List of Contents

Chapter 1: Utilities

Module 1.1: `nag_lib_support`
Library Support Facilities

`nag_lib_ident`	Prints details of the Library implementation
`nag_deallocate`	Deallocates storage from structures with types defined by the Library

Module 1.2: `nag_error_handling`
Error Handling

`nag_set_error`	Controls how errors are to be handled by the Library
`nag_error`	Communicates information about error handling between a user's program and the Library (type)

Module 1.3: `nag_write_mat`
Matrix Printing

`nag_write_gen_mat`	Writes a real, complex or integer general matrix
`nag_write_tri_mat`	Writes a real or complex triangular matrix
`nag_write_bnd_mat`	Writes a real or complex band matrix

Module 1.4: `nag_sort`
Sorting

`nag_sort_vec`	Sorts a vector of numeric or character data into ascending or descending order
`nag_rank_vec`	Ranks a vector of numeric or character data in ascending or descending order
`nag_reorder_vec`	Reorders a vector of numeric or character data into the order specified by a vector of ranks
`nag_rank_mat`	Ranks the rows or columns of a matrix of integer or real numbers in ascending or descending order
`nag_rank_arb_data`	Ranks arbitrary data according to a user-supplied comparison procedure
`nag_invert_perm`	Inverts a permutation, thus converts a rank vector to an index vector, or vice-versa
`nag_check_perm`	Checks the validity of a permutation
`nag_decomp_perm`	Decomposes a permutation into cycles, as an aid to reordering ranked data

Module 1.5: `nag_math_constants`
Mathematical Constants

`nag_pi`	Returns an approximation to π
`nag_euler_constant`	Returns an approximation to γ (Euler's constant)

Chapter 3: Special Functions

Module 3.1: `nag_inv_hyp_fun`
Inverse Hyperbolic Functions

`nag_arctanh`	Inverse hyperbolic tangent, $\operatorname{arctanh} x$
`nag_arcsinh`	Inverse hyperbolic sine, $\operatorname{arcsinh} x$
`nag_arccosh`	Inverse hyperbolic cosine, $\operatorname{arccosh} x$

Module 3.2: `nag_gamma_fun`
Gamma Functions

`nag_gamma`	Gamma function
`nag_log_gamma`	Log gamma function
`nag_polygamma`	Polygamma functions
`nag_incompl_gamma`	Incomplete gamma functions

Module 3.3: `nag_err_fun`
Error Functions

`nag_erf`	Error function erf x
`nag_erfc`	Complementary error function erfc x
`nag_dawson`	Dawson's integral $F(x)$

Module 3.4: `nag_bessel_fun`
Bessel Functions

`nag_bessel_j0`	Bessel function $J_0(x)$
`nag_bessel_j1`	Bessel function $J_1(x)$
`nag_bessel_j`	Bessel function $J_\nu(z)$
`nag_bessel_y0`	Bessel function $Y_0(x)$
`nag_bessel_y1`	Bessel function $Y_1(x)$
`nag_bessel_y`	Bessel function $Y_\nu(z)$
`nag_bessel_i0`	Modified Bessel function $I_0(x)$
`nag_bessel_i1`	Modified Bessel function $I_1(x)$
`nag_bessel_i`	Modified Bessel function $I_\nu(z)$
`nag_bessel_k0`	Modified Bessel function $K_0(x)$
`nag_bessel_k1`	Modified Bessel function $K_1(x)$
`nag_bessel_k`	Modified Bessel function $K_\nu(z)$

Module 3.5: `nag_fresnel_intg`
Fresnel Integrals

`nag_fresnel_s`	Fresnel integral $S(x)$
`nag_fresnel_c`	Fresnel integral $C(x)$

Module 3.6: `nag_ell_intg`
Elliptic Integrals

`nag_ell_rf`	Symmetrised elliptic integral of the first kind
`nag_ell_rc`	Degenerate form of elliptic integral of the first kind
`nag_ell_rd`	Symmetrised elliptic integral of the second kind
`nag_ell_rj`	Symmetrised elliptic integral of the third kind

Module 3.7: `nag_ell_fun`
Elliptic Functions

`nag_ell_jac`	Jacobian elliptic functions sn, cn and dn

Module 3.8: `nag_airy_fun`
Airy Functions

`nag_airy_ai`	Airy function $\mathrm{Ai}(z)$
`nag_airy_bi`	Airy function $\mathrm{Bi}(z)$

