quantum++ 0.1

Generated by Doxygen 1.8.7

Fri Oct 24 2014 01:31:37

Contents

1	qua	ntum++	- A C++1	1 qı	uantı	ım c	om	put	ing	lib	rar	y									1
2	Nam	nespace	Index																		5
	2.1	Names	space List	٠.									 	 	 	 			 		5
3	Hier	archica	l Index																		7
	3.1	Class	Hierarchy										 	 	 	 			 		7
4	Clas	s Index																			9
	4.1	Class	List										 	 	 	 			 		9
5	File	Index																			11
	5.1	File Lis	st										 		 	 			 		11
6	Nam	nespace	Docume	nta	tion																13
	6.1	qpp Na	amespace	Re	feren	ice .							 	 	 	 			 		13
		6.1.1	Typedef	Do	cume	ntati	on .						 	 	 	 			 		19
			6.1.1.1	bı	ra .								 	 	 	 			 		19
			6.1.1.2	CI	mat								 	 	 	 			 		19
			6.1.1.3	C	olx .								 	 	 	 			 		19
			6.1.1.4	dı	mat								 	 	 	 			 		19
			6.1.1.5	D	ynMa	at							 	 	 	 			 		19
			6.1.1.6	ke	et .								 	 	 	 			 		19
		6.1.2	Function	n Do	cum	entat	ion						 	 	 	 			 		20
			6.1.2.1	al	bsm								 	 	 	 			 		20
			6.1.2.2	a	djoint								 	 	 	 			 		21
			6.1.2.3	aı	ntico	nm .							 	 	 	 			 		22
			6.1.2.4	cl	hann	el							 	 	 	 			 		22
			6.1.2.5	cl	hann	el							 	 	 	 			 		23
			6.1.2.6	cl	hoi .								 	 	 	 			 		24
			6.1.2.7		hoi2k																24
			6.1.2.8	C	omm								 	 	 	 			 		25
			6120	0.	omor	orm															26

iv CONTENTS

6.1.2.10	conjugate	27
6.1.2.11	cosm	27
6.1.2.12	cwise	28
6.1.2.13	det	28
6.1.2.14	disp	29
6.1.2.15	disp	29
6.1.2.16	disp	29
6.1.2.17	disp	29
6.1.2.18	displn	30
6.1.2.19	displn	30
6.1.2.20	displn	30
6.1.2.21	displn	31
6.1.2.22	entanglement	31
6.1.2.23	evals	32
6.1.2.24	evects	33
6.1.2.25	expandout	33
6.1.2.26	expm	34
6.1.2.27	funm	35
6.1.2.28	gconcurrence	35
6.1.2.29	grams	36
6.1.2.30	grams	37
6.1.2.31	grams	37
6.1.2.32	hevals	38
6.1.2.33	hevects	38
6.1.2.34	inverse	39
6.1.2.35	invperm	39
6.1.2.36	kron	40
6.1.2.37	kron	40
6.1.2.38	kron	41
6.1.2.39	kron	41
6.1.2.40	kronpow	42
6.1.2.41	load	42
6.1.2.42	loadMATLABmatrix	42
6.1.2.43	loadMATLABmatrix	43
6.1.2.44	loadMATLABmatrix	43
6.1.2.45	logdet	43
6.1.2.46	logm	43
6.1.2.47	mket	44
6.1.2.48	mket	44
6.1.2.49	mket	45

CONTENTS

6.1.2.50	multiidx2n	45
6.1.2.51	n2multiidx	46
6.1.2.52	norm	46
6.1.2.53	omega	47
6.1.2.54	operator""""_i	47
6.1.2.55	operator""""_i	47
6.1.2.56	powm	47
6.1.2.57	prj	48
6.1.2.58	ptrace	49
6.1.2.59	ptrace1	50
6.1.2.60	ptrace2	51
6.1.2.61	ptranspose	52
6.1.2.62	qmutualinfo	53
6.1.2.63	rand	54
6.1.2.64	rand	54
6.1.2.65	rand	55
6.1.2.66	rand	55
6.1.2.67	randH	55
6.1.2.68	randint	55
6.1.2.69	randket	56
6.1.2.70	randkraus	56
6.1.2.71	randn	56
6.1.2.72	randn	56
6.1.2.73	randn	57
6.1.2.74	randn	57
6.1.2.75	randperm	57
6.1.2.76	randrho	58
6.1.2.77	randU	58
6.1.2.78	randV	58
6.1.2.79	renyi	58
6.1.2.80	renyi_inf	59
6.1.2.81	reshape	59
6.1.2.82	save	61
6.1.2.83	saveMATLABmatrix	61
6.1.2.84	saveMATLABmatrix	61
6.1.2.85	saveMATLABmatrix	62
6.1.2.86	schmidtcoeff	62
6.1.2.87	schmidtprob	63
6.1.2.88	schmidtU	63
6.1.2.89	schmidtV	64

vi CONTENTS

		6.1.2.90	shannon	65
		6.1.2.91	sinm	66
		6.1.2.92	spectralpowm	66
		6.1.2.93	sqrtm	67
		6.1.2.94	sum	68
		6.1.2.95	super	68
		6.1.2.96	syspermute	69
		6.1.2.97	trace	70
		6.1.2.98	transpose	71
		6.1.2.99	tsallis	71
	6.1.3	Variable I	Documentation	72
		6.1.3.1	chop	72
		6.1.3.2	ee	72
		6.1.3.3	eps	72
		6.1.3.4	gt	72
		6.1.3.5	maxn	73
		6.1.3.6	pi	73
		6.1.3.7	rdevs	73
		6.1.3.8	st	73
6.2	qpp::in	ternal Nan	nespace Reference	73
	6.2.1	Detailed	Description	74
	6.2.2	Function	Documentation	74
		6.2.2.1	_check_col_vector	74
		6.2.2.2	_check_dims	74
		6.2.2.3	_check_dims_match_cvect	74
		6.2.2.4	_check_dims_match_mat	74
		6.2.2.5	_check_dims_match_rvect	74
		6.2.2.6	_check_eq_dims	74
		6.2.2.7	_check_nonzero_size	74
		6.2.2.8	_check_perm	74
		6.2.2.9	_check_row_vector	74
		6.2.2.10	_check_square_mat	74
		6.2.2.11	_check_subsys_match_dims	74
		6.2.2.12	_check_vector	74
		6.2.2.13	_kron2	74
		6.2.2.14	_multiidx2n	75
		6.2.2.15	_n2multiidx	75
		6.2.2.16	variadic_vector_emplace	75
		6.2.2.17	variadic_vector_emplace	75

CONTENTS vii

7	Clas	s Docu	nentation	77
	7.1	qpp::D	screteDistribution Class Reference	77
		7.1.1	Constructor & Destructor Documentation	77
			7.1.1.1 Discrete Distribution	77
			7.1.1.2 DiscreteDistribution	77
			7.1.1.3 DiscreteDistribution	77
		7.1.2	Member Function Documentation	77
			7.1.2.1 probabilities	77
			7.1.2.2 sample	78
		7.1.3	Member Data Documentation	78
			7.1.3.1 _d	78
	7.2	qpp::D	screteDistributionAbsSquare Class Reference	78
		7.2.1	Constructor & Destructor Documentation	79
			7.2.1.1 DiscreteDistributionAbsSquare	79
			7.2.1.2 DiscreteDistributionAbsSquare	79
			7.2.1.3 DiscreteDistributionAbsSquare	79
			7.2.1.4 DiscreteDistributionAbsSquare	79
		7.2.2	Member Function Documentation	79
			7.2.2.1 cplx2weights	79
			7.2.2.2 probabilities	79
			7.2.2.3 sample	79
		7.2.3	Member Data Documentation	79
			7.2.3.1 _d	79
	7.3	qpp::E	cception Class Reference	79
		7.3.1	Member Enumeration Documentation	31
			7.3.1.1 Type	31
		7.3.2	Constructor & Destructor Documentation	32
			7.3.2.1 Exception	32
			7.3.2.2 Exception	32
		7.3.3	Member Function Documentation	32
			7.3.3.1 _construct_exception_msg	32
			7.3.3.2 what	32
		7.3.4	Member Data Documentation	32
			7.3.4.1 _custom	32
			7.3.4.2 _msg	32
			7.3.4.3 _type	32
			7.3.4.4 _where	32
	7.4	qpp::G	ates Class Reference	32
		7.4.1	Constructor & Destructor Documentation	34
			7.4.1.1 Gates	34

viii CONTENTS

	7.4.2	Member Function Documentation
		7.4.2.1 apply
		7.4.2.2 applyCTRL
		7.4.2.3 CTRL
		7.4.2.4 Fd
		7.4.2.5 ld
		7.4.2.6 Rn
		7.4.2.7 Xd
		7.4.2.8 Zd
	7.4.3	Friends And Related Function Documentation
		7.4.3.1 Singleton < const Gates >
	7.4.4	Member Data Documentation
		7.4.4.1 CNOTab
		7.4.4.2 CNOTba
		7.4.4.3 CZ
		7.4.4.4 FRED
		7.4.4.5 H
		7.4.4.6 ld2
		7.4.4.7 S
		7.4.4.8 SWAP
		7.4.4.9 T
		7.4.4.10 TOF
		7.4.4.11 X
		7.4.4.12 Y
		7.4.4.13 Z
7.5	qpp::No	ormalDistribution Class Reference
	7.5.1	Constructor & Destructor Documentation
		7.5.1.1 NormalDistribution
	7.5.2	Member Function Documentation
		7.5.2.1 sample
	7.5.3	Member Data Documentation
		7.5.3.1 _d
7.6	qpp::Qı	udit Class Reference
	7.6.1	Constructor & Destructor Documentation
		7.6.1.1 Qudit
	7.6.2	Member Function Documentation
		7.6.2.1 getD
		7.6.2.2 getRho
		7.6.2.3 measure
		7.6.2.4 measure

CONTENTS

	7.6.3	Member Data Documentation
		7.6.3.1 _D
		7.6.3.2 _rho
7.7	qpp::R	andomDevices Class Reference
	7.7.1	Constructor & Destructor Documentation
		7.7.1.1 RandomDevices
	7.7.2	Friends And Related Function Documentation
		7.7.2.1 Singleton < Random Devices >
	7.7.3	Member Data Documentation
		7.7.3.1 _rd
		7.7.3.2 _rng
7.8	qpp::S	ingleton < T > Class Template Reference
	7.8.1	Constructor & Destructor Documentation
		7.8.1.1 Singleton
		7.8.1.2 ~Singleton
		7.8.1.3 Singleton
	7.8.2	Member Function Documentation
		7.8.2.1 get_instance
		7.8.2.2 operator=
7.9	qpp::S	tates Class Reference
	7.9.1	Constructor & Destructor Documentation
		7.9.1.1 States
	7.9.2	Friends And Related Function Documentation
		7.9.2.1 Singleton < const States >
	7.9.3	Member Data Documentation
		7.9.3.1 b00
		7.9.3.2 b01
		7.9.3.3 b10
		7.9.3.4 b11
		7.9.3.5 GHZ
		7.9.3.6 pb00
		7.9.3.7 pb01
		7.9.3.8 pb10
		7.9.3.9 pb11
		7.9.3.10 pGHZ
		7.9.3.11 pW
		7.9.3.12 px0
		7.9.3.13 px1
		7.9.3.14 py0
		7.9.3.15 py1

CONTENTS

		7.9.3.16 pz0
		7.9.3.17 pz1
		7.9.3.18 W
		7.9.3.19 x0
		7.9.3.20 x1
		7.9.3.21 y0
		7.9.3.22 y1
		7.9.3.23 z0
		7.9.3.24 z1
7.10	qpp::Ti	mer Class Reference
	7.10.1	Constructor & Destructor Documentation
		7.10.1.1 Timer
	7.10.2	Member Function Documentation
		7.10.2.1 seconds
		7.10.2.2 tic
		7.10.2.3 toc
	7.10.3	Friends And Related Function Documentation
		7.10.3.1 operator<< 96
	7.10.4	Member Data Documentation
		7.10.4.1 _end
		7.10.4.2 _start
7.11	qpp::Ur	niformIntDistribution Class Reference
	7.11.1	Constructor & Destructor Documentation
		7.11.1.1 UniformIntDistribution
	7.11.2	Member Function Documentation
		7.11.2.1 sample
	7.11.3	Member Data Documentation
		7.11.3.1 _d
7.12		niformRealDistribution Class Reference
	7.12.1	Constructor & Destructor Documentation
		7.12.1.1 UniformRealDistribution
	7.12.2	Member Function Documentation
		7.12.2.1 sample
	7.12.3	Member Data Documentation
		7.12.3.1 _d
File	Docume	entation 99
8.1	include	/channels.h File Reference
8.2	include	/classes/exception.h File Reference
8.3	include	/classes/gates.h File Reference

8

CONTENTS xi

	8.4	include	/classes/q	udit.h File F	Reference			 		 		 . 101
	8.5		•	andevs.h Fi								
	8.6											
•	0.0			ingleton.h F								
		8.6.1	Macro De	efinition Doo								
			8.6.1.1	CLASS_C	ONST_SI	NGLET	TON	 	 	 	 	 . 102
			8.6.1.2	CLASS_S	INGLETO	Ν		 	 	 	 	 . 102
	8.7	include	e/classes/s	tat.h File Re	eference .			 	 	 	 	 . 103
	8.8	include	e/classes/s	tates.h File	Reference			 	 	 	 	 . 103
	8.9	include	e/classes/ti	mer.h File F	Reference			 	 	 	 	 . 104
	8.10	include	constants	.h File Refe	erence			 	 	 	 	 . 104
	8.11	include	e/entangler	nent.h File	Reference			 	 	 	 	 . 105
	8.12	include	e/entropies	.h File Refe	rence			 	 	 	 	 . 106
	8.13	include	/functions.	h File Refe	rence			 	 	 	 	 . 107
	8.14	include	/internal.h	File Refere	ence			 	 	 	 	 . 110
	8.15	include	/io.h File F	Reference				 	 	 	 	 . 111
	8.16	include	e/matlab.h	File Referei	nce			 	 	 	 	 . 112
	8.17	include	e/qpp.h File	Reference				 	 	 	 	 . 113
	8.18	include	e/random.h	File Refere	ence			 	 	 	 	 . 114
	8.19	include	/types.h F	ile Referend	ce			 	 	 	 	 . 115
Inde	ex											116

