## Public Member Functions

- [types::cmat Rn](#) (double theta, std::vector< double > n) const
- [types::cmat Zd](#) (std::size\_t D) const
- [types::cmat Fd](#) (std::size\_t D) const
- [types::cmat Xd](#) (std::size\_t D) const
- template<typename Derived = Eigen::MatrixXcd>  
Derived [ld](#) (std::size\_t D) const
- template<typename Derived1 , typename Derived2 >  
[types::DynMat](#)< typename  
Derived1::Scalar > [applyCTRL](#) (const Eigen::MatrixBase< Derived1 > &state, const Eigen::MatrixBase<  
Derived2 > &A, const std::vector< std::size\_t > &ctrl, const std::vector< std::size\_t > &subsys, std::size\_t  
n, std::size\_t d=2) const
- template<typename Derived1 , typename Derived2 >  
[types::DynMat](#)< typename  
Derived1::Scalar > [apply](#) (const Eigen::MatrixBase< Derived1 > &state, const Eigen::MatrixBase< De-  
rived2 > &A, const std::vector< std::size\_t > &subsys, const std::vector< std::size\_t > &dims) const
- template<typename Derived >  
[types::DynMat](#)< typename  
Derived::Scalar > [CTRL](#) (const Eigen::MatrixBase< Derived > &A, const std::vector< std::size\_t > &ctrl,  
const std::vector< std::size\_t > &subsys, std::size\_t n, std::size\_t d=2) const

## Public Attributes

- [types::cmat Id2](#) { types::cmat::Identity(2, 2) }
- [types::cmat H](#) { types::cmat::Zero(2, 2) }
- [types::cmat X](#) { types::cmat::Zero(2, 2) }
- [types::cmat Y](#) { types::cmat::Zero(2, 2) }
- [types::cmat Z](#) { types::cmat::Zero(2, 2) }
- [types::cmat S](#) { types::cmat::Zero(2, 2) }
- [types::cmat T](#) { types::cmat::Zero(2, 2) }
- [types::cmat CNOTab](#) { types::cmat::Identity(4, 4) }
- [types::cmat CZ](#) { types::cmat::Identity(4, 4) }
- [types::cmat CNOTba](#) { types::cmat::Zero(4, 4) }
- [types::cmat SWAP](#) { types::cmat::Identity(4, 4) }
- [types::cmat TOF](#) { types::cmat::Identity(8, 8) }
- [types::cmat FRED](#) { types::cmat::Identity(8, 8) }

## Private Member Functions

- [Gates](#) ()

## Friends

- class [Singleton< const Gates >](#)

## Additional Inherited Members

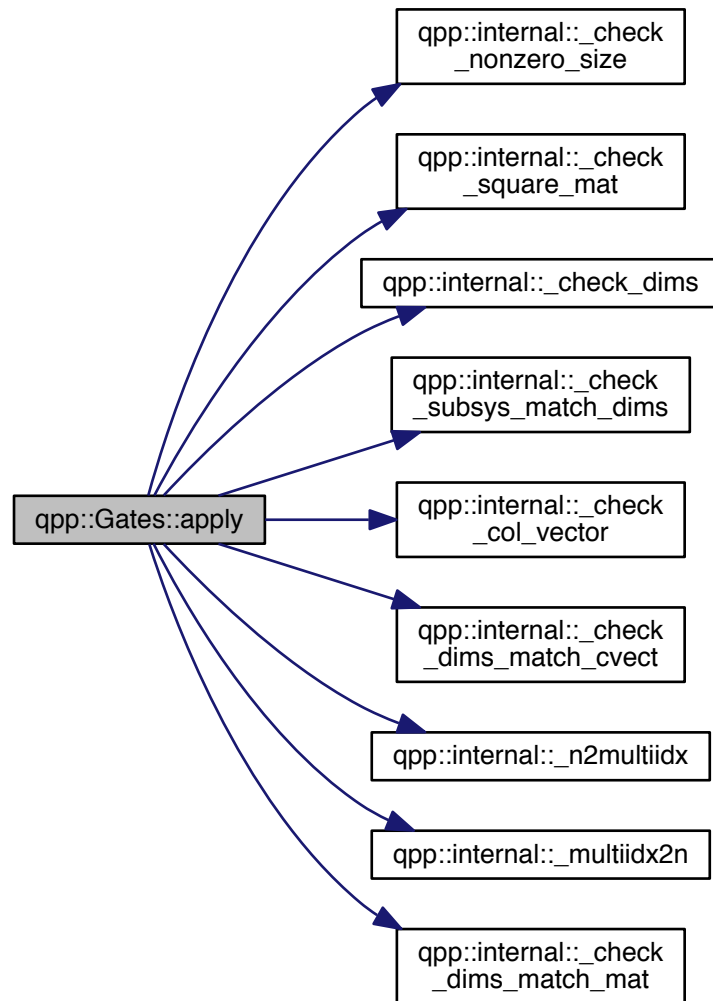
### 6.4.1 Constructor & Destructor Documentation

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

### 6.4.2 Member Function Documentation

```
6.4.2.1 template<typename Derived1 , typename Derived2 > types::DynMat<typename Derived1::Scalar>
qpp::Gates::apply ( const Eigen::MatrixBase< Derived1 > & state, const Eigen::MatrixBase< Derived2 > & A, const
std::vector< std::size_t > & subsys, const std::vector< std::size_t > & dims ) const [inline]
```

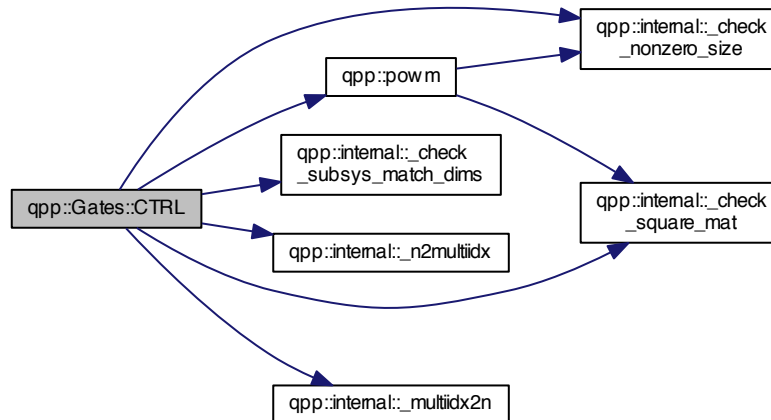
Here is the call graph for this function:



```
6.4.2.2 template<typename Derived1 , typename Derived2 > types::DynMat<typename Derived1::Scalar>
qpp::Gates::applyCTRL ( const Eigen::MatrixBase< Derived1 > & state, const Eigen::MatrixBase< Derived2 > & A,
const std::vector< std::size_t > & ctrl, const std::vector< std::size_t > & subsys, std::size_t n, std::size_t d = 2 )
const [inline]
```

6.4.2.3 `template<typename Derived > types::DynMat<typename Derived::Scalar> qpp::Gates::CTRL ( const Eigen::MatrixBase< Derived > & A, const std::vector< std::size_t > & ctrl, const std::vector< std::size_t > & subsys, std::size_t n, std::size_t d = 2 ) const [inline]`

Here is the call graph for this function:



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

Here is the call graph for this function:

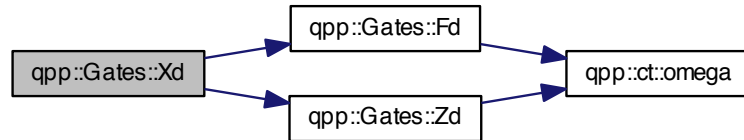


6.4.2.5 `template<typename Derived = Eigen::MatrixXcd> Derived qpp::Gates::Id ( std::size_t D ) const [inline]`

