

Function index

In the descriptions below, matrices are represented by capital letters, vectors by lower case letters and scalars by greek lower case letters.

Function	Description	Page
<code>band2mat()</code>	Convert band matrix to dense matrix	120
<code>bd_free()</code>	Deallocate (destroy) band matrix	68
<code>bd_get()</code>	Allocate and initialise band matrix	70
<code>bd_transp()</code>	Transpose band matrix	120
<code>bd_resize()</code>	Resize band matrix	77
<code>bdLDLfactor()</code>	Band LDL^T factorisation	121
<code>bdLDLsolve()</code>	Solve $Ax = b$ using band LDL^T factors	121
<code>bdLUfactor()</code>	Band LU factorisation	124
<code>bdLUsolve()</code>	Solve $Ax = b$ using band LU factors	124
<code>bisvd()</code>	SVD of bi-diagonal matrix	143
<code>BKPfactor()</code>	Bunch–Kaufman–Parlett factorisation	116
<code>BKPsolve()</code>	Bunch–Kaufman–Parlett solver	116
<code>catch()</code>	Catch a raised error (macro)	51
<code>catchall()</code>	Catch any raised error (macro)	51
<code>catch_FPE()</code>	Catch floating point error (sets flag)	51
<code>CHfactor()</code>	Dense Cholesky factorisation	118
<code>CHsolve()</code>	Cholesky solver	118
<code>d_save()</code>	Save real in MATLAB format	91
<code>Dsolve()</code>	Solve $Dx = y$, D diagonal	135
<code>ERRABORT()</code>	Abort on error (sets flag, macro)	57
<code>ERREXIT()</code>	Exit on error (sets flag, macro)	57
<code>error()</code>	Raise an error (macro, see <code>ev_err()</code>)	53
<code>err_list_attach()</code>	Attach new list of errors	53
<code>err_list_free()</code>	Discard list of errors	53
<code>err_is_list_attached()</code>	Checks for an error list	53
<code>ev_err()</code>	Raise an error (function)	53
<code>fft()</code>	Computes Fast Fourier Transform	147
<code>finput()</code>	Input a simple data item from a stream	67

Function	Description	Page
<code>fprompter()</code>	Print prompt to <code>stderr</code>	67
<code>get_col()</code>	Extract a column from a matrix	72
<code>get_row()</code>	Extract a row from a matrix	72
<code>givens()</code>	Compute Givens parameters	130
<code>hhtrcols()</code>	Compute AP^T where P is a Householder matrix	133
<code>hhtrrows()</code>	Compute PA where P is a Householder matrix	133
<code>hhtrvec()</code>	Compute Px where P is a Householder matrix	133
<code>hhvec()</code>	Compute parameters for a Householder matrix	133
<code>ifft()</code>	Computes inverse FFT	147
<code>in_prod()</code>	Inner product of vectors	75
<code>input()</code>	Input a simple data item from <code>stdin</code> (macro)	67
<code>iter_arnoldi()</code>	Arnoldi iterative method	177
<code>iter_arnoldi_iref()</code>	Arnoldi iterative method with refinement	177
<code>iter_ATx()</code>	Set A^T in <code>ITER</code> structure	171
<code>iter_Ax()</code>	Set A in <code>ITER</code> structure	171
<code>iter_Bx()</code>	Set preconditioner in <code>ITER</code> structure	171
<code>iter_cg()</code>	Conjugate gradients iterative method	173
<code>iter_cgne()</code>	Conjugate gradients for normal equations	173
<code>iter_cgs()</code>	CGS iterative method	173
<code>iter_copy()</code>	Copy <code>ITER</code> data structures	171
<code>iter_copy2()</code>	Shallow copy of <code>ITER</code> data structures	171
<code>iter_dump()</code>	Dump <code>ITER</code> data structure to a stream	171
<code>iter_free()</code>	Free (deallocate) <code>ITER</code> structure	171
<code>iter_get()</code>	Allocate <code>ITER</code> structure	171
<code>iter_gmres()</code>	GMRES iterative method	173
<code>iter_lanczos()</code>	Lanczos iterative method	177
<code>iter_lanczos2()</code>	Lanczos method with Cullum & Willoughby extensions	177
<code>iter_lsqr()</code>	LSQR iterative method	173
<code>iter_mgcr()</code>	MGCR iterative method	173
<code>iter_resize()</code>	Change sizes in <code>ITER</code> structure	171
<code>iter_sparnoldi()</code>	Sparse matrix Arnoldi method	177
<code>iter_sparnoldi_iref()</code>	Sparse matrix Arnoldi method with refinement	177
<code>iter_spcg()</code>	Sparse matrix CG method	173
<code>iter_spcgne()</code>	Sparse matrix CG method for normal equations	173