Chapter 1

quantum++ - A C++11 quantum computing library

Version

0.1

Author

Vlad Gheorghiu

Date

24 October 2014

A simple example:

```
#include "qpp.h"
//#include "matlab.h" // support for MATLAB
using namespace std;
using namespace qpp;
cplx pow3(const cplx& z) // a test function
    return std::pow(z, 3);
int main()
    cout << "Starting qpp..." << endl;</pre>
     // output format
     //cout << std::scientific;</pre>
    cout << std::fixed; // use fixed format for nice formatting</pre>
    cout << std::setprecision(4); // only for fixed or scientific modes</pre>
     // TESTING
     // testing channel and Gates::apply
     cout << endl << "Testing channel(...) and Gates::apply(...)." << endl;</pre>
     cmat rho = randrho(16);
cmat K = kron(gt.Id2, gt.X, gt.Y, gt.Z);
    vector<std::size_t> p = randperm(4); // permutation
cout << "Permutation: ";</pre>
     displn(p, ", ");
    vector<std::size_t> invp = invperm(p); // inverse permutation
cout << "Inverse permutation: ";
displn(invp, ", ");</pre>
     cmat r1 = channel(rho, { K }, p, { 2, 2, 2, 2 });
     cmat r2 = syspermute(channel(syspermute(rho, p, { 2, 2, 2 }), { K },
      { 0,
    1, 2, 3 }, { 2, 2, 2, 2 }), invp, { 2, 2, 2, 2 }); cout << norm(r1 - r2) << endl << endl;
     r1 = gt.apply(rho, K, p, { 2, 2, 2, 2 });
     gt.apply(syspermute(rho, p, { 2, 2, 2, 2 }), K, { 0, 1, 2, 3 }, { 2, 2, 2, 2 }), invp, { 2, 2, 2, 2 });
cout << norm(r1 - r2) << endl << endl;
```

```
displn(channel(prj(mket( { 0, 1 })), { gt.CNOTab }, { 1, 0 }, { 2, 2 }));
displn(gt.apply(mket( { 0, 0 }), gt.CNOTab, { 0, 1 }, { 2, 2 }));
// quantum teleportation
cout << endl << "Qudit teleportation." << endl;</pre>
ket psi = randket(2); // a random state;
cout << "|psi><psi|:" << endl;</pre>
ket psiout = telecircuit * psiin; // output state before measurement
// measure Alice's qubits, measurement results are 1 0
psiout = kron(prj(st.z1), prj(st.z0), gt.Id2) * psiout;
// apply correction
psiout = expandout(powm(gt.Z, 1) * powm(gt.X, 0), { 2 }, { 2, 2, 2 })
           psiout;
// not necessary to normalize, prj() takes care of it below
cmat rhoout = ptrace(prj(psiout), { 0, 1 }, { 2, 2, 2 });
cout << endl << "Teleported state:" << endl;</pre>
displn(rhoout);
cout << "Difference in norm: " << norm(prj(psi) - rhoout) << endl;</pre>
// qudit measurements
cout << endl << "Qudit measurements." << endl;</pre>
cout << "Initially in state |0><0|." << endl;</pre>
ket zd0(3);
zd0 << 1, 0, 0;
Qudit q(prj(zd0));
cout << "Measuring Z operator non-destructively. Results:" << endl;</pre>
cout << q.measure() << endl;</pre>
cout << q.measure() << endl;</pre>
cout << q.measure() << endl;
cout << q.measureing X operator non-destructively. Results:" << endl;
cout << q.measure(gt.Xd(3)) << endl;
cout << q.measure(gt.Xd(3)) << endl;</pre>
cout << q.measure(gt.Xd(3)) << endl;</pre>
// von Neumann projective measurement
cout << "Measuring X operator destructively (collapse). Results:" << endl;
cout << q.measure(gt.Xd(3), true) << endl;
cout << q.measure(gt.Xd(3)) << endl;</pre>
cout << q.measure(gt.Xd(3)) << endl;
cout << g.measure(gt.Xd(3)) << endl;
cout << "Finally measuring Z operator destructively. Results:" << endl;</pre>
cout << q.measure(true) << endl;</pre>
cout << q.measure() << endl;</pre>
cout << q.measure() << endl;
cout << "Final state of qudit:" << endl;</pre>
displn(q.getRho());
// Bell state generator
cout << endl << "Bell state generator: " << endl;</pre>
cmat circuit;
circuit = gt.CTRL(gt.X, { 0 }, { 1 }, 2) * expandout(gt.
    H, 0, { 2, 2 });
cmat input = kron(st.z0, st.z0);
cmat output = circuit * input;
cout << "Circuit matrix representation: " << endl;</pre>
displn(circuit);
cout << endl << "Output (|Bell_0> state) of the circuit on |00>: " << endl;</pre>
displn(output);
// 3-qubit repetion code
cout << endl << "3-qubit repetition code: " << endl;</pre>
cmat rep;
rep = gt.CTRL(gt.X, { 0 }, { 2 }, 3) * gt.CTRL(gt.X, { 0 }, { 1 }, 3);
input = kron(st.z1, st.z0, st.z0);
output = rep * input;
cout << "Circuit acting on |000> produces |111>. Check: " << endl;
displn(output);
// functor test
cout << endl << "Functor z^3 acting on:" << endl;</pre>
cmat a(2, 2);
a << 1, 2, 3, 4;</pre>
displn(a);
cout << "Result (with lambda):" << endl;</pre>
// functor z^3 componentwise, specify OutputScalar and Derived for lambdas  \frac{\text{displn}(\text{cwise} < \text{cplx, cmat} > (a, [](\text{const cplx\& z}) -> \text{cplx} } ) + cplx 
    return z*z*z;}));
cout << "Result (with proper function):" << endl;</pre>
// automatic type deduction for proper functions
displn(cwise(a, &pow3));
// Gram-Schmidt
cout << endl << "Gram-Schmidt on matrix:" << endl;</pre>
```

```
cmat A(3, 3);
A << 1, 1, 0, 0, 2, 0, 0, 0;
displn(A);
cmat Ags = grams(A);
cout << endl << "Result:" << endl:
displn(Aqs):
cout << endl << "Projector is:" << endl;
displn(Ags * adjoint(Ags));
// spectral decomposition test
cout << endl << "Spectral decomposition tests." << endl;</pre>
std::size_t D = 4;
cmat rH = randH(D);
dmat evalsH = hevals(rH);
cmat evectsH = hevects(rH);
cmat spec = cmat::Zero(D, D);
for (std::size_t i = 0; i < D; i++)
    spec += evalsH(i) * prj((cmat) evectsH.col(i));
cout << "Original matrix: " << endl;</pre>
displn(rH);
cout << endl << "Reconstructed from spectral decomposition: " << endl;</pre>
displn(spec);
cout << "Difference in norm: " << norm(spec - rH) << endl;</pre>
// channel tests
cout << endl << "Channel tests." << endl;</pre>
std::size\_t nk = 10, d = 2; // nk Kraus on d-dimensional system
std::vector<cmat> Ks = randkraus(nk, d);
cmat rho_in = randrho(d); // input state
cmat rho_out = channel(rho_in, Ks); // output state
cout << "Computing its Choi matrix..." << endl;</pre>
cmat choim = choi(Ks);
cout << "Choi matrix:" << endl;</pre>
displn(choim);
cout << endl << "The eigenvalues of the Choi matrix are: " << endl;</pre>
displn(transpose(hevals(choim)));
cout << endl << "Their sum is: " << sum(hevals(choim)) << endl;</pre>
std::vector<cmat> Kperps = choi2kraus(choim);
cout << endl << "The Kraus rank of the channel is: " << Kperps.size()</pre>
         << endl;
cmat rho_out1 = channel(rho_in, Kperps);
cout << endl << "Difference in norm on output states: "</pre>
         << norm(rho_out1 - rho_out) << endl;
cout << endl << "Superoperator matrix:" << endl;</pre>
cmat smat = super(Ks);
displn(smat);
cout << endl << "The eigenvalues of the superoperator matrix are: " << endl;</pre>
cmat evalsupop = evals(smat);
displn(transpose(evalsupop));
cout << endl << "Their absolute values are: " << endl;
for (std::size_t i = 0; i < (std::size_t) evalsupop.size(); i++)
    cout << std::abs((cplx) evalsupop(i)) << " ";</pre>
cout << endl << endl << "Diference in norm for superoperator action: ";</pre>
cmat rho_out2 = transpose(
         (cmat) reshape(smat * reshape(transpose(rho_in), d * d, 1), d, d));
cout << norm(rho_out - rho_out2) << endl;</pre>
// statistics tests
cout << endl << "Statistics tests." << endl;</pre>
std::vector<cplx> ampl = { 1. + 1_i, 1. - 1_i };
cmat va(1, 4);
va << 0.1, 1, 1. + 1_i, 1. + 2_i;
DiscreteDistributionAbsSquare dc(va);
cout << "The probabilities are: ";
displn(dc.probabilities(), ", ", "{", "}");</pre>
// // TIMING tests
cout << endl << "Timing tests..." << endl;
std::size_t n = 12; // number of qubits
std::size_t N = std::pow(2, n);
vector<std::size_t> dims(n, 2); // local dimensions
cout << "n = " << n << " qubits, matrix size " << N << " x " << N << "."
         << endl;
// matrix initialization
cout << endl << "Matrix initialization timing." << endl;</pre>
// start the timer, automatic tic() in the constructor
Timer t, total;
cmat randcmat = cmat::Random(N, N);
t.toc(); // read the time
cout << "Took " << t << " seconds." << endl;</pre>
// lazv matrix product
```

```
cout << endl << "Lazy matrix product timing." << endl;</pre>
auto lazyprod = randcmat * randcmat; // lazyprod has type GenMatProduct
t.toc(); // read the time
cout << "Took " << t << " seconds." << endl;
// ptrace1 timing
cout << endl << "ptrace1 timing." << endl;</pre>
t.tic(); // reset the chronometer
// trace away half of the qubits
ptrace1(randcmat,
{ (std::size_t) std::sqrt(N), (std::size_t) std::sqrt(N) }); t.toc(); // read the time cout << "Took " << t << " seconds." << endl;
// ptrace2 timing
cout << endl << "ptrace2 timing." << endl;</pre>
t.tic(); // reset the chronometer // trace away half of the qubits
ptrace2(randcmat,
{ (std::size_t) std::sqrt(N), (std::size_t) std::sqrt(N) });
t.toc(); // read the time
cout << "Took " << t << " seconds." << endl;
// ptrace
cout << endl << "ptrace timing." << endl;</pre>
vector<std::size_t> subsys_ptrace = { 0 };
cout << "Subsytem(s): ";
displn(subsys_ptrace, ", ");</pre>
t.tic();
ptrace(randcmat, subsys ptrace, dims);
t.toc();
cout << "Took " << t << " seconds." << endl;
// ptranspose
cout << endl << "ptranspose timing." << endl;</pre>
vector<std::size_t> subsys_ptranspose; // partially transpose n-1 subsystems
for (std::size_t i = 0; i < n - 1; i++)</pre>
     subsys_ptranspose.push_back(i);
cout << "Subsytem(s): ";
displn(subsys_ptranspose, ", ");
t.tic();
ptranspose (randcmat, subsys ptranspose, dims);
t.toc();
cout << "Took " << t << " seconds." << endl;
// syspermute
cout << endl << "syspermute timing." << endl;</pre>
fout < end < syspenmete timing. < end;
vector(std::size_t) perm; // left-shift all subsystems by 1
for (std::size_t i = 0; i < n; i++)
    perm.push_back((i + 1) % n);</pre>
cout << "Subsytem(s): ";</pre>
displn(perm, ", ");
t.tic();
syspermute(randcmat, perm, dims);
t.toc();
cout << "Took " << t << " seconds." << endl;
      // matrix product
cout << endl << "Matrix product timing." << endl;</pre>
11
      t.tic(); // reset the chronometer
11
//
      cmat prodmat = randcmat * randcmat; // explicit cmat now
      t.toc(); // read the time
cout << "Took " << t << " seconds." << endl;
// END TIMING
total.toc(); // read the total running time
cout << endl << "Total time: " << total.seconds() << " seconds.";
cout << endl << "Exiting qpp..." << endl;</pre>
```

Chapter 2

Namespace Index

	2.1	Names	pace	List
--	-----	--------------	------	------

Here is	a list of a	ll nar	nesp	ace	s wi	ith b	rie	f de	esc	rip	tio	ns:											
qpp													 										13
qpp	::internal												 										73

6 Namespace Index

Chapter 3

Hierarchical Index

3.1 Class Hierarchy

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

qpp::DiscreteDistribution	77
qpp::DiscreteDistributionAbsSquare	78
exception	
qpp::Exception	79
qpp::NormalDistribution	88
qpp::Qudit	89
$qpp \text{::Singleton} < T > \ \ \dots \dots$	92
qpp::Gates	82
qpp::RandomDevices	91
qpp::Singleton< const Gates >	92
qpp::Singleton < const States >	92
qpp::States	93
qpp::Singleton < RandomDevices >	92
qpp::Timer	96
qpp::UniformIntDistribution	97
gpp::UniformRealDistribution	97

8 **Hierarchical Index**

Chapter 4

Class Index

4.1 Class List

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

p::DiscreteDistribution	. 77
p::DiscreteDistributionAbsSquare	. 78
p::Exception	. 79
p::Gates	. 82
p::NormalDistribution	. 88
p::Qudit	. 89
p::RandomDevices	. 91
p::Singleton $<$ T $>$ \dots	. 92
p::States	. 93
p::Timer	. 96
p::UniformIntDistribution	. 97
p::UniformRealDistribution	. 97

10 Class Index

Chapter 5

File Index

5.1 File List

Here is a list of all files with brief descriptions:

include/channels.h
include/constants.h
include/entanglement.h
include/entropies.h
include/functions.h
include/internal.h
include/io.h
include/matlab.h
include/qpp.h
include/random.h
include/types.h
include/classes/exception.h
include/classes/gates.h
include/classes/qudit.h
include/classes/randevs.h
include/classes/singleton.h
include/classes/stat.h
include/classes/states.h
include/classes/timer.h