6.4.2.6 `types::cmat qpp::Gates::Rn ( double theta, std::vector< double > n ) const [inline]`

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

Here is the call graph for this function:



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

Here is the call graph for this function:



### 6.4.3 Friends And Related Function Documentation

#### 6.4.3.1 `friend class Singleton< const Gates > [friend]`

### 6.4.4 Member Data Documentation

#### 6.4.4.1 `types::cmat qpp::Gates::CNOTab { types::cmat::Identity(4, 4) }`

#### 6.4.4.2 `types::cmat qpp::Gates::CNOTba { types::cmat::Zero(4, 4) }`

#### 6.4.4.3 `types::cmat qpp::Gates::CZ { types::cmat::Identity(4, 4) }`

#### 6.4.4.4 `types::cmat qpp::Gates::FRED { types::cmat::Identity(8, 8) }`

#### 6.4.4.5 `types::cmat qpp::Gates::H { types::cmat::Zero(2, 2) }`

#### 6.4.4.6 `types::cmat qpp::Gates::Id2 { types::cmat::Identity(2, 2) }`

#### 6.4.4.7 `types::cmat qpp::Gates::S { types::cmat::Zero(2, 2) }`

#### 6.4.4.8 `types::cmat qpp::Gates::SWAP { types::cmat::Identity(4, 4) }`

#### 6.4.4.9 `types::cmat qpp::Gates::T { types::cmat::Zero(2, 2) }`

6.4.4.10 `types::cmat qpp::Gates::TOF { types::cmat::Identity(8, 8) }`

6.4.4.11 `types::cmat qpp::Gates::X { types::cmat::Zero(2, 2) }`

6.4.4.12 `types::cmat qpp::Gates::Y { types::cmat::Zero(2, 2) }`

6.4.4.13 `types::cmat qpp::Gates::Z { types::cmat::Zero(2, 2) }`

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

- [include/classes/gates.h](#)

## 6.5 qpp::NormalDistribution Class Reference

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

### Public Member Functions

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

### Protected Attributes

- `std::normal_distribution _d`

#### 6.5.1 Constructor & Destructor Documentation

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

#### 6.5.2 Member Function Documentation

6.5.2.1 `double qpp::NormalDistribution::sample ( )` `[inline]`

Here is the call graph for this function:



#### 6.5.3 Member Data Documentation

6.5.3.1 `std::normal_distribution qpp::NormalDistribution::_d` `[protected]`

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

- [include/classes/stat.h](#)



## 6.6 qpp::Qudit Class Reference

```
#include <qudit.h>
```

### Public Member Functions

- [Qudit](#) (const [types::cmat](#) &rho=[States::get\\_instance\(\)](#).pz0)
- [std::size\\_t measure](#) (const [types::cmat](#) &U, bool destructive=false)
- [std::size\\_t measure](#) (bool destructive=false)
- [types::cmat getRho](#) () const
- [std::size\\_t getD](#) () const

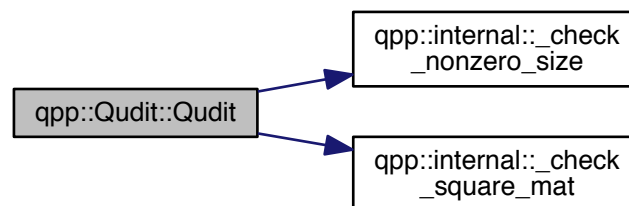
### Private Attributes

- [types::cmat\\_rho](#)
- [std::size\\_t \\_D](#)

### 6.6.1 Constructor & Destructor Documentation

6.6.1.1 `qpp::Qudit::Qudit ( const types::cmat & rho = States::get\_instance\(\) .pz0 )` `[inline]`

Here is the call graph for this function:



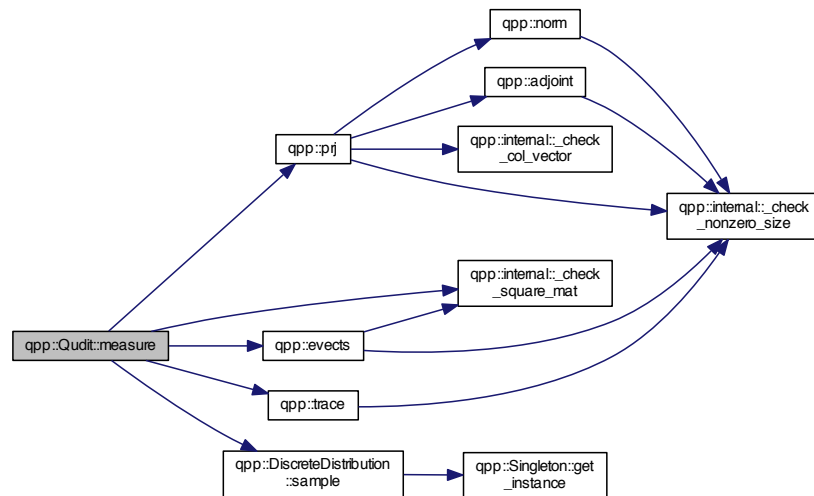
### 6.6.2 Member Function Documentation

6.6.2.1 `std::size_t qpp::Qudit::getD ( )` const `[inline]`

6.6.2.2 `types::cmat qpp::Qudit::getRho ( )` const `[inline]`

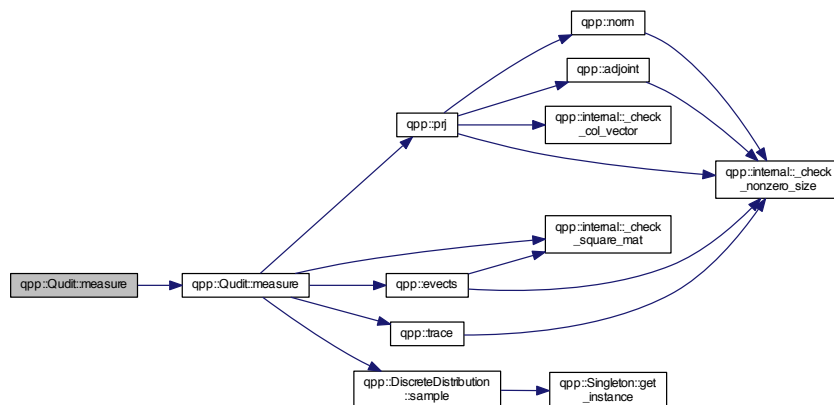
### 6.6.2.3 `std::size_t qpp::Qudit::measure ( const types::cmat & U, bool destructive = false ) [inline]`

Here is the call graph for this function:



### 6.6.2.4 `std::size_t qpp::Qudit::measure ( bool destructive = false ) [inline]`

Here is the call graph for this function:



## 6.6.3 Member Data Documentation

### 6.6.3.1 `std::size_t qpp::Qudit::_D [private]`

### 6.6.3.2 `types::cmat qpp::Qudit::_rho [private]`

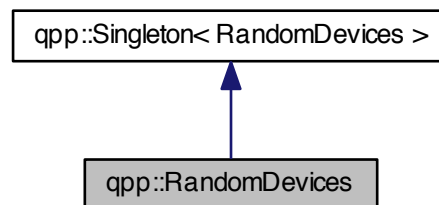
The documentation for this class was generated from the following file:

- [include/classes/qudit.h](#)

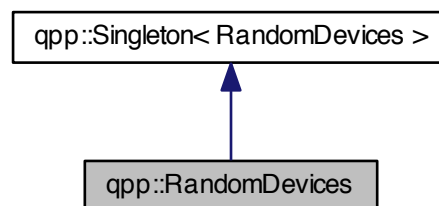
## 6.7 qpp::RandomDevices Class Reference