Function	Description	Page
<code>iter_spcgs()</code>	Sparse matrix CGS method	173
<code>iter_spgmres()</code>	Sparse matrix GMRES method	173
<code>iter_splanczos()</code>	Sparse matrix basic Lanczos method	177
<code>iter_splanczos2()</code>	Sparse matrix Cullum & Willoughby Lanczos method	177
<code>iter_splsqr()</code>	Sparse matrix LSQR method	173
<code>iter_spmgcr()</code>	Sparse matrix MGCR method	173
<code>iv_add()</code>	Add integer vectors	76
<code>iv_copy()</code>	Copy integer vector	59
<code>iv_dump()</code>	Dump integer vector to a stream	65
<code>iv_finput()</code>	Input integer vector from a stream	62
<code>iv_foutput()</code>	Output integer vector to a stream	65
<code>IV_FREE()</code>	Free (deallocate) an integer vector (macro)	68
<code>iv_free()</code>	Free (deallocate) integer vector (function)	68
<code>iv_free_vars()</code>	Free a list of integer vectors	68
<code>iv_get()</code>	Allocate and initialise an integer vector	70
<code>iv_get_vars()</code>	Allocate list of integer vectors	70
<code>iv_input()</code>	Input integer vector from <code>stdin</code> (macro)	62
<code>iv_output()</code>	Output integer vector to <code>stdout</code> (macro)	65
<code>iv_resize()</code>	Resize an integer vector	77
<code>iv_resize_vars()</code>	Resize a list of integer vectors	77
<code>iv_sub()</code>	Subtract integer vectors	76
<code>LDLfactor()</code>	LDL^T factorisation	118
<code>LDLsolve()</code>	LDL^T solver	118
<code>LDLupdate()</code>	Update LDL^T factorisation	137
<code>Lsolve()</code>	Solve $Lx = y$, L lower triangular	135
<code>LTsolve()</code>	Solve $L^T x = y$, L lower triangular	135
<code>LUcondest()</code>	Estimate a condition number using LU factors	122
<code>LUfactor()</code>	Compute LU factors with implicit scaled partial pivoting	122
<code>LUsolve()</code>	Solve $Ax = b$ using LU factors	122
<code>LUTsolve()</code>	Solve $A^T x = b$ using LU factors	122
<code>m_add()</code>	Add matrices	81
<code>makeQ()</code>	Form Q matrix for QR factorisation	129
<code>makeR()</code>	Form R matrix for QR factorisation	129
<code>mat2band()</code>	Extract band matrix from dense matrix	120