12 File Index

Chapter 6

Namespace Documentation

6.1 qpp Namespace Reference

Namespaces

· internal

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

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.
```

Functions

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

      Superoperator matrix representation.

    cmat choi (const std::vector < cmat > &Ks)

      Choi matrix representation.

    std::vector< cmat > choi2kraus (const cmat &A)

      Extracts orthogonal Kraus operators from Choi matrix.
 \bullet \ \ \mathsf{template} \mathord{<} \mathsf{typename} \ \mathsf{Derived} >
  cmat channel (const Eigen::MatrixBase< Derived > &rho, const std::vector< cmat > &Ks)
     Applies the channel specified by the set of Kraus operators Ks to the density matrix rho.

    template<typename Derived >

  cmat channel (const Eigen::MatrixBase < Derived > &rho, const std::vector < cmat > &Ks, const std::vector <
  std::size_t > &subsys, const std::vector< std::size_t > &dims)
      Applies the channel specified by the set of Kraus operators Ks to the part of the density matrix rho specified by
      subsys.

    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)

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

      D-th root of unity.
• template<typename Derived >
  cmat schmidtcoeff (const Eigen::MatrixBase< Derived > &A, const std::vector< std::size_t > &dims)
      Schmidt coefficients of the bi-partite pure state A.

    template<typename Derived >

  cmat schmidtU (const Eigen::MatrixBase< Derived > &A, const std::vector< std::size_t > &dims)
      Schmidt basis on Alice's side.

    template<typename Derived >

  cmat schmidtV (const Eigen::MatrixBase< Derived > &A, const std::vector< std::size t > &dims)
      Schmidt basis on Bob's side.

    template<typename Derived >

  cmat schmidtprob (const Eigen::MatrixBase< Derived > &A, const std::vector< std::size_t > &dims)
      Schmidt probabilities of the bi-partite pure state A.

    template<typename Derived >

  double entanglement (const Eigen::MatrixBase< Derived > &A, const std::vector< std::size_t > &dims)
      Entanglement of the bi-partite pure state A.

    template<typename Derived >

  double gconcurrence (const Eigen::MatrixBase< Derived > &A)
      G-concurrence of the bi-partite pure state A.
• template<typename Derived >
  double shannon (const Eigen::MatrixBase< Derived > &A)
      Shannon/von-Neumann entropy of the probability distribution/density matrix A.

    template<typename Derived >

  double renyi (const double alpha, const Eigen::MatrixBase< Derived > &A)
      Renyi- \alpha entropy of the probability distribution/density matrix A.

    template<typename Derived >

  double renyi_inf (const Eigen::MatrixBase< Derived > &A)
      Renyi- ∞ entropy (min entropy) of the probability distribution/density matrix A.

    template<typename Derived >

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

Tsallis- α entropy of the probability distribution/density matrix A.

```
• template<typename Derived >
  double qmutualinfo (const Eigen::MatrixBase< Derived > &A, const std::vector< std::size t > &subsys,
  const std::vector< std::size_t > &dims)
     Quantum mutual information between 2 subsystems of a composite system.
• template<typename Derived >
  DynMat< typename Derived::Scalar > transpose (const Eigen::MatrixBase< Derived > &A)
     Transpose.

    template<typename Derived >

  DynMat< typename Derived::Scalar > conjugate (const Eigen::MatrixBase< Derived > &A)
     Complex conjugate.
• template<typename Derived >
  DynMat< typename Derived::Scalar > adjoint (const Eigen::MatrixBase< Derived > &A)

    template<typename Derived >

  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 >

  cmat evals (const Eigen::MatrixBase< Derived > &A)
     Eigenvalues.

    template<typename Derived >

  cmat evects (const Eigen::MatrixBase< Derived > &A)
     Eigenvectors.

    template<typename Derived >

  dmat hevals (const Eigen::MatrixBase< Derived > &A)
     Hermitian eigenvalues.
• template<typename Derived >
  cmat hevects (const Eigen::MatrixBase< Derived > &A)
     Hermitian eigenvectors.

    template<typename Derived >

  cmat funm (const Eigen::MatrixBase< Derived > &A, cplx(*f)(const cplx &))
     Functional calculus f(A)

    template<typename Derived >

  cmat sqrtm (const Eigen::MatrixBase< Derived > &A)
     Matrix square root.

    template<typename Derived >

  cmat absm (const Eigen::MatrixBase< Derived > &A)
```

Matrix absolut value.

Partial trace.

```
• template<typename Derived >
  cmat expm (const Eigen::MatrixBase< Derived > &A)
     Matrix exponential.

    template<typename Derived >

  cmat logm (const Eigen::MatrixBase< Derived > &A)
     Matrix logarithm.
• template<typename Derived >
  cmat sinm (const Eigen::MatrixBase< Derived > &A)
     Matrix sin.
• template<typename Derived >
  cmat cosm (const Eigen::MatrixBase< Derived > &A)
     Matrix cos.

    template<typename Derived >

  cmat spectralpowm (const Eigen::MatrixBase< Derived > &A, const cplx z)
     Matrix power.
• template<typename Derived >
  DynMat< typename Derived::Scalar > powm (const Eigen::MatrixBase< Derived > &A, std::size t n)
- template<typename OutputScalar , typename Derived >
  DynMat< OutputScalar > cwise (const Eigen::MatrixBase< Derived > &A, OutputScalar(*f)(const typename
  Derived::Scalar &))
     Functor.
• template<typename T >
  DynMat< typename T::Scalar > kron (const T &head)
     Kronecker product (variadic overload)
• template<typename T , typename... Args>
  DynMat< typename T::Scalar > kron (const T &head, const Args &...tail)
     Kronecker product (variadic overload)

    template<typename Derived >

  DynMat< typename Derived::Scalar > kron (const std::vector< Derived > &As)
     Kronecker product (std::vector overload)

    template<typename Derived >

  DynMat< typename Derived::Scalar > kron (const std::initializer_list< Derived > &As)
     Kronecker product (std::initializer_list overload)

    template<typename Derived >

  DynMat< typename Derived::Scalar > kronpow (const Eigen::MatrixBase< Derived > &A, std::size t n)
     Kronecker power.

    template<typename Derived >

  DynMat< typename Derived::Scalar > reshape (const Eigen::MatrixBase< Derived > &A, std::size_t rows,
  std::size_t cols)
     Reshape.

    template<typename Derived >

  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 >

  DynMat< typename Derived::Scalar > ptrace1 (const Eigen::MatrixBase< Derived > &A, const std::vector<
  std::size t > \&dims)
     Partial trace.

    template<typename Derived >

  DynMat< typename Derived::Scalar > ptrace2 (const Eigen::MatrixBase< Derived > &A, const std::vector<
  std::size t > \&dims)
```

• template<typename Derived >

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 >

Partial transpose.

• template<typename Derived1 , typename Derived2 >

Commutator.

• template<typename Derived1 , typename Derived2 >

Anti-commutator.

template<typename Derived >

DynMat< typename Derived::Scalar > prj (const Eigen::MatrixBase< Derived > &V)

Projector.

template<typename Derived >

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 >

DynMat< typename Derived::Scalar > grams (const std::vector< Derived > &Vs)

Gram-Schmidt orthogonalization (std::vector overload)

template<typename Derived >

DynMat< typename Derived::Scalar > grams (const std::initializer_list< Derived > &Vs)

Gram-Schmidt orthogonalization (std::initializer_list overload)

template<typename Derived >

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.

ket mket (const std::vector< std::size_t > &mask)

Multi-partite qubit ket.

ket mket (const std::vector< std::size_t > &mask, const std::vector< std::size_t > &dims)

Multi-partite qudit ket (different dimensions overload)

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=chop, std::ostream &os=std::cout)
 \bullet \ \ \mathsf{template} \mathord{<} \mathsf{typename} \ \mathsf{Derived} >
  void displn (const Eigen::MatrixBase< Derived > &A, double chop=chop, std::ostream &os=std::cout)

    void disp (const cplx c, double chop=chop, std::ostream &os=std::cout)

    void displn (const cplx c, double chop=chop, std::ostream &os=std::cout)

    template<typename Derived >

  void save (const Eigen::MatrixBase< Derived > &A, const std::string &fname)

    template<typename Derived >

  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<>

  dmat loadMATLABmatrix (const std::string &mat file, const std::string &var name)
• template<>
  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 < dmat > &A, const std::string &mat_file, const std::string
  &var_name, const std::string &mode)
  void saveMATLABmatrix (const Eigen::MatrixBase < 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<>
  dmat rand (std::size_t rows, std::size_t cols, double a, double b)
template<>
  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<>
  dmat randn (std::size_t rows, std::size_t cols, double mean, double sigma)
template<>
  cmat randn (std::size t rows, std::size t cols, double mean, double sigma)

    double randn (double mean=0, double sigma=1)

    cmat randU (std::size t D)

    cmat randV (std::size t Din, std::size t Dout)

    std::vector< cmat > randkraus (std::size_t n, std::size_t D)

    cmat randH (std::size_t D)

• ket randket (std::size t D)

    cmat randrho (std::size t D)
```

std::vector< std::size_t > randperm (std::size_t n)

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

π

• constexpr double ee = 2.718281828459045235360287471352662497

Base of natural logarithm, e.

• RandomDevices & rdevs = RandomDevices::get_instance()

gpp::RandomDevices Singleton

const Gates & gt = Gates::get_instance()

qpp::Gates const Singleton

const States & st = States::get_instance()

qpp::States const Singleton

6.1.1 Typedef Documentation

6.1.1.1 using qpp::bra = typedef Eigen::Matrix<cplx, 1, Eigen::Dynamic>

Complex (double precision) dynamic Eigen row matrix.

6.1.1.2 using qpp::cmat = typedef Eigen::MatrixXcd

Complex (double precision) dynamic Eigen matrix.

6.1.1.3 using qpp::cplx = typedef std::complex < double >

Complex number in double precision.

6.1.1.4 using qpp::dmat = typedef Eigen::MatrixXd

Real (double precision) dynamic Eigen matrix.