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

Inheritance diagram for qpp::RandomDevices:



Collaboration diagram for qpp::RandomDevices:



### Public Attributes

- [std::mt19937 \\_rng](#)

### Private Member Functions

- [RandomDevices \(\)](#)

### Private Attributes

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

### Friends

- class [Singleton< RandomDevices >](#)

## Additional Inherited Members

### 6.7.1 Constructor & Destructor Documentation

6.7.1.1 `qpp::RandomDevices::RandomDevices ( )` `[inline]`, `[private]`

### 6.7.2 Friends And Related Function Documentation

6.7.2.1 `friend class Singleton< RandomDevices >` `[friend]`

### 6.7.3 Member Data Documentation

6.7.3.1 `std::random_device qpp::RandomDevices::_rd` `[private]`

6.7.3.2 `std::mt19937 qpp::RandomDevices::_rng`

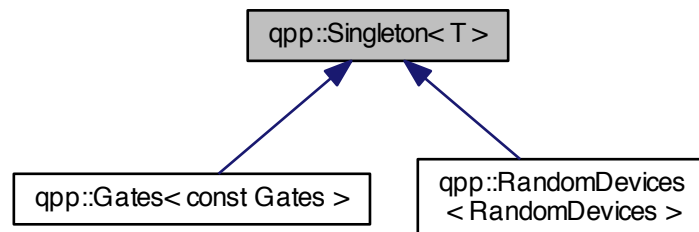
The documentation for this class was generated from the following file:

- `include/classes/randevs.h`

## 6.8 `qpp::Singleton< T >` Class Template Reference

```
#include <singleton.h>
```

Inheritance diagram for `qpp::Singleton< T >`:



## Static Public Member Functions

- static `T & get_instance ()`

## Protected Member Functions

- `Singleton ()`=default
- virtual `~Singleton ()`
- `Singleton (const Singleton &)=delete`
- `Singleton & operator= (const Singleton &)=delete`

### 6.8.1 Constructor & Destructor Documentation

6.8.1.1 `template<typename T> qpp::Singleton< T >::Singleton ( )` `[protected]`, `[default]`

6.8.1.2 `template<typename T> virtual qpp::Singleton< T >::~~Singleton ( )` `[inline]`, `[protected]`, `[virtual]`

6.8.1.3 `template<typename T> qpp::Singleton< T >::Singleton ( const Singleton< T > & )` `[protected]`, `[delete]`

### 6.8.2 Member Function Documentation

6.8.2.1 `template<typename T> static T& qpp::Singleton< T >::get_instance ( )` `[inline]`, `[static]`

6.8.2.2 `template<typename T> Singleton& qpp::Singleton< T >::operator= ( const Singleton< T > & )` `[protected]`, `[delete]`

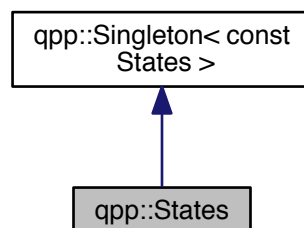
The documentation for this class was generated from the following file:

- `include/classes/singleton.h`

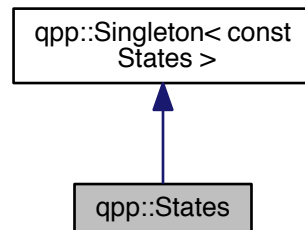
## 6.9 qpp::States Class Reference

```
#include <states.h>
```

Inheritance diagram for qpp::States:



Collaboration diagram for qpp::States:



## Public Attributes

- `types::ket x0` { `types::ket::Zero(2)` }
- `types::ket x1` { `types::ket::Zero(2)` }
- `types::ket y0` { `types::ket::Zero(2)` }
- `types::ket y1` { `types::ket::Zero(2)` }
- `types::ket z0` { `types::ket::Zero(2)` }
- `types::ket z1` { `types::ket::Zero(2)` }
- `types::cmat px0` { `types::cmat::Zero(2, 2)` }
- `types::cmat px1` { `types::cmat::Zero(2, 2)` }
- `types::cmat py0` { `types::cmat::Zero(2, 2)` }
- `types::cmat py1` { `types::cmat::Zero(2, 2)` }
- `types::cmat pz0` { `types::cmat::Zero(2, 2)` }
- `types::cmat pz1` { `types::cmat::Zero(2, 2)` }
- `types::ket b00` { `types::ket::Zero(4)` }
- `types::ket b01` { `types::ket::Zero(4)` }
- `types::ket b10` { `types::ket::Zero(4)` }
- `types::ket b11` { `types::ket::Zero(4)` }
- `types::cmat pb00` { `types::cmat::Zero(4, 4)` }
- `types::cmat pb01` { `types::cmat::Zero(4, 4)` }
- `types::cmat pb10` { `types::cmat::Zero(4, 4)` }
- `types::cmat pb11` { `types::cmat::Zero(4, 4)` }
- `types::ket GHZ` { `types::ket::Zero(8)` }
- `types::ket W` { `types::ket::Zero(8)` }
- `types::cmat pGHZ` { `types::cmat::Zero(8, 8)` }
- `types::cmat pW` { `types::cmat::Zero(8, 8)` }

## Private Member Functions

- `States` ()

## Friends

- class `Singleton< const States >`

## Additional Inherited Members

### 6.9.1 Constructor & Destructor Documentation

6.9.1.1 `qpp::States::States ( )` `[inline]`, `[private]`

### 6.9.2 Friends And Related Function Documentation

6.9.2.1 `friend class Singleton< const States >` `[friend]`

### 6.9.3 Member Data Documentation

6.9.3.1 `types::ket qpp::States::b00 { types::ket::Zero(4) }`

6.9.3.2 `types::ket qpp::States::b01 { types::ket::Zero(4) }`

6.9.3.3 `types::ket qpp::States::b10 { types::ket::Zero(4) }`

6.9.3.4 `types::ket qpp::States::b11 { types::ket::Zero(4) }`

6.9.3.5 `types::ket qpp::States::GHZ { types::ket::Zero(8) }`

6.9.3.6 `types::cmat qpp::States::pb00 { types::cmat::Zero(4, 4) }`

6.9.3.7 `types::cmat qpp::States::pb01 { types::cmat::Zero(4, 4) }`

6.9.3.8 `types::cmat qpp::States::pb10 { types::cmat::Zero(4, 4) }`

6.9.3.9 `types::cmat qpp::States::pb11 { types::cmat::Zero(4, 4) }`

6.9.3.10 `types::cmat qpp::States::pGHZ { types::cmat::Zero(8, 8) }`

6.9.3.11 `types::cmat qpp::States::pW { types::cmat::Zero(8, 8) }`

6.9.3.12 `types::cmat qpp::States::px0 { types::cmat::Zero(2, 2) }`

6.9.3.13 `types::cmat qpp::States::px1 { types::cmat::Zero(2, 2) }`

6.9.3.14 `types::cmat qpp::States::py0 { types::cmat::Zero(2, 2) }`

6.9.3.15 `types::cmat qpp::States::py1 { types::cmat::Zero(2, 2) }`

6.9.3.16 `types::cmat qpp::States::pz0 { types::cmat::Zero(2, 2) }`

6.9.3.17 `types::cmat qpp::States::pz1 { types::cmat::Zero(2, 2) }`

6.9.3.18 `types::ket qpp::States::W { types::ket::Zero(8) }`

6.9.3.19 `types::ket qpp::States::x0 { types::ket::Zero(2) }`

6.9.3.20 `types::ket qpp::States::x1 { types::ket::Zero(2) }`

6.9.3.21 `types::ket qpp::States::y0 { types::ket::Zero(2) }`

6.9.3.22 `types::ket qpp::States::y1 { types::ket::Zero(2) }`

6.9.3.23 `types::ket qpp::States::z0 { types::ket::Zero(2) }`

6.9.3.24 `types::ket qpp::States::z1 { types::ket::Zero(2) }`

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

- [include/classes/states.h](#)