Function	Description	Page
MCHfactor()	Modified Cholesky factorisation (actually factors $A + D$, D diagonal, instead of A)	118
m.copy()	Copy dense matrix	59
m.dump()	Dump matrix data structure to a stream	65
mem.attach_list()	Adds a new family of types	83
mem.bytes()	Notify change in memory usage (macro)	83
mem.bytes_list()	Notify change in memory usage	83
mem.free_list()	Frees a family of types	83
mem.info.bytes()	Number of bytes used by a type	83
mem.info.numvar()	Number of structures of a type	83
mem.info.file()	Print memory info to a stream	83
mem.info.is_on()	Is memory data being accumulated?	83
mem.info.on()	Turns memory info system on/off	83
mem.is_list_attached()	Is list of types attached?	83
mem.numvar()	Notify change in number of structures allocated (macro)	83
mem.numvar_list()	Notify change in number of structures allocated	83
mem.stat.dump()	Prints information on registered workspace	88
mem.stat.free()	Frees (deallocates) static workspace	88
mem.stat.mark()	Sets mark for workspace	88
MEM_STAT_REG()	Register static workspace (macro)	88
mem.stat.show.mark()	Current workspace group	88
m.exp()	Computes matrix exponential	145
m.finput()	Input matrix from a stream	62
m.foutput()	Output matrix to a stream	65
M_FREE()	Free (deallocate) a matrix (macro)	68
m.free()	Free (deallocate) matrix (function)	68
m.free.vars()	Free a list of matrices	68
m.get()	Allocate and initialise a matrix	70
m.get.vars()	Allocate list of matrices	70
m.ident()	Sets matrix to identity matrix	73
m.input()	Input matrix from <code>stdin</code> (macro)	62
m.inverse()	Invert matrix	122
m.load()	Load matrix in MATLAB format	91
m.mlt()	Multiplies matrices	81
mmtr.mlt()	Computes AB^T	93
m.norm1()	Computes $\ A\ _1$ of a matrix	94
m.norm_frob()	Computes the Frobenius norm of a matrix	94

Function	Description	Page
<code>m_norm_inf()</code>	Computes $\ A\ _\infty$ of a matrix	94
<code>m_ones()</code>	Set matrix to all 1's	73
<code>m_output()</code>	Output matrix to <code>stdout</code> (macro)	65
<code>m_poly()</code>	Computes a matrix polynomial	145
<code>m_pow()</code>	Computes integer power of a matrix	145
<code>mrand()</code>	Generates pseudo-random real number	73
<code>m_rand()</code>	Randomise entries of a matrix	73
<code>mrandlist()</code>	Generates array of pseudo-random numbers	73
<code>m_resize()</code>	Resize matrix	77
<code>m_resize_vars()</code>	Resize a list of matrices	77
<code>m_save()</code>	Save matrix in MATLAB format	91
<code>m_sub()</code>	Subtract matrices	81
<code>m_transp()</code>	Transpose matrix	93
<code>mtrm_mlt()</code>	Computes $A^T B$	93
<code>mv_mlt()</code>	Computes Ax	96
<code>mv_mltadd()</code>	Computes $y \leftarrow Ax + y$	96
<code>m_zero()</code>	Zero a matrix	73
<code>ON_ERROR()</code>	Error handler (macro)	57
<code>prompter()</code>	Print prompt message to <code>stdout</code>	67
<code>px_cols()</code>	Permute the columns of a matrix	99
<code>px_copy()</code>	Copy permutation	59
<code>px_dump()</code>	Dump permutation data structure to a stream	65
<code>px_finput()</code>	Input permutation from a stream	62
<code>px_foutput()</code>	Output permutation to a stream	65
<code>PX_FREE()</code>	Free (deallocate) a permutation (macro)	68
<code>px_free()</code>	Free (deallocate) permutation (function)	68
<code>px_free_vars()</code>	Free a list of permutations	68
<code>px_get()</code>	Allocate and initialise a permutation	70
<code>px_get_vars()</code>	Allocate a list of permutations	70
<code>px_ident()</code>	Sets permutation to identity	98
<code>px_input()</code>	Input permutation from <code>stdin</code> (macro)	62
<code>px_inv()</code>	Invert permutation	98
<code>pxinv_vec()</code>	Computes $P^T x$ where P is a permutation matrix	99
<code>pxinv_zvec()</code>	Computes $P^T x$ where P is a permutation matrix (complex)	99
<code>px_mlt()</code>	Multiply permutations	98
<code>px_output()</code>	Output permutation to <code>stdout</code> (macro)	65