6.1.1.5 template<typename Scalar > using qpp::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>
```

6.1.1.6 using qpp::ket = typedef Eigen::Matrix<cplx, Eigen::Dynamic, 1>

Complex (double precision) dynamic Eigen column matrix.

6.1.2 Function Documentation

6.1.2.1 template < typename Derived > cmat qpp::absm (const Eigen::MatrixBase < Derived > & A)

Matrix absolut value.

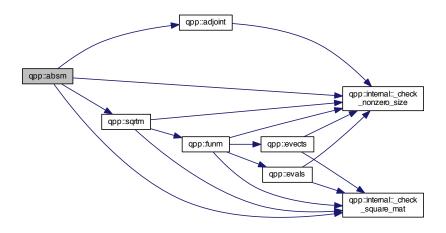
Parameters

Α	Eigen expression

Returns

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

Here is the call graph for this function:



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

Adjoint.

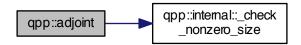
Parameters

Α	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:



6.1.2.3 template<typename Derived1 , typename Derived2 > 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

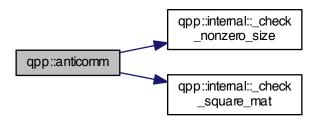
Parameters

Α	Eigen expression
В	Eigen expression

Returns

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

Here is the call graph for this function:



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

Applies the channel specified by the set of Kraus operators Ks to the density matrix rho.

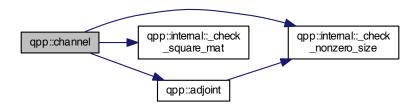
Parameters

rho	Eigen expression
Ks	std::vector of Eigen expressions representing the set of Kraus operators

Returns

Output density matrix, as a dynamic matrix over the complex field

Here is the call graph for this function:



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

Applies the channel specified by the set of Kraus operators *Ks* to the part of the density matrix *rho* specified by *subsys*.

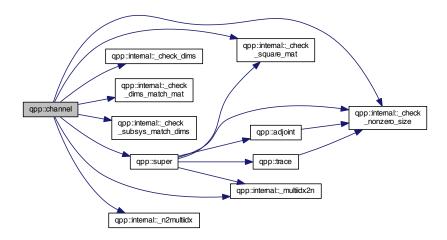
Parameters

rho	Eigen expression
Ks	std::vector of Eigen expressions representing the set of Kraus operators
subsys	Subsystems' indexes
dims	Dimensions of the multi-partite system

Returns

Output density matrix, as a dynamic matrix over the complex field

Here is the call graph for this function:



6.1.2.6 cmat qpp::choi (const std::vector< cmat > & Ks)

Choi matrix representation.

Constructs the Choi matrix of the channel specified by the set of Kraus operators Ks in the standard operator basis $\{|i\rangle\langle j|\}$ ordered in lexicographical order, i.e. $|0\rangle\langle 0|$, $|0\rangle\langle 1|$ etc.

Note

the superoperator matrix S and the Choi matrix C are related by $S_{ab,mn} = C_{ma,nb}$

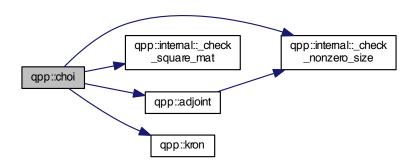
Parameters

Ks	std::vector of Eigen expressions representing the set of Kraus operators

Returns

Choi matrix representation, as a dynamic matrix over the complex field

Here is the call graph for this function:



6.1.2.7 std::vector<cmat> qpp::choi2kraus (const cmat & A)

Extracts orthogonal Kraus operators from Choi matrix.

Extracts a set of orthogonal (under Hilbert-Schmidt operator norm) Kraus operators from the Choi representation *A* of the channel

Note

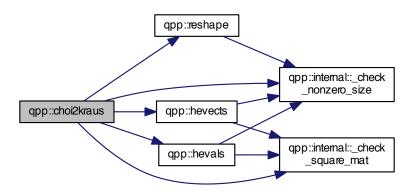
The Kraus operators satisfy $Tr(K_i^\dagger K_i) = \delta_{ij}$ for all $i \neq j$

Parameters

Α	Choi matrix

std::vector of dynamic matrices over the complex field representing the set of Kraus operators

Here is the call graph for this function:



6.1.2.8 template<typename Derived1 , typename Derived2 > DynMat<typename Derived1::Scalar> qpp::comm (const Eigen::MatrixBase< Derived2 > & B)

Commutator.

Commutator [A,B] = AB - BA

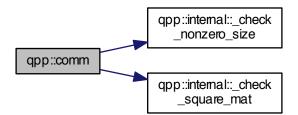
Both A and B must be Eigen expressions over the same scalar field

Parameters

Α	Eigen expression
В	Eigen expression

Returns

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



Compose permutations.

perm	Permutation
sigma	Permutation

Returns

Composition of the permutations *perm* o *sigma* = perm(sigma)

Here is the call graph for this function:



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

Complex conjugate.

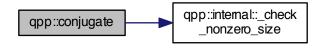
Parameters

	Α	Eigen expression
--	---	------------------

Returns

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

Here is the call graph for this function:



 $6.1.2.11 \quad template < typename \ Derived > cmat \ qpp::cosm \ (\ const \ Eigen::MatrixBase < Derived > \& \ \textit{A} \)$

Matrix cos.

Α	Eigen expression
---	------------------

Returns

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

Here is the call graph for this function:



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

Functor.

Parameters

Α	Eigen expression
f	Pointer-to-function from scalars of A to OutputScalar

Returns

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

Here is the call graph for this function:



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

Determinant.

Α	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:



- 6.1.2.14 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)
- 6.1.2.15 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)
- 6.1.2.16 template<typename Derived > void qpp::disp (const Eigen::MatrixBase< Derived > & A, double chop = chop, std::ostream & os = std::cout)
- 6.1.2.17 void qpp::disp (const cplx c, double chop = chop, std::ostream & os = std::cout)



6.1.2.18 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:



6.1.2.19 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:



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



6.1.2.21 void qpp::displn (const cplx c, double chop = chop, std::ostream & os = std::cout)

Here is the call graph for this function:



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

Entanglement of the bi-partite pure state A.

Note

Defined as the von-Neumann entropy of the reduced density matrix of one of the subsystems

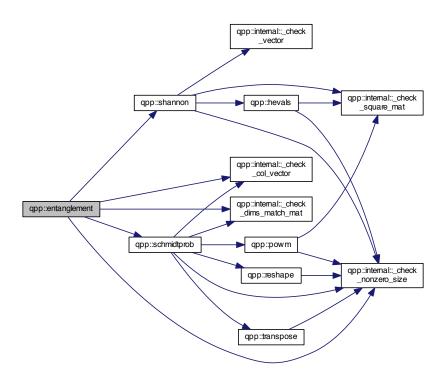
See also

qpp::shannon()

Α	Eigen expression
dims	Subsystems' dimensions

Entanglement, with the logarithm in base 2

Here is the call graph for this function:

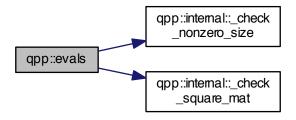


6.1.2.23 template < typename Derived > cmat qpp::evals (const Eigen::MatrixBase < Derived > & A)

Eigenvalues.

Α	Eigen expression

Eigenvalues of *A*, as a diagonal dynamic matrix over the complex field, with the eigenvalues on the diagonal Here is the call graph for this function:



6.1.2.24 template < typename Derived > cmat qpp::evects (const Eigen::MatrixBase < Derived > & A)

Eigenvectors.

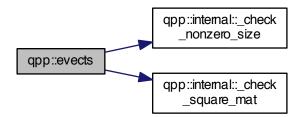
Parameters

Α	Eigen expression

Returns

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

Here is the call graph for this function:



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

Expand out.

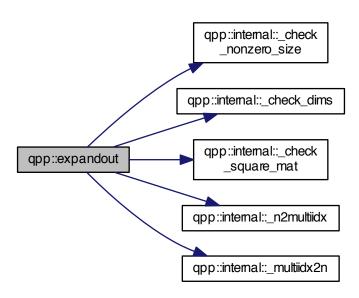
Expand out A as a matrix in a multi-partite system Faster than using *qpp::kron*(I, I, ..., I, A, I, ..., I)

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

Returns

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

Here is the call graph for this function:



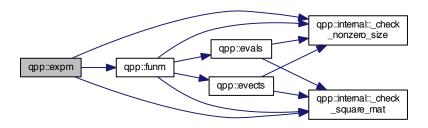
6.1.2.26 template < typename Derived > cmat qpp::expm (const Eigen::MatrixBase < Derived > & A)

Matrix exponential.

Α	Eigen expression

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

Here is the call graph for this function:



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

Functional calculus f(A)

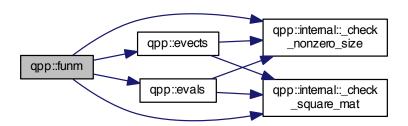
Parameters

Α	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:



6.1.2.28 template < typename Derived > double qpp::gconcurrence (const Eigen::MatrixBase < Derived > & A)

G-concurrence of the bi-partite pure state A.

Note

Uses qpp::logdet() to avoid overflows

See also

qpp::logdet()

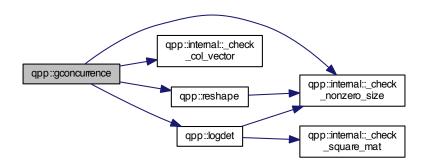
Parameters

Α	Eigen expression
dims	Subsystems' dimensions

Returns

G-concurrence

Here is the call graph for this function:



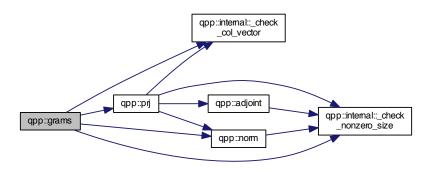
6.1.2.29 template < typename Derived > DynMat < typename Derived::Scalar > qpp::grams (const std::vector < Derived > & $\it Vs$)

Gram-Schmidt orthogonalization (std::vector overload)

Vs	std::vector of Eigen expressions as column vectors

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

Here is the call graph for this function:



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

Gram-Schmidt orthogonalization (std::initializer_list overload)

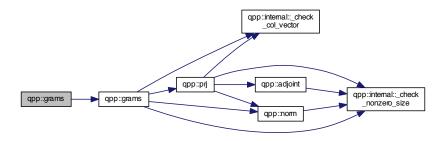
Parameters

Vs	std::initializer_list 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:



6.1.2.31 template<typename Derived > DynMat<typename Derived::Scalar> qpp::grams (const Eigen::MatrixBase< Derived > & A)

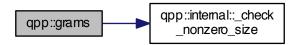
Gram-Schmidt orthogonalization (Eigen expression (matrix) overload)

A Eigen expression, the input vectors are the columns of A

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:



6.1.2.32 template<typename Derived > dmat qpp::hevals (const Eigen::MatrixBase< Derived > & A)

Hermitian eigenvalues.

Parameters

Α	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:



6.1.2.33 template < typename Derived > cmat qpp::hevects (const Eigen::MatrixBase < Derived > & A)

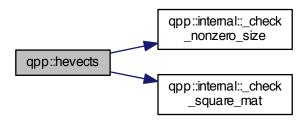
Hermitian eigenvectors.

Α	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:



6.1.2.34 template < typename Derived > DynMat < typename Derived::Scalar > qpp::inverse (const Eigen::MatrixBase < Derived > & A)

Inverse.

Parameters

A	Ligen expression

Returns

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

Here is the call graph for this function:



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

Inverse permutation.

perm	Permutation
------	-------------

Returns

Inverse of the permutation perm

Here is the call graph for this function:



6.1.2.36 template<typename T > 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

head Eigen expression	
-----------------------	--

Returns

Its argument head

6.1.2.37 template<typename T , typename... Args> DynMat<typename T::Scalar> qpp::kron (const T & head, const Args &... tail)

Kronecker product (variadic overload)

head	Eigen expression
tail	Variadic Eigen expression (zero or more parameters)

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:



6.1.2.38 template < typename Derived > DynMat < typename Derived::Scalar > qpp::kron (const std::vector < Derived > & As)

Kronecker product (std::vector overload)

Parameters

As	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:



6.1.2.39 template < typename Derived > DynMat < typename Derived::Scalar > qpp::kron (const std::initializer_list < Derived > & As)

Kronecker product (std::initializer_list overload)

ĺ	As	std::initializer_list of Eigen expressions, such as {A1, A2, ,Ak}

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:



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

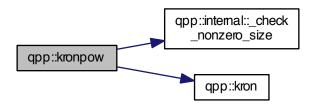
Kronecker power.

Parameters

Α	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



- 6.1.2.41 template<typename Derived > DynMat<typename Derived::Scalar> qpp::load (const std::string & fname)
- 6.1.2.42 template<typename Derived > Derived qpp::loadMATLABmatrix (const std::string & mat_file, const std::string & var_name)

- 6.1.2.43 template <> dmat qpp::loadMATLABmatrix (const std::string & mat_file, const std::string & var_name)
- 6.1.2.44 template <> cmat qpp::loadMATLABmatrix (const std::string & mat_file, const std::string & var_name)
- 6.1.2.45 template<typename Derived > Derived::Scalar qpp::logdet (const Eigen::MatrixBase< Derived > & A)

Logarithm of the determinant.

Especially useful when the determinant overflows/underflows

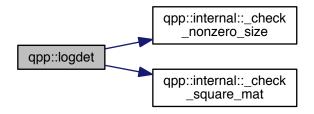
Parameters

Α	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:



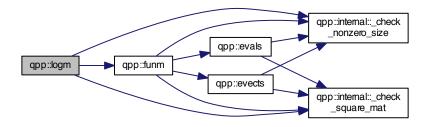
6.1.2.46 template<typename Derived > cmat qpp::logm (const Eigen::MatrixBase< Derived > & A)

Matrix logarithm.

Α	Eigen expression

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

Here is the call graph for this function:



6.1.2.47 ket qpp::mket (const std::vector< std::size_t > & mask)

Multi-partite qubit ket.

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

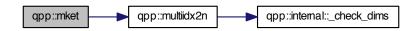
Parameters

|--|

Returns

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

Here is the call graph for this function:



6.1.2.48 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 $|mask\rangle$, 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

Generated on Fri Oct 24 2014 01:31:37 for quantum++ by Doxygen

mask	std::vector of non-negative integers
dims	Dimensions of the multi-partite system

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

Here is the call graph for this function:



6.1.2.49 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 $|mask\rangle$ 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

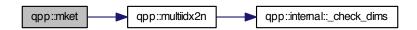
Parameters

mask	std::vector of non-negative integers
d	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:



6.1.2.50 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.

midx	Multi-index
dims	Dimensions of the multi-partite system

Returns

Non-negative integer index

Here is the call graph for this function:



 $6.1.2.51 \quad std::vector < std::size_t > qpp::n2multiidx (\ std::size_t \textit{n, } const \ std::vector < std::size_t > \& \textit{dims })$

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:



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

Trace norm.

```
A Eigen expression
```

Returns

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

Here is the call graph for this function:



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

D-th root of unity.

Parameters

```
D Non-negative integer
```

Returns

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

```
6.1.2.54 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>
```

6.1.2.55 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>
```

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

Matrix power.

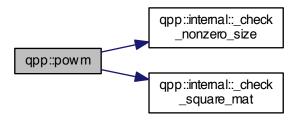
Explicitly multiplies the matrix ${\it A}$ with itself ${\it n}$ times By convention ${\it A}^0={\it I}$

Α	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:



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

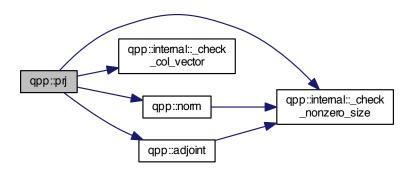
Projector.

Normalized projector onto state vector

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

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

Here is the call graph for this function:



6.1.2.58 template<typename Derived > 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

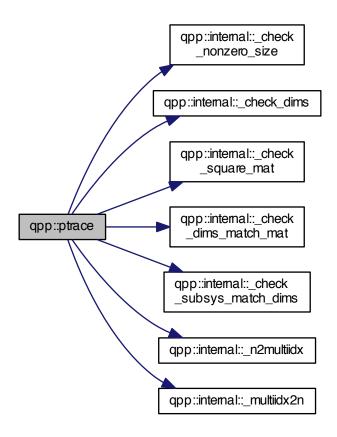
Parameters

Α	Eigen expression
subsys	Subsystems' indexes
dims	Dimensions of the multi-partite system

Returns

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

Here is the call graph for this function:



6.1.2.59 template < typename Derived > 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

Α	Eigen expression
dims	Dimensions of bi-partite system (must be a std::vector with 2 elements)

Partial trace $Tr_A(\cdot)$ over the first subsytem 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:



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

Partial trace.

Parameters

Α	Eigen expression
dims	Dimensions of bi-partite system (must be a std::vector 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:



Partial transpose.

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

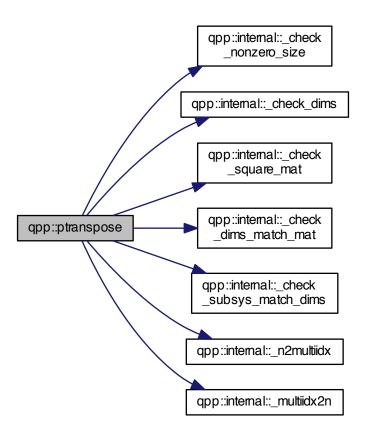
Parameters

Α	Eigen expression
subsys	Subsystems' indexes
dims	Dimensions of the multi-partite system

Returns

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

Here is the call graph for this function:

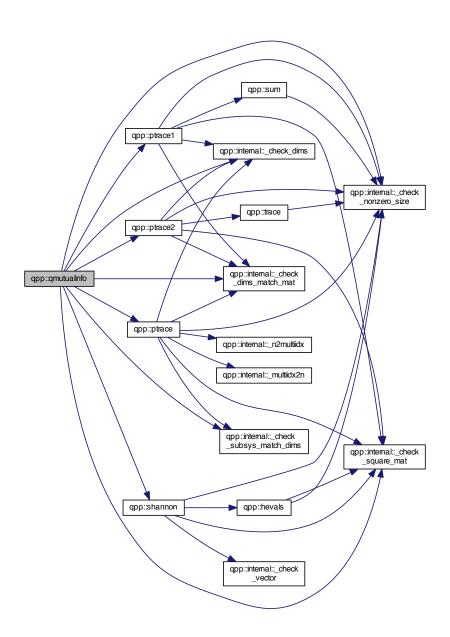


6.1.2.62 template<typename Derived > double qpp::qmutualinfo (const Eigen::MatrixBase< Derived > & A, const std::vector< std::size_t > & subsys, const std::vector< std::size_t > & dims)

Quantum mutual information between 2 subsystems of a composite system.