## 6.10 qpp::Timer Class Reference

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

### Public Member Functions

- [Timer](#) ()
- void [tic](#) ()
- void [toc](#) ()
- double [seconds](#) () const

### Protected Attributes

- `std::chrono::steady_clock::time_point` [\\_start](#)
- `std::chrono::steady_clock::time_point` [\\_end](#)

### Friends

- `std::ostream & operator<< (std::ostream &os, const Timer &rhs)`

### 6.10.1 Constructor & Destructor Documentation

6.10.1.1 `qpp::Timer::Timer ( )` [[inline](#)]

### 6.10.2 Member Function Documentation

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

6.10.2.2 `void qpp::Timer::tic ( )` [[inline](#)]

6.10.2.3 `void qpp::Timer::toc ( )` [[inline](#)]

### 6.10.3 Friends And Related Function Documentation

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

### 6.10.4 Member Data Documentation

6.10.4.1 `std::chrono::steady_clock::time_point` `qpp::Timer::_end` [[protected](#)]

6.10.4.2 `std::chrono::steady_clock::time_point` `qpp::Timer::_start` [[protected](#)]

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

- [include/classes/timer.h](#)



## 6.11 qpp::UniformIntDistribution Class Reference

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

### Public Member Functions

- [UniformIntDistribution](#) (int a=0, int b=1)
- int [sample](#) ()

### Protected Attributes

- std::uniform\_int\_distribution [\\_d](#)

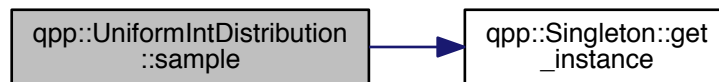
#### 6.11.1 Constructor & Destructor Documentation

6.11.1.1 `qpp::UniformIntDistribution::UniformIntDistribution ( int a = 0, int b = 1 )` `[inline]`

#### 6.11.2 Member Function Documentation

6.11.2.1 `int qpp::UniformIntDistribution::sample ( )` `[inline]`

Here is the call graph for this function:



#### 6.11.3 Member Data Documentation

6.11.3.1 `std::uniform_int_distribution qpp::UniformIntDistribution::_d` `[protected]`

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

- include/classes/[stat.h](#)

## 6.12 qpp::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`

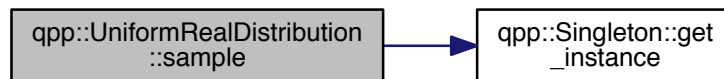
## 6.12.1 Constructor & Destructor Documentation

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

## 6.12.2 Member Function Documentation

6.12.2.1 `double qpp::UniformRealDistribution::sample ( )` `[inline]`

Here is the call graph for this function:



## 6.12.3 Member Data Documentation

6.12.3.1 `std::uniform_real_distribution qpp::UniformRealDistribution::_d` `[protected]`

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

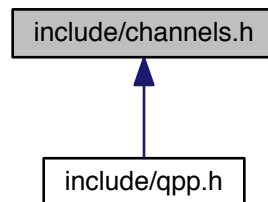
- `include/classes/stat.h`

## Chapter 7

# File Documentation

### 7.1 include/channels.h File Reference

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



### Namespaces

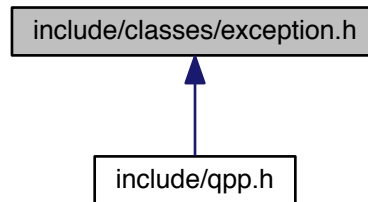
- [qpp](#)

### Functions

- `types::cmat qpp::super (const std::vector< types::cmat > &Ks)`
- `types::cmat qpp::choi (const std::vector< types::cmat > &Ks)`
- `std::vector< types::cmat > qpp::choi2kraus (const types::cmat &A)`
- `template<typename Derived >  
types::cmat qpp::channel (const Eigen::MatrixBase< Derived > &rho, const std::vector< types::cmat > &Ks)`
- `template<typename Derived >  
types::cmat qpp::channel (const Eigen::MatrixBase< Derived > &rho, const std::vector< types::cmat > &Ks,  
const std::vector< std::size_t > &subsys, const std::vector< std::size_t > &dims)`

## 7.2 include/classes/exception.h File Reference

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



### Classes

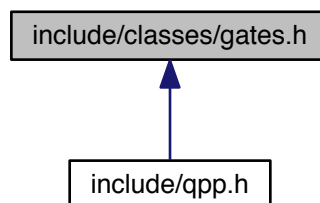
- class [qpp::Exception](#)

### Namespaces

- [qpp](#)

## 7.3 include/classes/gates.h File Reference

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



### Classes

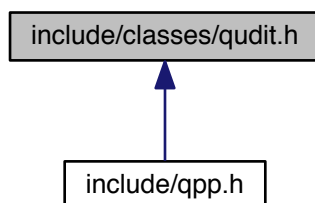
- class [qpp::Gates](#)

### Namespaces

- [qpp](#)

## 7.4 include/classes/qudit.h File Reference

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



### Classes

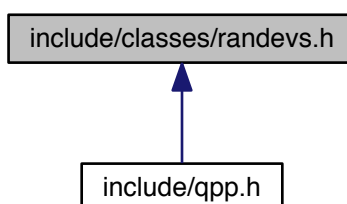
- class [qpp::Qudit](#)

### Namespaces

- [qpp](#)

## 7.5 include/classes/randevs.h File Reference

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



### Classes

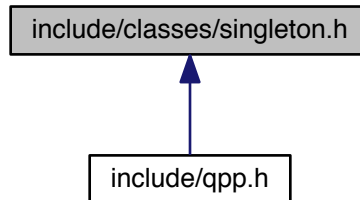
- class [qpp::RandomDevices](#)

### Namespaces

- [qpp](#)

## 7.6 include/classes/singleton.h File Reference

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



### Classes

- class [qpp::Singleton< T >](#)

### Namespaces

- [qpp](#)

### Macros

- [#define CLASS\\_SINGLETON\(Foo\)](#)
- [#define CLASS\\_CONST\\_SINGLETON\(Foo\)](#)

#### 7.6.1 Macro Definition Documentation

##### 7.6.1.1 [#define CLASS\\_CONST\\_SINGLETON\( Foo \)](#)

###### Value:

```

class Foo: public Singleton<const Foo>\
{
    friend class Singleton<const Foo>;
}
  
```

##### 7.6.1.2 [#define CLASS\\_SINGLETON\( Foo \)](#)

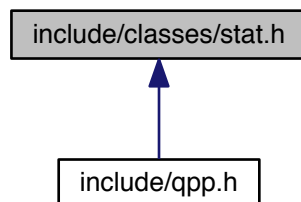
###### Value:

```

class Foo: public Singleton<Foo>\
{
    friend class Singleton<Foo>;
}
  
```

## 7.7 include/classes/stat.h File Reference

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



### Classes

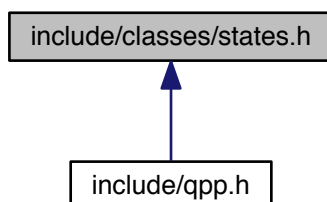
- class [qpp::NormalDistribution](#)
- class [qpp::UniformRealDistribution](#)
- class [qpp::UniformIntDistribution](#)
- class [qpp::DiscreteDistribution](#)
- class [qpp::DiscreteDistributionAbsSquare](#)

### Namespaces

- [qpp](#)