Function	Description	Page
<code>px_resize()</code>	Resize a permutation	77
<code>px_resize_vars()</code>	Resize a list of permutations	77
<code>px_rows()</code>	Permute the rows of a matrix	99
<code>px_sign()</code>	Returns the sign of the permutation	98
<code>px_transp()</code>	Transpose a pair of entries	98
<code>px_vec()</code>	Computes Px where P is a permutation matrix	99
<code>px_zvec()</code>	Computes Px where P is a permutation matrix (complex)	99
<code>QRCPfactor()</code>	QR factorisation with column pivoting	126
<code>QRfactor()</code>	QR factorisation	126
<code>QRsolve()</code>	Solve $Ax = b$ using QR factorisation	126
<code>QRTsolve()</code>	Solve $A^T x = b$ using QR factorisation	126
<code>QRupdate()</code>	Update explicit QR factors	137
<code>rot_cols()</code>	Apply Givens rotation to the columns of a matrix	130
<code>rot_rows()</code>	Apply Givens rotation to the rows of a matrix	130
<code>rot_vec()</code>	Apply Givens rotation to a vector	130
<code>rot_zvec()</code>	Apply complex Givens rotation to a vector	130
<code>schur()</code>	Compute real Schur form	139
<code>schur_evals()</code>	Compute eigenvalues from the real Schur form	142
<code>schur_vecs()</code>	Compute eigenvectors from the real Schur form	142
<code>set_col()</code>	Set the column of a matrix to a given vector	101
<code>set_err_flag()</code>	Control behaviour of <code>ev_err()</code>	53
<code>set_row()</code>	Set the row of a matrix to a given vector	101
<code>sm_mlt()</code>	Scalar-matrix multiplication	81
<code>smrand()</code>	Set seed for <code>mrand()</code>	73
<code>spBKPfactor()</code>	Sparse symmetric indefinite factorisation	169
<code>spBKPsolve()</code>	Sparse symmetric indefinite solver	169
<code>spCHfactor()</code>	Sparse Cholesky factorisation	165
<code>spCHsolve()</code>	Sparse Cholesky solver	165
<code>spCHsymb()</code>	Symbolic sparse Cholesky factorisation (no floating point operations)	165
<code>sp_col_access()</code>	Sets up column access paths for a sparse matrix	155
<code>sp_compact()</code>	Eliminates zero entries in a sparse matrix	149

Function	Description	Page
<code>sp_copy()</code>	Copies a sparse matrix	149
<code>sp_copy2()</code>	Copies a sparse matrix into another	149
<code>sp_diag_access()</code>	Sets up diagonal access paths for a sparse matrix	155
<code>sp_dump()</code>	Dump sparse matrix data structure to a stream	158
<code>sp_finput()</code>	Input sparse matrix from a stream	160
<code>sp_foutput()</code>	Output a sparse matrix to a stream	158
<code>sp_free()</code>	Free (deallocate) a sparse matrix	149
<code>sp_get()</code>	Allocate and initialise a sparse matrix	149
<code>sp_get_val()</code>	Get the (i, j) entry of a sparse matrix	153
<code>spICHfactor()</code>	Sparse incomplete Cholesky factorisation	165
<code>sp_input()</code>	Input a sparse matrix form <code>stdin</code>	160
<code>spLUfactor()</code>	Sparse LU factorisation using partial pivoting	167
<code>spLUsolve()</code>	Solves $Ax = b$ using sparse LU factors	167
<code>spLUTsolve()</code>	Solves $A^T x = b$ using sparse LU factors	167
<code>sp_mv_mlt()</code>	Computes Ax for sparse A	154
<code>sp_output()</code>	Outputs a sparse matrix to a stream (macro)	158
<code>sp_resize()</code>	Resize a sparse matrix	149
<code>sprow_add()</code>	Adds a pair of sparse rows	162
<code>sprow_foutput()</code>	Output sparse row to a stream	162
<code>sprow_get()</code>	Allocate and initialise a sparse row	162
<code>sprow_get_idx()</code>	Get location of an entry in a sparse row	162
<code>sprow_merge()</code>	Merge two sparse rows	162
<code>sprow_mltadd()</code>	Sparse row vector multiply-and-add	162
<code>sprow_set_val()</code>	Set an entry in a sparse row	162
<code>sprow_smlt()</code>	Multiplies a sparse row by a scalar	162
<code>sprow_sub()</code>	Subtracts a sparse row from another	162
<code>sprow_xpd()</code>	Expand a sparse row	162
<code>sp_set_val()</code>	Set the (i, j) entry of a sparse matrix	153
<code>sp_vm_mlt()</code>	Compute $x^T A$ for sparse A	154
<code>sp_zero()</code>	Zero (but do not remove) all entries of a sparse matrix	157
<code>svd()</code>	Compute the SVD of a matrix	143
<code>sv_mlt()</code>	Scalar-vector multiply	102
<code>symmeig()</code>	Compute eigenvalues/vectors of a symmetric matrix	139
<code>tracecatch()</code>	Catch and re-raise errors (macro)	51
<code>trieig()</code>	Compute eigenvalues/vectors of a symmetric tridiagonal matrix	139