Α	Eigen expression
subsys	Subsystems' indexes
dims	Subsystems' dimensions

Mutual information between the 2 subsystems



- 6.1.2.63 template < typename Derived > Derived qpp::rand (std::size_t rows, std::size_t cols, double a = 0, double b = 1)
- 6.1.2.64 template <> dmat qpp::rand (std::size_t rows, std::size_t cols, double a, double b)

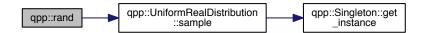
6.1.2.65 template <> cmat qpp::rand (std::size_t rows, std::size_t cols, double a, double b)

Here is the call graph for this function:



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

Here is the call graph for this function:

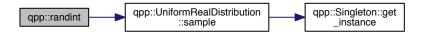


6.1.2.67 cmat qpp::randH (std::size_t D)

Here is the call graph for this function:

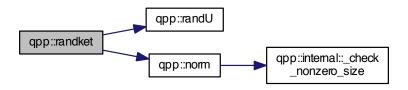


6.1.2.68 long long qpp::randint (long long a, long long b)



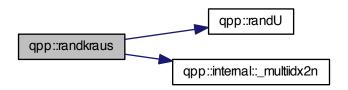
6.1.2.69 ket qpp::randket (std::size_t D)

Here is the call graph for this function:

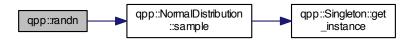


6.1.2.70 std::vector<cmat> qpp::randkraus (std::size_t n, std::size_t D)

Here is the call graph for this function:



- 6.1.2.71 template<typename Derived > Derived qpp::randn (std::size_t rows, std::size_t cols, double mean = 0, double sigma = 1)
- 6.1.2.72 template<> dmat qpp::randn (std::size_t rows, std::size_t cols, double mean, double sigma)



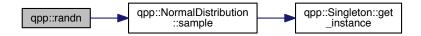
6.1.2.73 template<> cmat qpp::randn (std::size_t rows, std::size_t cols, double mean, double sigma)

Here is the call graph for this function:



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

Here is the call graph for this function:

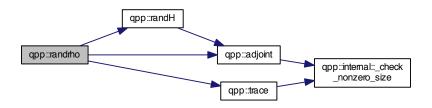


6.1.2.75 std::vector<std::size_t> qpp::randperm (std::size_t n)



6.1.2.76 cmat qpp::randrho (std::size_t D)

Here is the call graph for this function:



6.1.2.77 cmat qpp::randU (std::size_t D)

6.1.2.78 cmat qpp::randV (std::size_t Din, std::size_t Dout)

Here is the call graph for this function:



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

Renyi- α entropy of the probability distribution/density matrix \emph{A} .

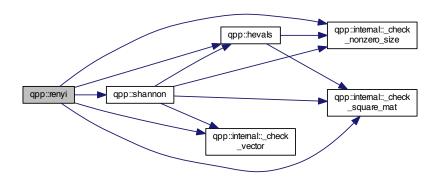
Note

 $\alpha \ge 0$

alpha	Non-negative real number
Α	Eigen expression, representing a probability distribution (dynamic column vector) or a density
	matrix (dynamic matrix over the complex field)

Renyi- α entropy, with the logarithm in base 2

Here is the call graph for this function:



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

Renyi- ∞ entropy (min entropy) of the probability distribution/density matrix *A*.

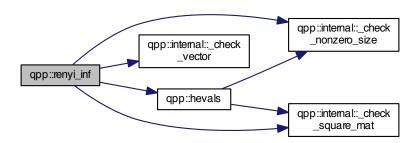
Parameters

A Eigen expression, representing a probability distribution (dynamic column vector) or a density matrix (dynamic matrix over the complex field)

Returns

Renyi- ∞ entropy (min entropy), with the logarithm in base 2

Here is the call graph for this function:



Reshape.

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

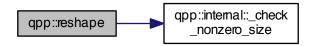
Parameters

Α	Eigen expression
rows	Number of rows of the reshaped matrix
cols	Number of columns of the reshaped matrix

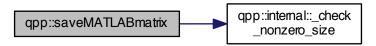
Returns

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:

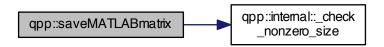


- 6.1.2.83 template < typename Derived > void qpp::saveMATLABmatrix (const Eigen::MatrixBase < Derived > & A, const std::string & mat_file, const std::string & var_name, const std::string & mode)
- 6.1.2.84 template<> void qpp::saveMATLABmatrix (const Eigen::MatrixBase< dmat > & A, const std::string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



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

Here is the call graph for this function:



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

Schmidt coefficients of the bi-partite pure state A.

Note

The sum of the squares of the Schmidt coefficients equals 1

See also

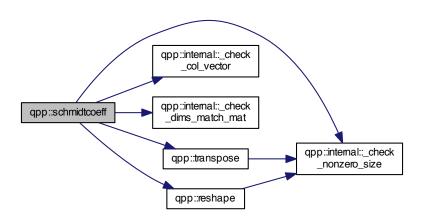
qpp::schmidtprob()

Parameters

Α	Eigen expression
dims	Subsystems' dimensions

Returns

Schmidt coefficients of A, as a dynamic matrix over the complex field, with the Schmidt coefficients on the diagonal



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

Schmidt probabilities of the bi-partite pure state A.

Note

Defined as the squares of the Schmidt coefficients The sum of the Schmidt probabilities equals 1

See also

qpp::schmidtcoeff()

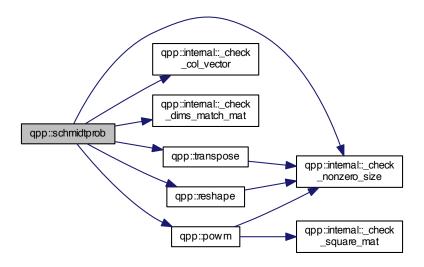
Parameters

Α	Eigen expression
dims	Subsystems' dimensions

Returns

Schmidt probabilites of A, as a dynamic matrix over the complex field, with the Schmidt probabilities on the diagonal

Here is the call graph for this function:



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

Schmidt basis on Alice's side.

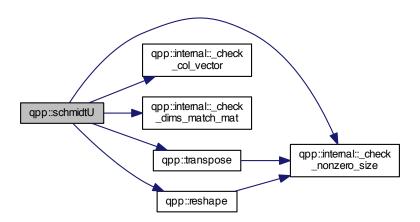
Parameters

Α	Eigen expression
dims	Subsystems' dimensions

Returns

Unitary matrix U representing the Schmidt basis on Alice's side, as a dynamic matrix over the complex field, acting on the computational basis as $U|j\rangle=|\bar{j}\rangle$ (Schmidt vector)

Here is the call graph for this function:



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

Schmidt basis on Bob's side.

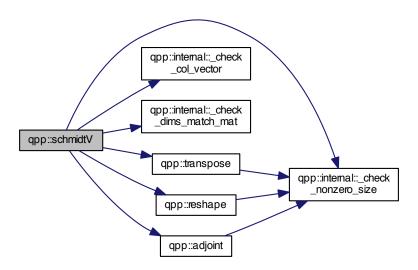
Parameters

Α	Eigen expression
dims	Subsystems' dimensions

Returns

Unitary matrix V representing the Schmidt basis on Bob's side, as a dynamic matrix over the complex field, acting on the computational basis as $V|j\rangle = |\bar{j}\rangle$ (Schmidt vector)

Here is the call graph for this function:



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

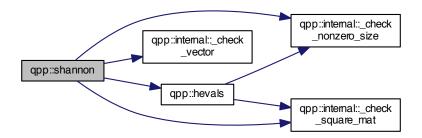
Shannon/von-Neumann entropy of the probability distribution/density matrix A.

Parameters

A Eigen expression, representing a probability distribution (dynamic column vector) or a density matrix (dynamic matrix over the complex field)

Returns

Shannon/von-Neumann entropy, with the logarithm in base 2



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

Matrix sin.

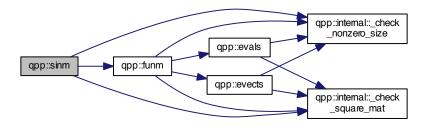
Parameters

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

Returns

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

Here is the call graph for this function:



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

Matrix power.

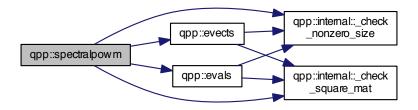
Uses the spectral decomposition of \emph{A} to compute the matrix power By convention $\emph{A}^0 = \emph{I}$

Parameters

Α	Eigen expression
Z	Complex number

Returns

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



Matrix square root.

 $6.1.2.93 \quad template < typename \ Derived > \mathbf{cmat} \ \mathsf{qpp} :: \mathsf{sqrtm} \ (\ \mathsf{const} \ \mathsf{Eigen} :: \mathsf{MatrixBase} < \mathsf{Derived} > \& \ \mathsf{A} \)$

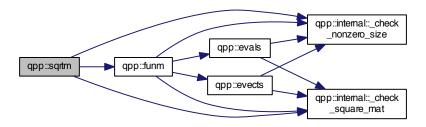
Parameters

Α	Eigen expression
---	------------------

Returns

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

Here is the call graph for this function:



6.1.2.94 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:



6.1.2.95 cmat qpp::super (const std::vector < cmat > & Ks)

Superoperator matrix representation.

Constructs the superoperator matrix of the channel specified by the set of Kraus operators Ks in the standard operator basis $\{|i\rangle\langle j|\}$ ordered in lexicographical order, i.e. $|0\rangle\langle 0|$, $|0\rangle\langle 1|$ etc.

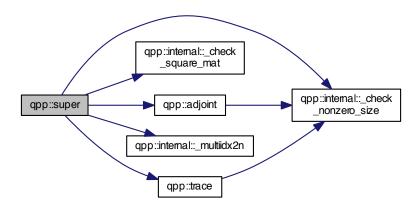
Parameters

Ks	std::vector of Eigen expressions representing the set of Kraus operators
----	--

Returns

Superoperator matrix representation, as a dynamic matrix over the complex field

Here is the call graph for this function:



System permutation.

Permutes the subsystems in a state vector or density matrix The qubit perm[i] is permuted to the location i

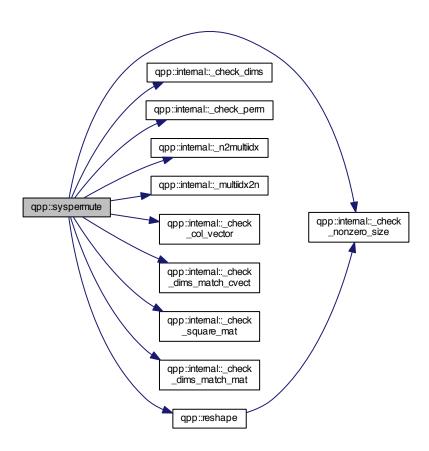
Parameters

Α	Eigen expression
perm	Permutation
dims	Subsystems' dimensions

Returns

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

Here is the call graph for this function:



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

Trace.

Parameters

Α	Eigen expression

Returns

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

Here is the call graph for this function:



6.1.2.98 template<typename Derived > DynMat<typename Derived::Scalar> qpp::transpose (const Eigen::MatrixBase< Derived > & A)

Transpose.

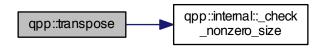
Parameters

Α	Eigen expression

Returns

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

Here is the call graph for this function:



 $6.1.2.99 \quad template < type name\ Derived > double\ qpp::tsallis\ (\ const\ double\ alpha,\ const\ Eigen::MatrixBase < Derived > \&\ A\)$

Tsallis- α entropy of the probability distribution/density matrix A.

Note

 $\alpha > 0$

When $\alpha \to 1$ the Tsallis entropy converges to the Shannon/von-Neumann entropy, with the logarithm in base ℓ

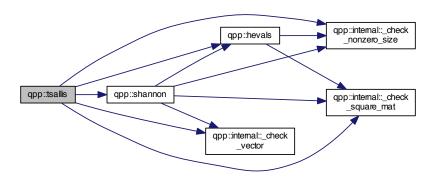
Parameters

alpha	Non-negative real number
Α	Eigen expression, representing a probability distribution (dynamic column vector) or a density
	matrix (dynamic matrix over the complex field)

Returns

Renyi- α entropy, with the logarithm in base 2

Here is the call graph for this function:



6.1.3 Variable Documentation

6.1.3.1 constexpr double qpp::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*.

6.1.3.2 constexpr double qpp::ee = 2.718281828459045235360287471352662497

Base of natural logarithm, e.

6.1.3.3 constexpr double qpp::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</pre>
```

6.1.3.4 const Gates& qpp::gt = Gates::get_instance()

qpp::Gates const Singleton

Initializes the gates, see the class qpp::Gates

```
6.1.3.5 constexpr std::size_t qpp::maxn = 64
Maximum number of qubits.
Used internally to statically allocate arrays (for speed reasons)
6.1.3.6 constexpr double qpp::pi = 3.141592653589793238462643383279502884
π
6.1.3.7 RandomDevices& qpp::rdevs = RandomDevices::get_instance()
qpp::RandomDevices Singleton
Initializes the random devices, see the class qpp::RandomDevices
6.1.3.8 const States& qpp::st = States::get_instance()
qpp::States const Singleton
```

6.2 qpp::internal Namespace Reference

Initializes the states, see the class *qpp::States*

Functions

> &V)

> &dims)

```
• 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 <u>_check_square_mat</u> (const Eigen::MatrixBase< Derived > &A)

    template<typename Derived >

  bool <u>_check_vector</u> (const Eigen::MatrixBase< Derived > &A)
• template<typename Derived >
  bool <u>_check_row_vector</u> (const Eigen::MatrixBase< Derived > &A)

    template<typename Derived >

  bool check col vector (const Eigen::MatrixBase < Derived > &A)
template<typename T >
  bool <u>_check_nonzero_size</u> (const T &x)

    bool <u>_check_dims</u> (const std::vector < std::size_t > &dims)

    template<typename Derived >

  bool <u>_check_dims_match_mat</u> (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
```

bool check subsys match dims (const std::vector< std::size t > &subsys, const std::vector< std::size t

DynMat< typename Derived1::Scalar > _kron2 (const Eigen::MatrixBase< Derived1 > &A, const Eigen::←

bool <u>_check_eq_dims</u> (const std::vector< std::size_t > &dims, std::size_t dim)

bool check perm (const std::vector < std::size t > &perm)

MatrixBase < Derived2 > &B)

template<typename Derived1 , typename Derived2 >

- 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)

6.2.1 Detailed Description

Internal functions, do not modify or use them directly

6.2.2 Function Documentation

- 6.2.2.1 template < typename Derived > bool qpp::internal::_check_col_vector (const Eigen::MatrixBase < Derived > & A)
- 6.2.2.2 bool qpp::internal::_check_dims (const std::vector < std::size_t > & dims)
- 6.2.2.3 template<typename Derived > bool qpp::internal::_check_dims_match_cvect (const std::vector< std::size_t > & dims, const Eigen::MatrixBase< Derived > & V)
- 6.2.2.4 template<typename Derived > bool qpp::internal::_check_dims_match_mat (const std::vector< std::size_t > & dims, const Eigen::MatrixBase< Derived > & A)
- 6.2.2.5 template<typename Derived > bool qpp::internal::_check_dims_match_rvect (const std::vector< std::size_t > & dims, const Eigen::MatrixBase< Derived > & V)
- 6.2.2.6 bool qpp::internal::_check_eq_dims (const std::vector < std::size_t > & dims, std::size_t dim)
- 6.2.2.7 template < typename T > bool qpp::internal::_check_nonzero_size (const T & x)
- 6.2.2.8 bool qpp::internal::_check_perm (const std::vector < std::size_t > & perm)
- 6.2.2.9 template < typename Derived > bool qpp::internal:: check_row_vector(_const Eigen::MatrixBase < Derived > & A)
- 6.2.2.10 template < typename Derived > bool qpp::internal::_check_square_mat (const Eigen::MatrixBase < Derived > & A)
- 6.2.2.11 bool qpp::internal::_check_subsys_match_dims (const std::vector< std::size_t > & subsys, const std::vector< std::size_t > & dims)
- 6.2.2.12 template < typename Derived > bool qpp::internal::_check_vector (const Eigen::MatrixBase < Derived > & A)



6.2.2.14 std::size_t qpp::internal::_multiidx2n (const std::size_t * midx, std::size_t numdims, const std::size_t * dims)
6.2.2.15 void qpp::internal::_n2multiidx (std::size_t n, std::size_t numdims, const std::size_t * dims, std::size_t * result)
6.2.2.16 template<typename T > void qpp::internal::variadic_vector_emplace (std::vector< T > &)
6.2.2.17 template<typename T , typename First , typename... Args> void qpp::internal::variadic_vector_emplace (

Here is the call graph for this function:

std::vector < T > & v, First && first, Args &&... args)



Namespace	Documer	ntation

Chapter 7

Class Documentation

7.1 qpp::DiscreteDistribution Class Reference

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

Public Member Functions

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

Protected Attributes