## 7.8 include/classes/states.h File Reference

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



### Classes

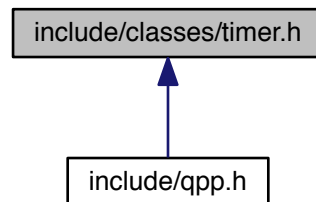
- class [qpp::States](#)

## Namespaces

- [qpp](#)

## 7.9 include/classes/timer.h File Reference

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



## Classes

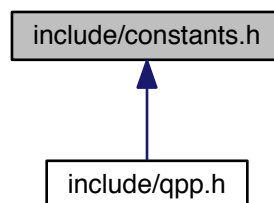
- class [qpp::Timer](#)

## Namespaces

- [qpp](#)

## 7.10 include/constants.h File Reference

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



## Namespaces

- [qpp](#)



- [qpp::ct](#)

## Functions

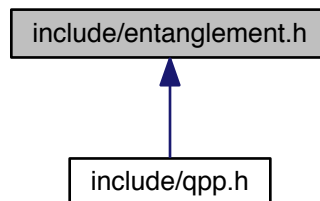
- constexpr std::complex< double > [qpp::operator""\\_i](#) (unsigned long long int x)  
*User-defined literal for complex  $i = \sqrt{-1}$  (integer overload)*
- constexpr std::complex< double > [qpp::operator""\\_i](#) (long double x)  
*User-defined literal for complex  $i = \sqrt{-1}$  (real overload)*
- std::complex< double > [qpp::ct::omega](#) (std::size\_t D)  
*D-th root of unity.*

## Variables

- constexpr double [qpp::ct::chop](#) = 1e-10  
*Used in [qpp::disp\(\)](#) and [qpp::displn\(\)](#) for setting to zero numbers that have their absolute value smaller than [qpp::ct::chop](#).*
- constexpr double [qpp::ct::eps](#) = 1e-12  
*Used to decide whether a number or expression in double precision is zero or not.*
- constexpr std::size\_t [qpp::ct::maxn](#) = 64  
*Maximum number of qubits.*
- constexpr double [qpp::ct::pi](#) = 3.141592653589793238462643383279502884  
 $\pi$
- constexpr double [qpp::ct::ee](#) = 2.718281828459045235360287471352662497  
*Base of natural logarithm,  $e$ .*

## 7.11 include/entanglement.h File Reference

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



## Namespaces

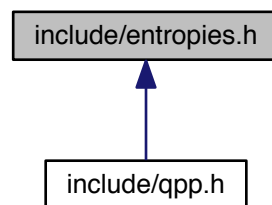
- [qpp](#)

## Functions

- `template<typename Derived >`  
`types::cmat qpp::schmidtcoeff (const Eigen::MatrixBase< Derived > &A, const std::vector< std::size_t > &dims)`
- `template<typename Derived >`  
`types::cmat qpp::schmidtU (const Eigen::MatrixBase< Derived > &A, const std::vector< std::size_t > &dims)`
- `template<typename Derived >`  
`types::cmat qpp::schmidtV (const Eigen::MatrixBase< Derived > &A, const std::vector< std::size_t > &dims)`
- `template<typename Derived >`  
`types::cmat qpp::schmidtprob (const Eigen::MatrixBase< Derived > &A, const std::vector< std::size_t > &dims)`
- `template<typename Derived >`  
`double qpp::entanglement (const Eigen::MatrixBase< Derived > &A, const std::vector< std::size_t > &dims)`
- `template<typename Derived >`  
`double qpp::gconcurrence (const Eigen::MatrixBase< Derived > &A)`

## 7.12 include/entropies.h File Reference

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



## Namespaces

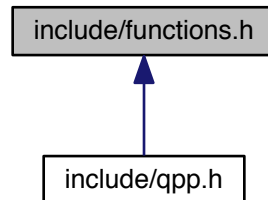
- [qpp](#)

## Functions

- `template<typename Derived >`  
`double qpp::shannon (const Eigen::MatrixBase< Derived > &A)`
- `template<typename Derived >`  
`double qpp::renyi (const double alpha, const Eigen::MatrixBase< Derived > &A)`
- `template<typename Derived >`  
`double qpp::renyi\_inf (const Eigen::MatrixBase< Derived > &A)`
- `template<typename Derived >`  
`double qpp::tsallis (const double alpha, const Eigen::MatrixBase< Derived > &A)`
- `template<typename Derived >`  
`double qpp::qmutualinfo (const Eigen::MatrixBase< Derived > &A, const std::vector< std::size_t > &subsys, const std::vector< std::size_t > &dims)`

## 7.13 include/functions.h File Reference

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



### Namespaces

- [qpp](#)

### Functions

- `template<typename Derived >`  
`types::DynMat< typename`  
`Derived::Scalar > qpp::transpose (const Eigen::MatrixBase< Derived > &A)`  
*Transpose.*
- `template<typename Derived >`  
`types::DynMat< typename`  
`Derived::Scalar > qpp::conjugate (const Eigen::MatrixBase< Derived > &A)`  
*Complex conjugate.*
- `template<typename Derived >`  
`types::DynMat< typename`  
`Derived::Scalar > qpp::adjoint (const Eigen::MatrixBase< Derived > &A)`  
*Adjoint.*
- `template<typename Derived >`  
`types::DynMat< typename`  
`Derived::Scalar > qpp::inverse (const Eigen::MatrixBase< Derived > &A)`  
*Inverse.*
- `template<typename Derived >`  
`Derived::Scalar qpp::trace (const Eigen::MatrixBase< Derived > &A)`  
*Trace.*
- `template<typename Derived >`  
`Derived::Scalar qpp::det (const Eigen::MatrixBase< Derived > &A)`  
*Determinant.*
- `template<typename Derived >`  
`Derived::Scalar qpp::logdet (const Eigen::MatrixBase< Derived > &A)`  
*Logarithm of the determinant.*
- `template<typename Derived >`  
`Derived::Scalar qpp::sum (const Eigen::MatrixBase< Derived > &A)`  
*Element-wise sum.*

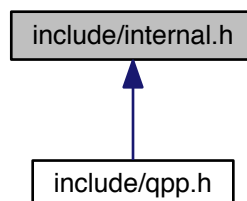
- `template<typename Derived >`  
`double qpp::norm (const Eigen::MatrixBase< Derived > &A)`  
*Trace norm.*
- `template<typename Derived >`  
`types::cmat qpp::evals (const Eigen::MatrixBase< Derived > &A)`  
*Eigenvalues.*
- `template<typename Derived >`  
`types::cmat qpp::evecs (const Eigen::MatrixBase< Derived > &A)`  
*Eigenvectors.*
- `template<typename Derived >`  
`types::dmat qpp::hevals (const Eigen::MatrixBase< Derived > &A)`  
*Hermitian eigenvalues.*
- `template<typename Derived >`  
`types::cmat qpp::hevecs (const Eigen::MatrixBase< Derived > &A)`  
*Hermitian eigenvectors.*
- `template<typename Derived >`  
`types::cmat qpp::funm (const Eigen::MatrixBase< Derived > &A, types::cplx(*f)(const types::cplx &))`  
*Functional calculus  $f(A)$*
- `template<typename Derived >`  
`types::cmat qpp::sqrtm (const Eigen::MatrixBase< Derived > &A)`  
*Matrix square root.*
- `template<typename Derived >`  
`types::cmat qpp::absm (const Eigen::MatrixBase< Derived > &A)`  
*Matrix absolut value.*
- `template<typename Derived >`  
`types::cmat qpp::expm (const Eigen::MatrixBase< Derived > &A)`  
*Matrix exponential.*
- `template<typename Derived >`  
`types::cmat qpp::logm (const Eigen::MatrixBase< Derived > &A)`  
*Matrix logarithm.*
- `template<typename Derived >`  
`types::cmat qpp::sinm (const Eigen::MatrixBase< Derived > &A)`  
*Matrix sin.*
- `template<typename Derived >`  
`types::cmat qpp::cosm (const Eigen::MatrixBase< Derived > &A)`  
*Matrix cos.*
- `template<typename Derived >`  
`types::cmat qpp::spectralpwm (const Eigen::MatrixBase< Derived > &A, const types::cplx z)`  
*Matrix power.*
- `template<typename Derived >`  
`types::DynMat< typename`  
`Derived::Scalar > qpp::pwm (const Eigen::MatrixBase< Derived > &A, std::size_t n)`  
*Matrix power.*
- `template<typename OutputScalar , typename Derived >`  
`types::DynMat< OutputScalar > qpp::cwise (const Eigen::MatrixBase< Derived > &A, OutputScalar(*f)(const typename Derived::Scalar &))`  
*Functor.*
- `template<typename T >`  
`types::DynMat< typename T::Scalar > qpp::kron (const T &head)`  
*Kronecker product (variadic overload)*
- `template<typename T , typename... Args>`  
`types::DynMat< typename T::Scalar > qpp::kron (const T &head, const Args &...tail)`  
*Kronecker product (variadic overload)*