Function	Description	Page
Usolve()	Solve $Ux = b$ where U is upper triangular	135
UTsolve()	Solve $U^T x = b$ where U is upper triangular	135
v_add()	Add vectors	102
v_conv()	Convolution product of vectors	104
v_copy()	Copy vector	59
v_dump()	Dump vector data structure to a stream	65
v_finput()	Input vector from a stream	62
v_foutput()	Output vector to a stream	65
V_FREE()	Free (deallocate) a vector (macro)	68
v_free()	Free (deallocate) vector (function)	68
v_free_vars()	Free a list of vectors	68
v_get()	Allocate and initialise a vector	70
v_get_vars()	Allocate list of vectors	70
v_input()	Input vector from stdin (macro)	62
v_lincomb()	Compute $\sum_i a_i x_i$ for an array of vectors	107
v_linlist()	Compute $\sum_i a_i x_i$ for a list of vectors	107
v_map()	Apply function componentwise to a vector	104
v_max()	Computes max vector entry & index	104
v_min()	Computes min vector entry & index	104
v_mltadd()	Computes $y \leftarrow \alpha x + y$ for vectors x, y	102
vm_mlt()	Computes $x^T A$	96
vm_mltadd()	Computes $y^T \leftarrow y^T + x^T A$	96
v_norm1()	Computes $\ x\ _1$ for a vector	109
v_norm2()	Computes $\ x\ _2$ (the Euclidean norm) of a vector	109
v_norm_inf()	Computes $\ x\ _\infty$ for a vector	109
v_ones()	Set vector to all 1's	73
v_output()	Output vector to stdout (macro)	65
v_pconv()	Periodic convolution of two vectors	104
v_rand()	Randomise entries of a vector	73
v_resize()	Resize a vector	77
v_resize_vars()	Resize a list of vectors	77
v_save()	Save a vector in MATLAB format	91
v_slash()	Computes componentwise ratio of vectors	104
v_sort()	Sorts vector components	104
v_star()	Componentwise vector product	104
v_sub()	Subtract two vectors	102
v_sum()	Sum of components of a vector	104