```
std::discrete_distributionstd::size_t > _d
```

7.1.1 Constructor & Destructor Documentation

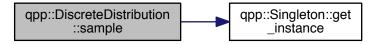
- 7.1.1.1 template < typename InputIterator > qpp::DiscreteDistribution::DiscreteDistribution (InputIterator first, InputIterator last) [inline]
- 7.1.1.2 qpp::DiscreteDistribution::DiscreteDistribution (std::initializer_list< double > weights) [inline]
- 7.1.1.3 qpp::DiscreteDistribution::DiscreteDistribution (std::vector< double > weights) [inline]

7.1.2 Member Function Documentation

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

7.1.2.2 std::size_t qpp::DiscreteDistribution::sample() [inline]

Here is the call graph for this function:



7.1.3 Member Data Documentation

7.1.3.1 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

7.2 qpp::DiscreteDistributionAbsSquare Class Reference

#include <stat.h>

Public Member Functions

- template<typename InputIterator >
 DiscreteDistributionAbsSquare (InputIterator first, InputIterator last)
- DiscreteDistributionAbsSquare (std::initializer_list< cplx > amplitudes)
- DiscreteDistributionAbsSquare (std::vector< 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_distributionstd::size_t > _d

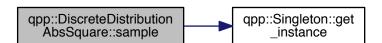
7.2.1 Constructor & Destructor Documentation

- 7.2.1.1 template<typename InputIterator > qpp::DiscreteDistributionAbsSquare::DiscreteDistributionAbsSquare (InputIterator *first*, InputIterator *last*) [inline]
- 7.2.1.2 qpp::DiscreteDistributionAbsSquare::DiscreteDistributionAbsSquare (std::initializer_list< cplx > amplitudes) [inline]
- 7.2.1.3 qpp::DiscreteDistributionAbsSquare::DiscreteDistributionAbsSquare (std::vector < cplx > amplitudes) [inline]
- 7.2.1.4 template<typename Derived > qpp::DiscreteDistributionAbsSquare::DiscreteDistributionAbsSquare (const Eigen::MatrixBase< Derived > & V) [inline]

7.2.2 Member Function Documentation

- 7.2.2.1 template<typename InputIterator > std::vector<double> qpp::DiscreteDistributionAbsSquare::cplx2weights (InputIterator first, InputIterator last) const [inline], [protected]
- **7.2.2.2** std::vector<double> qpp::DiscreteDistributionAbsSquare::probabilities () const [inline]
- 7.2.2.3 std::size_t qpp::DiscreteDistributionAbsSquare::sample() [inline]

Here is the call graph for this function:



7.2.3 Member Data Documentation

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

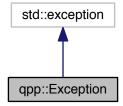
The documentation for this class was generated from the following file:

• include/classes/stat.h

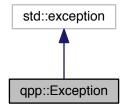
7.3 qpp::Exception Class Reference

#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_C↔ VECTOR, Type::MATRIX_NOT_SQUARE_OR_RVECTOR,

Type::MATRIX_NOT_SQUARE_OR_VECTOR, Type::DIMS_INVALID, Type::DIMS_NOT_EQUAL, Type::D↔ IMS_MISMATCH_MATRIX,

 $\label{type::DIMS_MISMATCH_CVECTOR} Type::DIMS_MISMATCH_RVECTOR, Type::DIMS_MISMATCH_VE \leftarrow CTOR, Type::SUBSYS_MISMATCH_DIMS,$

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_EXCEPT → ION }

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 noexceptoverride

Private Member Functions

• std::string _construct_exception_msg ()

Private Attributes

- · std::string _where
- std::string _msg
- Type _type
- std::string custom

7.3.1 Member Enumeration Documentation

7.3.1.1 enum qpp::Exception::Type [strong]

Enumerator

UNKNOWN_EXCEPTION Unknown exception

ZERO_SIZE Zero sized object, e.g. empty Eigen::Matrix or std::vector with no elements

MATRIX_NOT_SQUARE Eigen::Matrix is not square

MATRIX_NOT_CVECTOR Eigen::Matrix is not a column vector

MATRIX_NOT_RVECTOR Eigen::Matrix is not a row vector

MATRIX_NOT_VECTOR Eigen::Matrix is not a row/column vector

MATRIX NOT SQUARE OR CVECTOR Eigen::Matrix is not square nor a column vector

MATRIX_NOT_SQUARE_OR_RVECTOR Eigen::Matrix is not square nor a row vector

MATRIX_NOT_SQUARE_OR_VECTOR Eigen::Matrix is not square nor a row/column vector

DIMS_INVALID std::vector<std::size_t> representing the dimensions has zero size or contains zeros

DIMS_NOT_EQUAL std::vector<std::size_t> representing the dimensions contains non-equal elements

DIMS_MISMATCH_MATRIX Product of the dimenisons' std::vector<std::size_t> is not equal to the number of rows of Eigen::Matrix (assumed to be square)

DIMS_MISMATCH_CVECTOR Product of the dimenisons' std::vector<std::size_t> is not equal to the number of cols of Eigen::Matrix (assumed to be a column vector)

DIMS_MISMATCH_RVECTOR Product of the dimenisons' std::vector<std::size_t> is not equal to the number of cols of Eigen::Matrix (assumed to be a row vector)

DIMS_MISMATCH_VECTOR Product of the dimenisons' std::vector<std::size_t> is not equal to the number of cols of Eigen::Matrix (assumed to be a row/column vector)

SUBSYS_MISMATCH_DIMS std::vector<std::size_t> representing the subsystems' labels has duplicatates, or has entries that are larger than the size of the std::vector<std::size_t> representing the dimensions

PERM_INVALID Invalid std::vector<std::size_t> permutation

NOT_QUBIT_GATE Eigen::Matrix is not 2 x 2

NOT_QUBIT_SUBSYS Subsystems are not 2-dimensional

NOT_BIPARTITE std::vector<std::size t> representing the dimensions has size different from 2

OUT_OF_RANGE Parameter out of range

TYPE_MISMATCH Types do not match (i.e. Matrix<double> vs Matrix<cplx>)

UNDEFINED_TYPE Templated function not defined for this type

CUSTOM_EXCEPTION Custom exception, user must provide a custom message

7.3.2 Constructor & Destructor Documentation

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

Here is the call graph for this function:



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

Here is the call graph for this function:



7.3.3 Member Function Documentation

- 7.3.3.1 std::string qpp::Exception::_construct_exception_msg() [inline], [private]
- 7.3.3.2 virtual const char* qpp::Exception::what () const [inline], [override], [virtual], [noexcept]
- 7.3.4 Member Data Documentation
- **7.3.4.1 std::string qpp::Exception::_custom** [private]
- **7.3.4.2 std::string qpp::Exception::_msg** [private]
- **7.3.4.3 Type qpp::Exception::_type** [private]
- **7.3.4.4 std::string qpp::Exception::_where** [private]

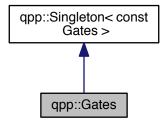
The documentation for this class was generated from the following file:

• include/classes/exception.h

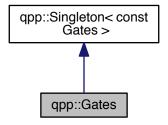
7.4 qpp::Gates Class Reference

#include <gates.h>

Inheritance diagram for qpp::Gates:



Collaboration diagram for qpp::Gates:



Public Member Functions

- cmat Rn (double theta, std::vector< double > n) const
- cmat Zd (std::size t D) const
- cmat Fd (std::size t D) const
- cmat Xd (std::size_t D) const
- template<typename Derived = Eigen::MatrixXcd>
 Derived Id (std::Size_t D) const
- template<typename Derived1 , typename Derived2 >
 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 Derived >
 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

```
cmat Id2 { cmat::Identity(2, 2) }
cmat H { cmat::Zero(2, 2) }
cmat X { cmat::Zero(2, 2) }
cmat Y { cmat::Zero(2, 2) }
cmat Z { cmat::Zero(2, 2) }
cmat S { cmat::Zero(2, 2) }
cmat T { cmat::Zero(2, 2) }
cmat CNOTab { cmat::Identity(4, 4) }
cmat CX { cmat::Identity(4, 4) }
cmat CNOTba { cmat::Zero(4, 4) }
cmat SWAP { cmat::Identity(4, 4) }
cmat TOF { cmat::Identity(8, 8) }
cmat FRED { cmat::Identity(8, 8) }
```

Private Member Functions

• Gates ()

Friends

class Singleton < const Gates >

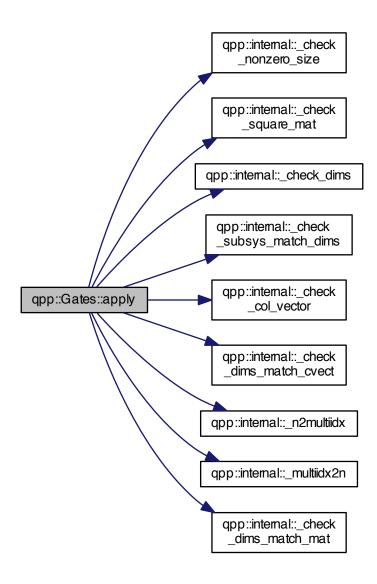
Additional Inherited Members

7.4.1 Constructor & Destructor Documentation

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

7.4.2 Member Function Documentation

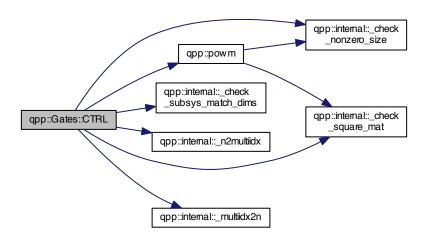
Here is the call graph for this function:



7.4.2.2 template<typename Derived1 , typename Derived2 > 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 n, std::size_t n, std::size_t d = 2) const [inline]

7.4.2.3 template<typename Derived > 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:



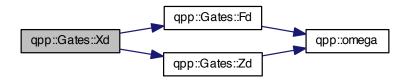
7.4.2.4 cmat qpp::Gates::Fd (std::size_t D) const [inline]



- 7.4.2.5 template<typename Derived = Eigen::MatrixXcd> Derived qpp::Gates::Id (std::size_t D) const [inline]
- 7.4.2.6 cmat qpp::Gates::Rn (double theta, std::vector < double > n) const [inline]

7.4.2.7 cmat qpp::Gates::Xd (std::size_t D) const [inline]

Here is the call graph for this function:



7.4.2.8 cmat qpp::Gates::Zd (std::size_t D) const [inline]



- 7.4.3 Friends And Related Function Documentation
- **7.4.3.1** friend class Singleton < const Gates > [friend]
- 7.4.4 Member Data Documentation
- 7.4.4.1 cmat qpp::Gates::CNOTab { cmat::Identity(4, 4) }
- 7.4.4.2 cmat qpp::Gates::CNOTba { cmat::Zero(4, 4) }
- 7.4.4.3 cmat qpp::Gates::CZ { cmat::Identity(4, 4) }
- 7.4.4.4 cmat qpp::Gates::FRED { cmat::Identity(8, 8) }
- 7.4.4.5 cmat qpp::Gates::H { cmat::Zero(2, 2) }
- 7.4.4.6 cmat qpp::Gates::ld2 { cmat::ldentity(2, 2) }
- 7.4.4.7 cmat qpp::Gates::S { cmat::Zero(2, 2) }
- 7.4.4.8 cmat qpp::Gates::SWAP { cmat::Identity(4, 4) }
- 7.4.4.9 cmat qpp::Gates::T { cmat::Zero(2, 2) }

```
    7.4.4.10 cmat qpp::Gates::TOF { cmat::Identity(8, 8) }
    7.4.4.11 cmat qpp::Gates::X { cmat::Zero(2, 2) }
    7.4.4.12 cmat qpp::Gates::Y { cmat::Zero(2, 2) }
    7.4.4.13 cmat qpp::Gates::Z { cmat::Zero(2, 2) }
```

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

• include/classes/gates.h

7.5 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

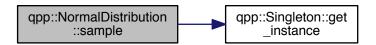
7.5.1 Constructor & Destructor Documentation

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

7.5.2 Member Function Documentation

7.5.2.1 double qpp::NormalDistribution::sample() [inline]

Here is the call graph for this function:



7.5.3 Member Data Documentation

7.5.3.1 std::normal_distribution qpp::NormalDistribution::_d [protected]

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

• include/classes/stat.h

7.6 qpp::Qudit Class Reference

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

Public Member Functions

- Qudit (const cmat &rho=States::get_instance().pz0)
- std::size_t measure (const cmat &U, bool destructive=false)
- std::size_t measure (bool destructive=false)
- cmat getRho () const
- std::size_t getD () const

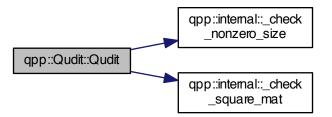
Private Attributes

- cmat _rho
- std::size_t _D

7.6.1 Constructor & Destructor Documentation