- `template<typename Derived >`  
`types::DynMat< typename`  
`Derived::Scalar > qpp::kron (const std::vector< Derived > &As)`  
*Kronecker product (std::vector overload)*
- `template<typename Derived >`  
`types::DynMat< typename`  
`Derived::Scalar > qpp::kron (const std::initializer_list< Derived > &As)`  
*Kronecker product (std::initializer\_list overload)*
- `template<typename Derived >`  
`types::DynMat< typename`  
`Derived::Scalar > qpp::kronpow (const Eigen::MatrixBase< Derived > &A, std::size_t n)`  
*Kronecker power.*
- `template<typename Derived >`  
`types::DynMat< typename`  
`Derived::Scalar > qpp::reshape (const Eigen::MatrixBase< Derived > &A, std::size_t rows, std::size_t cols)`  
*Reshape.*
- `template<typename Derived >`  
`types::DynMat< typename`  
`Derived::Scalar > qpp::syspermute (const Eigen::MatrixBase< Derived > &A, const std::vector< std::size_t`  
`> &perm, const std::vector< std::size_t > &dims)`  
*System permutation.*
- `template<typename Derived >`  
`types::DynMat< typename`  
`Derived::Scalar > qpp::ptrace1 (const Eigen::MatrixBase< Derived > &A, const std::vector< std::size_t >`  
`&dims)`  
*Partial trace.*
- `template<typename Derived >`  
`types::DynMat< typename`  
`Derived::Scalar > qpp::ptrace2 (const Eigen::MatrixBase< Derived > &A, const std::vector< std::size_t >`  
`&dims)`  
*Partial trace.*
- `template<typename Derived >`  
`types::DynMat< typename`  
`Derived::Scalar > qpp::ptrace (const Eigen::MatrixBase< Derived > &A, const std::vector< std::size_t >`  
`&subsys, const std::vector< std::size_t > &dims)`  
*Partial trace.*
- `template<typename Derived >`  
`types::DynMat< typename`  
`Derived::Scalar > qpp::ptranspose (const Eigen::MatrixBase< Derived > &A, const std::vector< std::size_t`  
`> &subsys, const std::vector< std::size_t > &dims)`  
*Partial transpose.*
- `template<typename Derived1 , typename Derived2 >`  
`types::DynMat< typename`  
`Derived1::Scalar > qpp::comm (const Eigen::MatrixBase< Derived1 > &A, const Eigen::MatrixBase< De-`  
`rived2 > &B)`  
*Commutator.*
- `template<typename Derived1 , typename Derived2 >`  
`types::DynMat< typename`  
`Derived1::Scalar > qpp::anticomm (const Eigen::MatrixBase< Derived1 > &A, const Eigen::MatrixBase<`  
`Derived2 > &B)`  
*Anti-commutator.*
- `template<typename Derived >`  
`types::DynMat< typename`  
`Derived::Scalar > qpp::prj (const Eigen::MatrixBase< Derived > &V)`  
*Projector.*

- `template<typename Derived >`  
`types::DynMat< typename`  
`Derived::Scalar > qpp::expandout (const Eigen::MatrixBase< Derived > &A, std::size_t pos, const std::vector< std::size_t > &dims)`  
*Expand out.*
- `template<typename Derived >`  
`types::DynMat< typename`  
`Derived::Scalar > qpp::grams (const std::vector< Derived > &Vs)`  
*Gram-Schmidt orthogonalization (std::vector overload)*
- `template<typename Derived >`  
`types::DynMat< typename`  
`Derived::Scalar > qpp::grams (const std::initializer_list< Derived > &Vs)`  
*Gram-Schmidt orthogonalization (std::initializer\_list overload)*
- `template<typename Derived >`  
`types::DynMat< typename`  
`Derived::Scalar > qpp::grams (const Eigen::MatrixBase< Derived > &A)`  
*Gram-Schmidt orthogonalization (Eigen expression (matrix) overload)*
- `std::vector< std::size_t > qpp::n2multiidx (std::size_t n, const std::vector< std::size_t > &dims)`  
*Non-negative integer index to multi-index.*
- `std::size_t qpp::multiidx2n (const std::vector< std::size_t > &midx, const std::vector< std::size_t > &dims)`  
*Multi-index to non-negative integer index.*
- `types::ket qpp::mket (const std::vector< std::size_t > &mask)`  
*Multi-partite qubit ket.*
- `types::ket qpp::mket (const std::vector< std::size_t > &mask, const std::vector< std::size_t > &dims)`  
*Multi-partite qudit ket (different dimensions overload)*
- `types::ket qpp::mket (const std::vector< std::size_t > &mask, std::size_t d)`  
*Multi-partite qudit ket (same dimensions overload)*
- `std::vector< std::size_t > qpp::invperm (const std::vector< std::size_t > &perm)`  
*Inverse permutation.*
- `std::vector< std::size_t > qpp::compperm (const std::vector< std::size_t > &perm, const std::vector< std::size_t > &sigma)`  
*Compose permutations.*

## 7.14 include/internal.h File Reference

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



## Namespaces

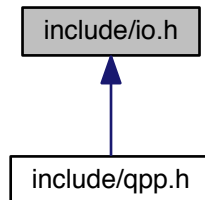
- [qpp](#)
- [qpp::internal](#)