Function	Description	Page
<code>v_zero()</code>	Zero a vector	73
<code>zabs()</code>	Complex absolute value (modulus)	111
<code>zadd()</code>	Add complex numbers	111
<code>zconj()</code>	Conjugate complex number	111
<code>zdiv()</code>	Divide complex numbers	111
<code>zexp()</code>	Complex exponential	111
<code>z_finput()</code>	Read complex number from file or stream	62
<code>z_foutput()</code>	Prints complex number to file or stream	65
<code>zgivens()</code>	Compute complex Givens' rotation	130
<code>zhhtrcols()</code>	Apply Householder transformation: PA (complex)	133
<code>zhhtrows()</code>	Apply Householder transformation: AP (complex)	133
<code>zhhtvec()</code>	Apply Householder transformation: Px (complex)	133
<code>zhhvec()</code>	Compute Householder transformation	133
<code>zin_prod()</code>	Complex inner product	111
<code>z_input()</code>	Read complex number from <code>stdin</code>	62
<code>zinv()</code>	Computes $1/z$ (complex)	111
<code>zLAsolve()</code>	Solve $L^*x = b$, L complex lower triangular	135
<code>zlog()</code>	Complex logarithm	111
<code>zLsolve()</code>	Solve $Lx = b$, L complex lower triangular	135
<code>zLUAsolve()</code>	Solve $A^*x = b$ using complex LU factorisation	122
<code>zLUcondest()</code>	Complex LU condition estimate	122
<code>zLUfactor()</code>	Complex LU factorisation	122
<code>zLUsolve()</code>	Solve $Ax = b$ using complex LU factorisation	122
<code>zm_add()</code>	Add complex matrices	81
<code>zm_adjoint()</code>	Computes adjoint of complex matrix	93
<code>zmake()</code>	Construct complex number from real and imaginary parts	111
<code>zmakeQ()</code>	Construct Q matrix for complex QR	129
<code>zmakeR()</code>	Construct R matrix for complex QR	129
<code>zmam_mlt()</code>	Computes A^*B (complex)	93
<code>zm_dump()</code>	Dump complex matrix to stream	65
<code>zm_finput()</code>	Input complex matrix from stream	62
<code>ZM_FREE()</code>	Free (deallocate) complex matrix (macro)	68

Function	Description	Page
<code>zm_free()</code>	Free (deallocate) complex matrix (function)	68
<code>zm_free_vars()</code>	Free a list of complex matrices	68
<code>zm_get()</code>	Allocate complex matrix	70
<code>zm_get_vars()</code>	Allocate a list of complex matrices	70
<code>zm_input()</code>	Input complex matrix from <code>stdin</code>	62
<code>zm_inverse()</code>	Compute inverse of complex matrix	122
<code>zm_load()</code>	Load complex matrix in MATLAB format	91
<code>zmlt()</code>	Multiply complex numbers	111
<code>zmma_mlt()</code>	Computes AB^* (complex)	93
<code>zm_mlt()</code>	Multiply complex matrices	81
<code>zm_norm1()</code>	Complex matrix 1-norm	94
<code>zm_norm_frob()</code>	Complex matrix Frobenius norm	94
<code>zm_norm_inf()</code>	Complex matrix ∞ -norm	94
<code>zm_rand()</code>	Randomise complex matrix	73
<code>zm_resize()</code>	Resize complex matrix	77
<code>zm_resize_vars()</code>	Resize a list of complex matrices	77
<code>zm_save()</code>	Save complex matrix in MATLAB format	91
<code>zm_sub()</code>	Subtract complex matrices	81
<code>zmv_mlt()</code>	Complex matrix-vector multiply	96
<code>zmv_mltadd()</code>	Complex matrix-vector multiply and add	96
<code>zm_zero()</code>	Zero complex matrix	73
<code>zneg()</code>	Computes $-z$ (complex)	111
<code>z_output()</code>	Print complex number to <code>stdout</code>	65
<code>zQRCPfactor()</code>	Complex QR factorisation with column pivoting	126
<code>zQRCPsolve()</code>	Solve $Ax = b$ using complex QR factorisation	126
<code>zQRfactor()</code>	Complex QR factorisation	126
<code>zQRASolve()</code>	Solve $A^*x = b$ using complex QR factorisation	126
<code>zQRsolve()</code>	Solve $Ax = b$ using complex QR factorisation	126
<code>zrot_cols()</code>	Complex Givens' rotation of columns	130
<code>zrot_rows()</code>	Complex Givens' rotation of rows	130
<code>z_save()</code>	Save complex number in MATLAB format	91
<code>zschur()</code>	Complex Schur factorisation	139
<code>zset_col()</code>	Set column of complex matrix	101
<code>zset_row()</code>	Set row of complex matrix	101