```
7.6.1.1 qpp::Qudit::Qudit ( const cmat & rho = States::get_instance () .pz0 ) [inline]
```

Here is the call graph for this function:

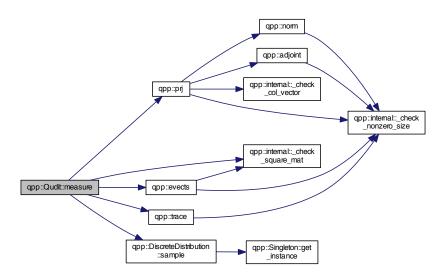


7.6.2 Member Function Documentation

- 7.6.2.1 std::size_t qpp::Qudit::getD() const [inline]
- 7.6.2.2 cmat qpp::Qudit::getRho() const [inline]

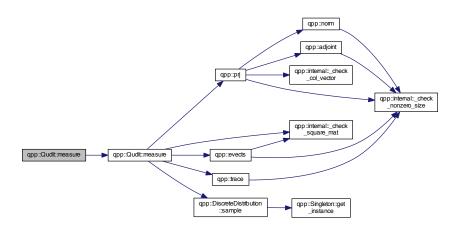
7.6.2.3 std::size_t qpp::Qudit::measure (const cmat & U, bool destructive = false) [inline]

Here is the call graph for this function:



7.6.2.4 std::size_t qpp::Qudit::measure (bool destructive = false) [inline]

Here is the call graph for this function:



7.6.3 Member Data Documentation

7.6.3.1 std::size_t qpp::Qudit::_D [private]

7.6.3.2 cmat qpp::Qudit::_rho [private]

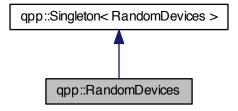
The documentation for this class was generated from the following file:

• include/classes/qudit.h

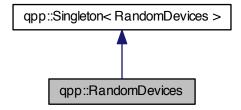
7.7 qpp::RandomDevices Class Reference

#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 < Random Devices >

Additional Inherited Members

7.7.1 Constructor & Destructor Documentation

7.7.1.1 qpp::RandomDevices::RandomDevices() [inline], [private]

7.7.2 Friends And Related Function Documentation

7.7.2.1 friend class Singleton < Random Devices > [friend]

7.7.3 Member Data Documentation

7.7.3.1 std::random_device qpp::RandomDevices::_rd [private]

7.7.3.2 std::mt19937 qpp::RandomDevices::_rng

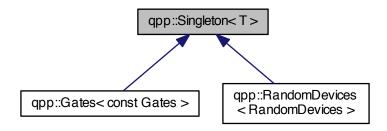
The documentation for this class was generated from the following file:

• include/classes/randevs.h

7.8 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

7.8.1 Constructor & Destructor Documentation

- 7.8.1.1 template<typename T> qpp::Singleton< T>::Singleton() [protected], [default]
- 7.8.1.2 template<typename T> virtual qpp::Singleton < T>:: \sim Singleton () [inline], [protected], [virtual]

7.8.2 Member Function Documentation

- 7.8.2.1 template<typename T> static T& qpp::Singleton < T>::get_instance() [inline], [static]
- 7.8.2.2 template<typename T> Singleton& qpp::Singleton< T>::operator= (const Singleton< T>&) [protected], [delete]

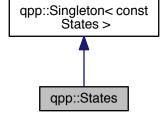
The documentation for this class was generated from the following file:

• include/classes/singleton.h

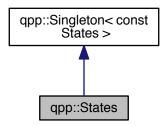
7.9 qpp::States Class Reference

#include <states.h>

Inheritance diagram for qpp::States:



Collaboration diagram for qpp::States:



Public Attributes

```
• ket x0 { ket::Zero(2) }
```

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

Private Member Functions

• States ()

Friends

class Singleton < const States >

Additional Inherited Members

```
Constructor & Destructor Documentation
7.9.1.1
        qpp::States::States( ) [inline],[private]
7.9.2
        Friends And Related Function Documentation
7.9.2.1 friend class Singleton < const States > [friend]
        Member Data Documentation
7.9.3.1
        ket qpp::States::b00 { ket::Zero(4) }
7.9.3.2
        ket qpp::States::b01 { ket::Zero(4) }
        ket qpp::States::b10 { ket::Zero(4) }
7.9.3.3
        ket qpp::States::b11 { ket::Zero(4) }
        ket qpp::States::GHZ { ket::Zero(8) }
        cmat qpp::States::pb00 { cmat::Zero(4, 4) }
7.9.3.7
        cmat qpp::States::pb01 { cmat::Zero(4, 4) }
7.9.3.8 cmat qpp::States::pb10 { cmat::Zero(4, 4) }
7.9.3.9
        cmat qpp::States::pb11 { cmat::Zero(4, 4) }
7.9.3.10 cmat qpp::States::pGHZ { cmat::Zero(8, 8) }
7.9.3.11 cmat qpp::States::pW { cmat::Zero(8, 8) }
7.9.3.12 cmat qpp::States::px0 { cmat::Zero(2, 2) }
7.9.3.13 cmat qpp::States::px1 { cmat::Zero(2, 2) }
7.9.3.14 cmat qpp::States::py0 { cmat::Zero(2, 2) }
7.9.3.15 cmat qpp::States::py1 { cmat::Zero(2, 2) }
7.9.3.16 cmat qpp::States::pz0 { cmat::Zero(2, 2) }
7.9.3.17 cmat qpp::States::pz1 { cmat::Zero(2, 2) }
7.9.3.18 ket qpp::States::W { ket::Zero(8) }
7.9.3.19 ket qpp::States::x0 { ket::Zero(2) }
7.9.3.20 ket qpp::States::x1 { ket::Zero(2) }
7.9.3.21 ket qpp::States::y0 { ket::Zero(2) }
7.9.3.22 ket qpp::States::y1 { ket::Zero(2) }
```

```
7.9.3.23 ket qpp::States::z0 { ket::Zero(2) }7.9.3.24 ket qpp::States::z1 { ket::Zero(2) }
```

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

• include/classes/states.h

7.10 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)

7.10.1 Constructor & Destructor Documentation

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

7.10.2 Member Function Documentation

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

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

- 7.10.2.3 void qpp::Timer::toc() [inline]
- 7.10.3 Friends And Related Function Documentation
- 7.10.3.1 std::ostream& operator << (std::ostream & os, const Timer & rhs) [friend]

7.10.4 Member Data Documentation

- **7.10.4.1** std::chrono::steady_clock::time_point qpp::Timer::_end [protected]
- **7.10.4.2** 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

7.11 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

7.11.1 Constructor & Destructor Documentation

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

7.11.2 Member Function Documentation

7.11.2.1 int qpp::UniformIntDistribution::sample() [inline]

Here is the call graph for this function:



7.11.3 Member Data Documentation

7.11.3.1 std::uniform_int_distribution qpp::UniformIntDistribution::_d [protected]

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

• include/classes/stat.h

7.12 qpp::UniformRealDistribution Class Reference

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

Public Member Functions

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

98 Class Documentation

Protected Attributes

· std::uniform_real_distribution _d

7.12.1 Constructor & Destructor Documentation

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

7.12.2 Member Function Documentation

7.12.2.1 double qpp::UniformRealDistribution::sample() [inline]

Here is the call graph for this function:



7.12.3 Member Data Documentation

7.12.3.1 std::uniform_real_distribution qpp::UniformRealDistribution::_d [protected]

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

• include/classes/stat.h

Chapter 8

File Documentation

8.1 include/channels.h File Reference

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



Namespaces

• qpp

Functions

- cmat qpp::super (const std::vector< cmat > &Ks)
 - Superoperator matrix representation.
- cmat qpp::choi (const std::vector< cmat > &Ks)

Choi matrix representation.

- std::vector< cmat > qpp::choi2kraus (const cmat &A)
 - Extracts orthogonal Kraus operators from Choi matrix.
- $\bullet \ \ {\it template}{<} {\it typename Derived}>$
 - cmat qpp::channel (const Eigen::MatrixBase< Derived > &rho, const std::vector< cmat > &Ks)

Applies the channel specified by the set of Kraus operators Ks to the density matrix rho.

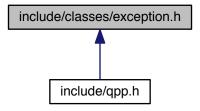
 $\bullet \ \ \mathsf{template} \mathord{<} \mathsf{typename} \ \mathsf{Derived} >$

cmat qpp::channel (const Eigen::MatrixBase< Derived > &rho, const std::vector< cmat > &Ks, const std \leftrightarrow ::vector< std::size_t > &subsys, const std::vector< std::size_t > &dims)

Applies the channel specified by the set of Kraus operators Ks to the part of the density matrix rho specified by subsys.

8.2 include/classes/exception.h File Reference

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



Classes

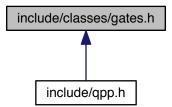
· class qpp::Exception

Namespaces

• qpp

8.3 include/classes/gates.h File Reference

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



Classes

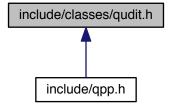
class qpp::Gates

Namespaces

dbb

8.4 include/classes/qudit.h File Reference

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



Classes

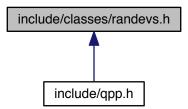
· class qpp::Qudit

Namespaces

• qpp

8.5 include/classes/randevs.h File Reference

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



Classes

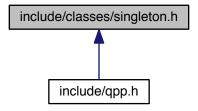
• class qpp::RandomDevices

Namespaces

dbb

8.6 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)

8.6.1 Macro Definition Documentation

8.6.1.1 #define CLASS_CONST_SINGLETON(Foo)

Value:

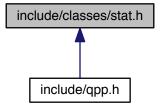
8.6.1.2 #define CLASS_SINGLETON(Foo)

Value:

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

8.7 include/classes/stat.h File Reference

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



Classes

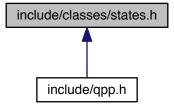
- class qpp::NormalDistribution
- class qpp::UniformRealDistribution
- class qpp::UniformIntDistribution
- class qpp::DiscreteDistribution
- class qpp::DiscreteDistributionAbsSquare

Namespaces

• qpp

8.8 include/classes/states.h File Reference

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



Classes

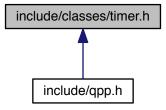
class qpp::States

Namespaces

• qpp

8.9 include/classes/timer.h File Reference

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



Classes

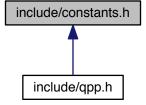
class qpp::Timer

Namespaces

• qpp

8.10 include/constants.h File Reference

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



Namespaces

qpp

Functions

```
 \bullet \ \ 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::omega (std::size_t D)

D-th root of unity.

Variables

constexpr double qpp::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::eps = 1e-12

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

• constexpr std::size_t qpp::maxn = 64

Maximum number of qubits.

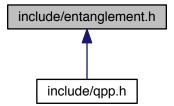
• constexpr double qpp::pi = 3.141592653589793238462643383279502884

π

constexpr double qpp::ee = 2.718281828459045235360287471352662497
 Base of natural logarithm, e.

8.11 include/entanglement.h File Reference

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



Namespaces

qpp

Functions

template<typename Derived >
 cmat qpp::schmidtcoeff (const Eigen::MatrixBase< Derived > &A, const std::vector< std::size_t > &dims)
 Schmidt coefficients of the bi-partite pure state A.

```
    template<typename Derived >
        cmat qpp::schmidtU (const Eigen::MatrixBase< Derived > &A, const std::vector< std::size_t > &dims)
    Schmidt basis on Alice's side.
```

template<typename Derived >
 cmat qpp::schmidtV (const Eigen::MatrixBase< Derived > &A, const std::vector< std::size_t > &dims)
 Schmidt basis on Bob's side.

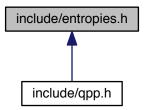
template<typename Derived >
 cmat qpp::schmidtprob (const Eigen::MatrixBase< Derived > &A, const std::vector< std::size_t > &dims)
 Schmidt probabilities of the bi-partite pure state A.

template<typename Derived >
 double qpp::entanglement (const Eigen::MatrixBase< Derived > &A, const std::vector< std::size_t > &dims)
 Entanglement of the bi-partite pure state A.

template < typename Derived >
 double qpp::gconcurrence (const Eigen::MatrixBase < Derived > &A)
 G-concurrence of the bi-partite pure state A.

8.12 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)

 $Shannon/von-Neumann\ entropy\ of\ the\ probability\ distribution/density\ matrix\ A.$

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

Renyi- lpha entropy of the probability distribution/density matrix A.

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

Renyi- ∞ entropy (min entropy) of the probability distribution/density matrix A.

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

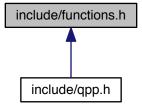
Tsallis- α entropy of the probability distribution/density matrix A.

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)

Quantum mutual information between 2 subsystems of a composite system.

8.13 include/functions.h File Reference

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



Namespaces

dbb

Functions

```
    template<typename Derived >
        DynMat< typename Derived::Scalar > qpp::transpose (const Eigen::MatrixBase< Derived > &A)
        Transpose.
```

- template<typename Derived >
 - DynMat< typename Derived::Scalar > qpp::conjugate (const Eigen::MatrixBase< Derived > &A) Complex conjugate.
- $\bullet \ \ \text{template}{<} \text{typename Derived} >$
 - DynMat< typename Derived::Scalar > qpp::adjoint (const Eigen::MatrixBase< Derived > &A) Adjoint.
- template<typename Derived >
 - DynMat< typename Derived::Scalar > qpp::inverse (const Eigen::MatrixBase< Derived > &A)
- $\bullet \ \ \text{template}{<} \text{typename Derived} >$
 - Derived::Scalar qpp::trace (const Eigen::MatrixBase< Derived > &A)

Trace.

 $\bullet \ \ \text{template}{<} \text{typename Derived}>$

Derived::Scalar qpp::det (const Eigen::MatrixBase< Derived > &A)

Determinant.