## Functions

- void [qpp::internal::\\_n2multiidx](#) (std::size\_t n, std::size\_t numdims, const std::size\_t \*dims, std::size\_t \*result)
- std::size\_t [qpp::internal::\\_multiidx2n](#) (const std::size\_t \*midx, std::size\_t numdims, const std::size\_t \*dims)
- template<typename Derived >  
bool [qpp::internal::\\_check\\_square\\_mat](#) (const Eigen::MatrixBase< Derived > &A)
- template<typename Derived >  
bool [qpp::internal::\\_check\\_vector](#) (const Eigen::MatrixBase< Derived > &A)
- template<typename Derived >  
bool [qpp::internal::\\_check\\_row\\_vector](#) (const Eigen::MatrixBase< Derived > &A)
- template<typename Derived >  
bool [qpp::internal::\\_check\\_col\\_vector](#) (const Eigen::MatrixBase< Derived > &A)
- template<typename T >  
bool [qpp::internal::\\_check\\_nonzero\\_size](#) (const T &x)
- bool [qpp::internal::\\_check\\_dims](#) (const std::vector< std::size\_t > &dims)
- template<typename Derived >  
bool [qpp::internal::\\_check\\_dims\\_match\\_mat](#) (const std::vector< std::size\_t > &dims, const Eigen::Matrix↵  
Base< Derived > &A)
- template<typename Derived >  
bool [qpp::internal::\\_check\\_dims\\_match\\_cvect](#) (const std::vector< std::size\_t > &dims, const Eigen::Matrix↵  
Base< Derived > &V)
- template<typename Derived >  
bool [qpp::internal::\\_check\\_dims\\_match\\_rvect](#) (const std::vector< std::size\_t > &dims, const Eigen::Matrix↵  
Base< Derived > &V)
- bool [qpp::internal::\\_check\\_eq\\_dims](#) (const std::vector< std::size\_t > &dims, std::size\_t dim)
- bool [qpp::internal::\\_check\\_subsys\\_match\\_dims](#) (const std::vector< std::size\_t > &subsys, const std↵  
::vector< std::size\_t > &dims)
- bool [qpp::internal::\\_check\\_perm](#) (const std::vector< std::size\_t > &perm)
- template<typename Derived1 , typename Derived2 >  
types::DynMat< typename  
Derived1::Scalar > [qpp::internal::\\_kron2](#) (const Eigen::MatrixBase< Derived1 > &A, const Eigen::Matrix↵  
Base< Derived2 > &B)
- template<typename T >  
void [qpp::internal::variadic\\_vector\\_emplace](#) (std::vector< T > &)
- template<typename T , typename First , typename... Args>  
void [qpp::internal::variadic\\_vector\\_emplace](#) (std::vector< T > &v, First &&first, Args &&...args)

## 7.15 include/io.h File Reference

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



### Namespaces

- [qpp](#)

### Functions

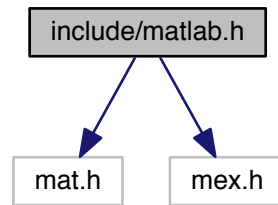
- `template<typename T >`  
`void qpp::disp (const T &x, const std::string &separator, const std::string &start="[" , const std::string &end="]", std::ostream &os=std::cout)`
- `template<typename T >`  
`void qpp::displn (const T &x, const std::string &separator, const std::string &start="[" , const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

## 7.16 include/matlab.h File Reference

```
#include "mat.h"
#include "mex.h"
```



Include dependency graph for matlab.h:



## Namespaces

- [qpp](#)

## Functions

- `template<typename Derived >`  
`Derived qpp::loadMATLABmatrix (const std::string &mat_file, const std::string &var_name)`
- `template<>`  
`types::dmat qpp::loadMATLABmatrix (const std::string &mat_file, const std::string &var_name)`
- `template<>`  
`types::cmat qpp::loadMATLABmatrix (const std::string &mat_file, const std::string &var_name)`
- `template<typename Derived >`  
`void qpp::saveMATLABmatrix (const Eigen::MatrixBase< Derived > &A, const std::string &mat_file, const std::string &var_name, const std::string &mode)`
- `template<>`  
`void qpp::saveMATLABmatrix (const Eigen::MatrixBase< types::dmat > &A, const std::string &mat_file, const std::string &var_name, const std::string &mode)`
- `template<>`  
`void qpp::saveMATLABmatrix (const Eigen::MatrixBase< typename types::cmat > &A, const std::string &mat_file, const std::string &var_name, const std::string &mode)`

## 7.17 include/qpp.h File Reference

```
#include <algorithm>
#include <chrono>
#include <cmath>
#include <complex>
#include <cstdlib>
#include <cstring>
#include <exception>
#include <fstream>
#include <functional>
#include <iomanip>
#include <iostream>
#include <iterator>
#include <numeric>
#include <ostream>
#include <random>
#include <stdexcept>
#include <string>
#include <type_traits>
#include <utility>
#include <vector>
#include <Eigen/Dense>
#include <Eigen/SVD>
#include "constants.h"
#include "types.h"
#include "classes/exception.h"
#include "classes/singleton.h"
#include "classes/states.h"
#include "classes/randevs.h"
#include "internal.h"
#include "functions.h"
#include "classes/gates.h"
#include "classes/stat.h"
#include "entropies.h"
#include "entanglement.h"
#include "channels.h"
#include "io.h"
#include "random.h"
#include "classes/qudit.h"
#include "classes/timer.h"
```

Include dependency graph for qpp.h:



### Namespaces

- [qpp](#)

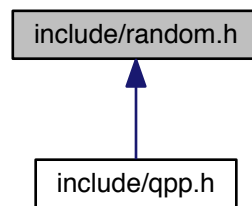
### Variables

- RandomDevices & [qpp::rdevs](#) = RandomDevices::get\_instance()  
[qpp::RandomDevices Singleton](#)
- const Gates & [qpp::gt](#) = Gates::get\_instance()

- `qpp::Gates` *const Singleton*
- `const States & qpp::st = States::get_instance()`
- `qpp::States` *const Singleton*

## 7.18 include/random.h File Reference

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



### Namespaces

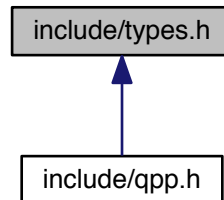
- `qpp`

### Functions

- `template<typename Derived >`  
Derived `qpp::rand` (`std::size_t` rows, `std::size_t` cols, double a=0, double b=1)
- `template<>`  
`types::dmat qpp::rand` (`std::size_t` rows, `std::size_t` cols, double a, double b)
- `template<>`  
`types::cmat qpp::rand` (`std::size_t` rows, `std::size_t` cols, double a, double b)
- double `qpp::rand` (double a=0, double b=1)
- long long `qpp::randint` (long long a, long long b)
- `template<typename Derived >`  
Derived `qpp::randn` (`std::size_t` rows, `std::size_t` cols, double mean=0, double sigma=1)
- `template<>`  
`types::dmat qpp::randn` (`std::size_t` rows, `std::size_t` cols, double mean, double sigma)
- `template<>`  
`types::cmat qpp::randn` (`std::size_t` rows, `std::size_t` cols, double mean, double sigma)
- double `qpp::randn` (double mean=0, double sigma=1)
- `types::cmat qpp::randU` (`std::size_t` D)
- `types::cmat qpp::randV` (`std::size_t` Din, `std::size_t` Dout)
- `std::vector< types::cmat > qpp::randkraus` (`std::size_t` n, `std::size_t` D)
- `types::cmat qpp::randH` (`std::size_t` D)
- `types::ket qpp::randket` (`std::size_t` D)
- `types::cmat qpp::randrho` (`std::size_t` D)
- `std::vector< std::size_t > qpp::randperm` (`std::size_t` n)

## 7.19 include/types.h File Reference

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



### Namespaces

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

### Typedefs

- using [qpp::types::cplx](#) = `std::complex< double >`  
*Complex number in double precision.*
- using [qpp::types::cmat](#) = `Eigen::MatrixXcd`  
*Complex (double precision) dynamic Eigen matrix.*
- using [qpp::types::dmat](#) = `Eigen::MatrixXd`  
*Real (double precision) dynamic Eigen matrix.*
- using [qpp::types::ket](#) = `Eigen::Matrix< cplx, Eigen::Dynamic, 1 >`  
*Complex (double precision) dynamic Eigen column matrix.*
- using [qpp::types::bra](#) = `Eigen::Matrix< cplx, 1, Eigen::Dynamic >`  
*Complex (double precision) dynamic Eigen row matrix.*
- `template<typename Scalar >`  
using [qpp::types::DynMat](#) = `Eigen::Matrix< Scalar, Eigen::Dynamic, Eigen::Dynamic >`  
*Dynamic Eigen matrix over the field specified by Scalar.*