Function	Description	Page
<code>zsm_mlt()</code>	Complex scalar–matrix product	81
<code>zsqrt()</code>	Square root \sqrt{z} (complex)	111
<code>zsub()</code>	Subtract complex numbers	111
<code>zUAsolve()</code>	Solve $U^*x = b$, U complex upper triangular	135
<code>zUsolve()</code>	Solve $Ux = b$, U complex upper triangular	135
<code>zv_add()</code>	Add complex vectors	102
<code>zv_copy()</code>	Copy complex vector	59
<code>zv_dump()</code>	Dump complex vector to a stream	65
<code>zv_finput()</code>	Input complex vector from a stream	62
<code>ZV_FREE()</code>	Free (deallocate) complex vector (macro)	68
<code>zv_free()</code>	Free (deallocate) complex vector (function)	68
<code>zv_free_vars()</code>	Free a list of complex vectors	68
<code>zv_get()</code>	Allocate complex vector	70
<code>zv_get_vars()</code>	Allocate a list of complex vectors	70
<code>zv_input()</code>	Input complex vector from a <code>stdin</code>	62
<code>zv_lincomb()</code>	Compute $\sum_i a_i x_i$ for an array of vectors	107
<code>zv_linlist()</code>	Compute $\sum_i a_i x_i$ for a list of vectors	107
<code>zv_map()</code>	Apply function componentwise to a complex vector	104
<code>zv_mlt()</code>	Complex scalar–vector product	102
<code>zv_mltadd()</code>	Complex scalar–vector multiply and add	102
<code>zvm_mlt()</code>	Computes A^*x (complex)	96
<code>zvm_mltadd()</code>	Computes $A^*x + y$ (complex)	96
<code>zv_norm1()</code>	Complex vector 1–norm	109
<code>zv_norm2()</code>	Complex vector 2– (or Euclidean) norm	109
<code>zv_norm_inf()</code>	Complex vector ∞ – (or supremum) norm	109
<code>zv_rand()</code>	Randomise complex vector	73
<code>zv_resize()</code>	Resize complex vector	77
<code>zv_resize_vars()</code>	Resize a list of complex vectors	77
<code>zv_save()</code>	Save complex vector in MATLAB format	91
<code>zv_slash()</code>	Componentwise ratio of complex vectors	104
<code>zv_star()</code>	Componentwise product of complex vectors	104

Function	Description	Page
<code>zv_sub()</code>	Subtract complex vectors	102
<code>zv_sum()</code>	Sum of components of a complex vector	104
<code>zv_zero()</code>	Zero complex vector	73

Low level routines

Function	Description	Page
<code>__add__()</code>	Add arrays	113
<code>__ip__()</code>	Inner product of arrays	113
<code>MEM_COPY()</code>	Copy memory (macro)	113
<code>MEM_ZERO()</code>	Zero memory (macro)	113
<code>__mltadd__()</code>	Forms $x + \alpha y$ for arrays	113
<code>__smlt__()</code>	Scalar-vector multiplication for arrays	113
<code>__sub__()</code>	Subtract an array from another	113
<code>__zadd__()</code>	Add complex arrays	113
<code>__zconj__()</code>	Conjugate complex array	113
<code>__zero__()</code>	Zero an array	113
<code>__zip__()</code>	Complex inner product of arrays	113
<code>__zmlt__()</code>	Complex array scalar product	113
<code>__zmltadd__()</code>	Complex array saxpy	113
<code>__zsub__()</code>	Subtract complex arrays	113
<code>__zzero__()</code>	Zero a complex array	113