 $\bullet \ \ \mathsf{template} \mathord{<} \mathsf{typename} \ \mathsf{Derived} >$

Derived::Scalar qpp::logdet (const Eigen::MatrixBase< Derived > &A)

Logarithm of the determinant.

```
• template<typename Derived >
  Derived::Scalar <a href="mailto:qpp::sum">qpp::sum</a> (const Eigen::MatrixBase</a> Derived > &A)
     Element-wise sum.

    template<typename Derived >

  double <a href="mailto:qpp::norm">qpp::norm</a> (const Eigen::MatrixBase</a> Derived > &A)
• template<typename Derived >
  cmat gpp::evals (const Eigen::MatrixBase< Derived > &A)
     Eigenvalues.

    template<typename Derived >

  cmat qpp::evects (const Eigen::MatrixBase< Derived > &A)
      Eigenvectors.

    template<typename Derived >

  dmat qpp::hevals (const Eigen::MatrixBase< Derived > &A)
     Hermitian eigenvalues.

    template<typename Derived >

  cmat qpp::hevects (const Eigen::MatrixBase< Derived > &A)
      Hermitian eigenvectors.
• template<typename Derived >
  cmat qpp::funm (const Eigen::MatrixBase< Derived > &A, cplx(*f)(const cplx &))
      Functional calculus f(A)

    template<typename Derived >

  cmat qpp::sqrtm (const Eigen::MatrixBase< Derived > &A)
     Matrix square root.
• template<typename Derived >
  cmat qpp::absm (const Eigen::MatrixBase< Derived > &A)
     Matrix absolut value.

    template<typename Derived >

  cmat <a href="mailto:qpp::expm">qpp::expm</a> (const Eigen::MatrixBase</a> Derived > &A)
      Matrix exponential.
• template<typename Derived >
  cmat qpp::logm (const Eigen::MatrixBase< Derived > &A)
     Matrix logarithm.
• template<typename Derived >
  cmat qpp::sinm (const Eigen::MatrixBase< Derived > &A)
     Matrix sin.
template<typename Derived >
  cmat qpp::cosm (const Eigen::MatrixBase< Derived > &A)
     Matrix cos.

    template<typename Derived >

  cmat qpp::spectralpowm (const Eigen::MatrixBase< Derived > &A, const cplx z)
     Matrix power.

    template<typename Derived >

  DynMat< typename Derived::Scalar > qpp::powm (const Eigen::MatrixBase< Derived > &A, std::size_t n)
      Matrix power.
• template<typename OutputScalar , typename Derived >
  DynMat< OutputScalar > qpp::cwise (const Eigen::MatrixBase< Derived > &A, OutputScalar(*f)(const type-
  name Derived::Scalar &))
      Functor.

    template<typename T >

  DynMat< typename T::Scalar > qpp::kron (const T &head)
      Kronecker product (variadic overload)
```

• template<typename T , typename... Args> DynMat< typename T::Scalar > qpp::kron (const T &head, const Args &...tail) Kronecker product (variadic overload) template<typename Derived > DynMat< typename Derived::Scalar > qpp::kron (const std::vector< Derived > &As) Kronecker product (std::vector overload) template<typename Derived > DynMat< typename Derived::Scalar > qpp::kron (const std::initializer_list< Derived > &As) Kronecker product (std::initializer_list overload) template<typename Derived > DynMat< typename Derived::Scalar > qpp::kronpow (const Eigen::MatrixBase< Derived > &A, std::size_t n) Kronecker power. template<typename Derived > DynMat< typename Derived::Scalar > qpp::reshape (const Eigen::MatrixBase< Derived > &A, std::size t rows, std::size_t cols) Reshape. • template<typename Derived > 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 > DynMat< typename Derived::Scalar > qpp::ptrace1 (const Eigen::MatrixBase< Derived > &A, const std↔ ::vector< std::size_t > &dims) Partial trace. $\bullet \ \ \mathsf{template} \mathord{<} \mathsf{typename} \ \mathsf{Derived} >$ DynMat< typename Derived::Scalar > qpp::ptrace2 (const Eigen::MatrixBase< Derived > &A, const std← ::vector< std::size_t > &dims) Partial trace. template<typename Derived > 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 > 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 > DynMat< typename Derived1::Scalar > qpp::comm (const Eigen::MatrixBase< Derived1 > &A, const Eigen::MatrixBase< Derived2 > &B) Commutator. template<typename Derived1 , typename Derived2 > DynMat< typename Derived1::Scalar > qpp::anticomm (const Eigen::MatrixBase< Derived1 > &A, const Eigen::MatrixBase< Derived2 > &B) Anti-commutator. template<typename Derived > DynMat< typename Derived::Scalar > qpp::prj (const Eigen::MatrixBase< Derived > &V) Projector. template<typename Derived >

DynMat< typename Derived::Scalar > qpp::expandout (const Eigen::MatrixBase< Derived > &A, std::size ←

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

Generated on Fri Oct 24 2014 01:31:37 for quantum++ by Doxygen

Expand out.
• template<typename Derived >

_t pos, const std::vector< std::size_t > &dims)

Gram-Schmidt orthogonalization (std::vector overload)

• template<typename Derived >

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

Gram-Schmidt orthogonalization (std::initializer list overload)

template<typename Derived >

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.

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

Multi-partite qubit ket.

ket qpp::mket (const std::vector < std::size_t > &mask, const std::vector < std::size_t > &dims)

Multi-partite qudit ket (different dimensions overload)

ket qpp::mket (const std::vector< std::size_t > &mask, std::size_t d)

Multi-partite qudit ket (same dimensions overload)

 $\bullet \ \, \text{std::vector} < \text{std::size_t} > \text{qpp::invperm} \ \, (\text{const std::vector} < \text{std::size_t} > \text{\&perm}) \\$

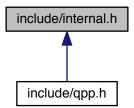
Inverse permutation.

std::vector< std::size_t > app::compperm (const std::vector< std::size_t > aperm, const std::vector< std
 ::size_t > aperm, const std::vector< std
 ::size_t

Compose permutations.

8.14 include/internal.h File Reference

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



Namespaces

- · qpp::internal
- qpp

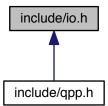
Functions

- 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 >
 DynMat< typename Derived1::Scalar > qpp::internal::_kron2 (const Eigen::MatrixBase< Derived1 > &A,
 const Eigen::MatrixBase< 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)

8.15 include/io.h File Reference

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



Namespaces

qpp

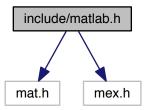
Functions

- 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=chop, std::ostream &os=std::cout)
- template<typename Derived > void qpp::displn (const Eigen::MatrixBase< Derived > &A, double chop=chop, std::ostream &os=std::cout)
- void qpp::disp (const cplx c, double chop=chop, std::ostream &os=std::cout)
- void qpp::displn (const cplx c, double chop=chop, std::ostream &os=std::cout)
- template<typename Derived > void qpp::save (const Eigen::MatrixBase< Derived > &A, const std::string &fname)
- template < typename Derived >
 DynMat < typename Derived::Scalar > qpp::load (const std::string &fname)

8.16 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<>
 dmat qpp::loadMATLABmatrix (const std::string &mat_file, const std::string &var_name)
- r template<> 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< dmat > &A, const std::string &mat_file, const std
 ::string &var_name, const std::string &mode)
- template<>
 void qpp::saveMATLABmatrix (const Eigen::MatrixBase< cmat > &A, const std::string &mat_file, const std
 ::string &var_name, const std::string &mode)

8.17 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()

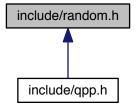
qpp::Gates const Singleton

• const States & qpp::st = States::get_instance()

qpp::States const Singleton

8.18 include/random.h File Reference

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



Namespaces

qpp

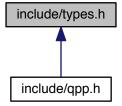
Functions

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

8.19 include/types.h File Reference

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



Namespaces

• qpp

Typedefs

- using qpp::cplx = std::complex < double >
 Complex number in double precision.
- using qpp::cmat = Eigen::MatrixXcd

Complex (double precision) dynamic Eigen matrix.

using qpp::dmat = Eigen::MatrixXd

Real (double precision) dynamic Eigen matrix.

• using qpp::ket = Eigen::Matrix< cplx, Eigen::Dynamic, 1 >

Complex (double precision) dynamic Eigen column matrix.

using qpp::bra = Eigen::Matrix < cplx, 1, Eigen::Dynamic >

Complex (double precision) dynamic Eigen row matrix.

 $\bullet \ \ \text{template}{<} \text{typename Scalar} >$

using qpp::DynMat = Eigen::Matrix < Scalar, Eigen::Dynamic, Eigen::Dynamic >

Dynamic Eigen matrix over the field specified by Scalar.

Index

absm	ann 20
qpp, 20	qpp, <mark>29</mark> displn
adjoint	qpp, 29, 30
qpp, 21	dmat
anticomm	qpp, 19
qpp, 21	qpp, 10
4pp, 21	ee
bra	qpp, 72
qpp, 19	entanglement
11 1 7	qpp, 31
CUSTOM_EXCEPTION	eps
qpp::Exception, 81	qpp, 72
channel	evals
qpp, 22, 23	qpp, 32
choi	evects
qpp, 23	qpp, 33
choi2kraus	expandout
qpp, 24	qpp, 33
chop	expm
qpp, 72	qpp, 34
cmat	,
qpp, 19	funm
comm	qpp, 35
qpp, 25	aconcurrence
compperm	gconcurrence qpp, 35
qpp, 25	grams
conjugate	qpp, 36, 37
qpp, 27	gt
cosm	qpp, 72
qpp, 27	4PP; *=
cplx	hevals
qpp, 19	qpp, 38
cwise	hevects
qpp, 28	qpp, 38
DIMS INVALID	
qpp::Exception, 81	inverse
DIMS_MISMATCH_CVECTOR	qpp, 39
qpp::Exception, 81	invperm
DIMS MISMATCH MATRIX	qpp, 39
qpp::Exception, 81	ket
DIMS MISMATCH RVECTOR	qpp, 19
qpp::Exception, 81	kron
DIMS_MISMATCH_VECTOR	qpp, 40, 41
qpp::Exception, 81	kronpow
DIMS_NOT_EQUAL	qpp, 42
qpp::Exception, 81	ـــ نجا عالـ
det	load
qpp, 28	qpp, 42
disp	logdet

INDEX 117

qpp, 43		qpp, 53
logm	qpp,	
qpp, 43		absm, 20
MATRIX_NOT_CVECTOR		adjoint, 21
qpp::Exception, 81		anticomm, 21 bra, 19
MATRIX NOT RVECTOR		channel, 22, 23
qpp::Exception, 81		choi, 23
MATRIX NOT SQUARE		choi2kraus, 24
qpp::Exception, 81		chop, 72
MATRIX_NOT_SQUARE_OR_CVECTOR		cmat, 19
qpp::Exception, 81		comm, 25
MATRIX_NOT_SQUARE_OR_RVECTOR		compperm, 25
qpp::Exception, 81		conjugate, 27
MATRIX_NOT_SQUARE_OR_VECTOR		cosm, 27
qpp::Exception, 81		cplx, 19
MATRIX_NOT_VECTOR		cwise, 28
qpp::Exception, 81		det, 28
maxn		disp, 29
qpp, 72 mket		displn, 29, 30
qpp, 44, 45		dmat, 19
multiidx2n		ee, 72
qpp, 45		entanglement, 31
4pp; 10		eps, 72
n2multiidx		evals, 32
qpp, 46		evects, 33 expandout, 33
NOT_BIPARTITE		expandout, 33
qpp::Exception, 81		funm, 35
NOT_QUBIT_GATE		gconcurrence, 35
qpp::Exception, 81		grams, 36, 37
NOT_QUBIT_SUBSYS		gt, 72
qpp::Exception, 81		hevals, 38
norm		hevects, 38
qpp, 46		inverse, 39
OUT OF RANGE		invperm, 39
app::Exception, 81		ket, 19
omega		kron, 40, 41
qpp, 47		kronpow, 42
		load, 42
PERM_INVALID		logdet, 43
qpp::Exception, 81		logm, 43
pi		maxn, 72
qpp, 73		mket, 44, 45 multiidx2n, 45
powm		n2multiidx, 46
qpp, 47		norm, 46
prj		omega, 47
qpp, 48 ptrace		pi, 73
qpp, 49		powm, 47
ptrace1		prj, 48
qpp, 50		ptrace, 49
ptrace2		ptrace1, 50
qpp, 51		ptrace2, 51
ptranspose		ptranspose, 52
qpp, 52		qmutualinfo, 53
		rand, 54, 55
qmutualinfo		randint, 55

118 INDEX

randket, 56	qpp, 57
randkraus, 56	randrho
randn, 56, 57	qpp, 57
randperm, 57	rdevs
randrho, 57	qpp, 73
rdevs, 73	renyi
renyi, 58	qpp, 58
reshape, 59	reshape
save, 61	qpp, 59
schmidtcoeff, 62	
schmidtprob, 63	SUBSYS_MISMATCH_DIMS
shannon, 65	qpp::Exception, 81
sinm, 65	save
spectralpowm, 66	qpp, 61
sqrtm, 66	schmidtcoeff
st, 73	qpp, 62
sum, 68	schmidtprob
super, 68	qpp, 63
syspermute, 69	shannon
trace, 70	qpp, 65
transpose, 71	sinm
tsallis, 71	qpp, 65
qpp::Exception	spectralpowm
CUSTOM EXCEPTION, 81	qpp, 66
DIMS INVALID, 81	sgrtm
DIMS MISMATCH CVECTOR, 81	qpp, 66
DIMS MISMATCH MATRIX, 81	st
	qpp, 73
DIMS_MISMATCH_RVECTOR, 81	sum
DIMS_MISMATCH_VECTOR, 81	qpp, 68
DIMS_NOT_EQUAL, 81	super
MATRIX_NOT_CVECTOR, 81	•
MATRIX_NOT_RVECTOR, 81	qpp, 68
MATRIX_NOT_SQUARE, 81	syspermute qpp, 69
MATRIX_NOT_SQUARE_OR_CVECTOR, 81	4ρρ, 09
MATRIX_NOT_SQUARE_OR_RVECTOR, 81	TYPE MISMATCH
MATRIX_NOT_SQUARE_OR_VECTOR, 81	qpp::Exception, 81
MATRIX_NOT_VECTOR, 81	trace
NOT_BIPARTITE, 81	qpp, 70
NOT_QUBIT_GATE, 81	transpose
NOT_QUBIT_SUBSYS, 81	qpp, 71
OUT_OF_RANGE, 81	tsallis
PERM_INVALID, 81	qpp, 71
SUBSYS_MISMATCH_DIMS, 81	чрр, 71
TYPE_MISMATCH, 81	UNDEFINED TYPE
UNDEFINED_TYPE, 81	qpp::Exception, 81
UNKNOWN_EXCEPTION, 81	UNKNOWN EXCEPTION
ZERO_SIZE, 81	qpp::Exception, 81
rand	4pp.:=//ooption, 01
rand	ZERO SIZE
qpp, 54, 55	qpp::Exception, 81
randint	
qpp, 55	
randket	
qpp, 56	
randkraus	
qpp, 56	
randn	
qpp, 56, 57	
randperm	