# Index

- absm
  - qpp, [14](#)
- adjoint
  - qpp, [15](#)
- anticomm
  - qpp, [15](#)
- CUSTOM\_EXCEPTION
  - qpp::Exception, [69](#)
- channel
  - qpp, [16](#)
- choi
  - qpp, [17](#)
- choi2kraus
  - qpp, [17](#)
- comm
  - qpp, [18](#)
- compperm
  - qpp, [18](#)
- conjugate
  - qpp, [20](#)
- cosm
  - qpp, [20](#)
- cwise
  - qpp, [21](#)
- DIMS\_INVALID
  - qpp::Exception, [69](#)
- DIMS\_MISMATCH\_CVECTOR
  - qpp::Exception, [69](#)
- DIMS\_MISMATCH\_MATRIX
  - qpp::Exception, [69](#)
- DIMS\_MISMATCH\_RVECTOR
  - qpp::Exception, [69](#)
- DIMS\_MISMATCH\_VECTOR
  - qpp::Exception, [69](#)
- DIMS\_NOT\_EQUAL
  - qpp::Exception, [69](#)
- det
  - qpp, [21](#)
- disp
  - qpp, [22](#)
- displn
  - qpp, [22](#), [23](#)
- entanglement
  - qpp, [24](#)
- evals
  - qpp, [24](#)
- evects
  - qpp, [25](#)
- expandout
  - qpp, [25](#)
- expm
  - qpp, [26](#)
- funm
  - qpp, [27](#)
- gconcurrency
  - qpp, [27](#)
- grams
  - qpp, [28](#), [29](#)
- gt
  - qpp, [60](#)
- hevals
  - qpp, [29](#)
- hevects
  - qpp, [30](#)
- inverse
  - qpp, [30](#)
- invperm
  - qpp, [31](#)
- kron
  - qpp, [31–33](#)
- kronpow
  - qpp, [33](#)
- load
  - qpp, [34](#)
- logdet
  - qpp, [34](#)
- logm
  - qpp, [35](#)
- MATRIX\_NOT\_CVECTOR
  - qpp::Exception, [69](#)
- MATRIX\_NOT\_RVECTOR
  - qpp::Exception, [69](#)
- MATRIX\_NOT\_SQUARE
  - qpp::Exception, [69](#)
- MATRIX\_NOT\_SQUARE\_OR\_CVECTOR
  - qpp::Exception, [69](#)
- MATRIX\_NOT\_SQUARE\_OR\_RVECTOR
  - qpp::Exception, [69](#)
- MATRIX\_NOT\_SQUARE\_OR\_VECTOR
  - qpp::Exception, [69](#)
- MATRIX\_NOT\_VECTOR
  - qpp, [69](#)

- qpp::Exception, 69
- mket
  - qpp, 35, 36
- multiidx2n
  - qpp, 37
- n2multiidx
  - qpp, 37
- NOT\_BIPARTITE
  - qpp::Exception, 69
- NOT\_QUBIT\_GATE
  - qpp::Exception, 69
- NOT\_QUBIT\_SUBSYS
  - qpp::Exception, 69
- norm
  - qpp, 38
- OUT\_OF\_RANGE
  - qpp::Exception, 69
- PERM\_INVALID
  - qpp::Exception, 69
- powm
  - qpp, 39
- prj
  - qpp, 39
- ptrace
  - qpp, 40
- ptrace1
  - qpp, 41
- ptrace2
  - qpp, 42
- ptranspose
  - qpp, 43
- qmutualinfo
  - qpp, 44
- qpp, 9
  - absm, 14
  - adjoint, 15
  - anticomm, 15
  - channel, 16
  - choi, 17
  - choi2kraus, 17
  - comm, 18
  - compperm, 18
  - conjugate, 20
  - cosm, 20
  - cwise, 21
  - det, 21
  - disp, 22
  - displn, 22, 23
  - entanglement, 24
  - evals, 24
  - evects, 25
  - expandout, 25
  - expm, 26
  - funm, 27
  - gconcurrency, 27
  - grams, 28, 29
  - gt, 60
  - hevals, 29
  - hevects, 30
  - inverse, 30
  - invperm, 31
  - kron, 31–33
  - kronpow, 33
  - load, 34
  - logdet, 34
  - logm, 35
  - mket, 35, 36
  - multiidx2n, 37
  - n2multiidx, 37
  - norm, 38
  - powm, 39
  - prj, 39
  - ptrace, 40
  - ptrace1, 41
  - ptrace2, 42
  - ptranspose, 43
  - qmutualinfo, 44
  - rand, 45, 46
  - randint, 46
  - randket, 46
  - randkraus, 47
  - randn, 47, 48
  - randperm, 48
  - randrho, 48
  - rdevs, 60
  - renyi, 49
  - reshape, 50
  - save, 50
  - schmidtcoeff, 51
  - schmidtprob, 52
  - shannon, 53
  - sinm, 53
  - spectralpowm, 54
  - sqrtn, 54
  - st, 60
  - sum, 56
  - super, 56
  - syspermute, 57
  - trace, 58
  - transpose, 59
  - tsallis, 59
- qpp::Exception
  - CUSTOM\_EXCEPTION, 69
  - DIMS\_INVALID, 69
  - DIMS\_MISMATCH\_CVECTOR, 69
  - DIMS\_MISMATCH\_MATRIX, 69
  - DIMS\_MISMATCH\_RVECTOR, 69
  - DIMS\_MISMATCH\_VECTOR, 69
  - DIMS\_NOT\_EQUAL, 69
  - MATRIX\_NOT\_CVECTOR, 69
  - MATRIX\_NOT\_RVECTOR, 69
  - MATRIX\_NOT\_SQUARE, 69
  - MATRIX\_NOT\_SQUARE\_OR\_CVECTOR, 69

MATRIX\_NOT\_SQUARE\_OR\_RVECTOR, 69  
MATRIX\_NOT\_SQUARE\_OR\_VECTOR, 69  
MATRIX\_NOT\_VECTOR, 69  
NOT\_BIPARTITE, 69  
NOT\_QUBIT\_GATE, 69  
NOT\_QUBIT\_SUBSYS, 69  
OUT\_OF\_RANGE, 69  
PERM\_INVALID, 69  
SUBSYS\_MISMATCH\_DIMS, 69  
TYPE\_MISMATCH, 69  
UNDEFINED\_TYPE, 69  
UNKNOWN\_EXCEPTION, 69  
ZERO\_SIZE, 69

rand  
    qpp, 45, 46  
randint  
    qpp, 46  
randket  
    qpp, 46  
randkraus  
    qpp, 47  
randn  
    qpp, 47, 48  
randperm  
    qpp, 48  
randrho  
    qpp, 48  
rdevs  
    qpp, 60  
renyi  
    qpp, 49  
reshape  
    qpp, 50

SUBSYS\_MISMATCH\_DIMS  
    qpp::Exception, 69  
save  
    qpp, 50  
schmidtcoeff  
    qpp, 51  
schmidtprob  
    qpp, 52  
shannon  
    qpp, 53  
sinm  
    qpp, 53  
spectralpowm  
    qpp, 54  
sqrtm  
    qpp, 54  
st  
    qpp, 60  
sum  
    qpp, 56  
super  
    qpp, 56  
syspermute  
    qpp, 57

TYPE\_MISMATCH  
    qpp::Exception, 69  
trace  
    qpp, 58  
transpose  
    qpp, 59  
tsallis  
    qpp, 59

UNDEFINED\_TYPE  
    qpp::Exception, 69  
UNKNOWN\_EXCEPTION  
    qpp::Exception, 69

ZERO\_SIZE  
    qpp::Exception, 69