Module 3.9: `nag_kelvin_fun`
Kelvin Functions

`nag_kelvin_ber`	Kelvin function ber x
`nag_kelvin_bei`	Kelvin function bei x
`nag_kelvin_ker`	Kelvin function ker x
`nag_kelvin_kei`	Kelvin function kei x

Chapter 4: Matrix and Vector Operations

Module 4.1: `nag_mat_norm`
Norms of a Matrix

`nag_gen_mat_norm`	Computes a norm, or the element of largest absolute value, of a general real or complex matrix
`nag_gen_bnd_mat_norm`	Computes a norm, or the element of largest absolute value, of a real or complex square banded matrix
`nag_sym_mat_norm`	Computes a norm, or the element of largest absolute value, of a real or complex, symmetric or Hermitian matrix, stored in conventional or packed storage
`nag_sym_bnd_mat_norm`	Computes a norm, or the element of largest absolute value, of a real or complex, symmetric or Hermitian band matrix
`nag_trap_mat_norm`	Computes a norm, or the element of largest absolute value, of a real or complex trapezoidal matrix
`nag_tri_mat_norm`	Computes a norm, or the element of largest absolute value, of a real or complex triangular matrix, stored in conventional or packed storage
`nag_tri_bnd_mat_norm`	Computes a norm, or the element of largest absolute value, of a real or complex triangular band matrix
`nag_hessen_mat_norm`	Computes a norm, or the element of largest absolute value, of a real or complex upper Hessenberg matrix

Module 4.2: `nag_mat_inv`
Matrix Inversion

`nag_gen_mat_inv`	Computes the inverse of a general real or complex matrix
`nag_gen_mat_inv_fac`	Computes the inverse of a general real or complex matrix, with the matrix previously factorized using `nag_gen_lin_fac`
`nag_sym_mat_inv`	Computes the inverse of a real or complex, symmetric or Hermitian matrix
`nag_sym_mat_inv_fac`	Computes the inverse of a real or complex, symmetric or Hermitian matrix, with the matrix previously factorized using `nag_sym_lin_fac`
`nag_tri_mat_inv`	Computes the inverse of a real or complex triangular matrix

Module 4.3: `nag_sparse_mat`
Sparse Matrix Utilities

`nag_sparse_mat_init_coo`	Initializes a sparse matrix data structure from COO format
`nag_sparse_mat_init_csc`	Initializes a sparse matrix data structure from CSC format
`nag_sparse_mat_init_csr`	Initializes a sparse matrix data structure from CSR format
`nag_sparse_mat_init_dia`	Initializes a sparse matrix data structure from DIA format
`nag_sparse_mat_extract`	Extracts details of a sparse matrix from a structure of type `nag_sparse_mat_real_`*wp* or `nag_sparse_mat_cmplx_`*wp*
`nag_sparse_matvec`	Matrix–vector multiply for a sparse matrix
`nag_sparse_mat_real_`*wp*	Represents a real sparse matrix (type)
`nag_sparse_mat_cmplx_`*wp*	Represents a complex sparse matrix (type)

Chapter 5: Linear Equations

Module 5.1: `nag_gen_lin_sys`
General Systems of Linear Equations

`nag_gen_lin_sol`	Solves a general real or complex system of linear equations with one or many right-hand sides
`nag_gen_lin_fac`	Performs an *LU* factorization of a general real or complex matrix
`nag_gen_lin_sol_fac`	Solves a general real or complex system of linear equations, with coefficient matrix previously factorized by `nag_gen_lin_fac`

Module 5.2: `nag_sym_lin_sys`
Symmetric Systems of Linear Equations

`nag_sym_lin_sol`	Solves a real or complex, symmetric or Hermitian system of linear equations with one or many right-hand sides
`nag_sym_lin_fac`	Performs a Cholesky or Bunch–Kaufman factorization of a real or complex, symmetric or Hermitian matrix
`nag_sym_lin_sol_fac`	Solves a real or complex, symmetric or Hermitian system of linear equations, with coefficient matrix previously factorized by `nag_sym_lin_fac`

Module 5.3: `nag_tri_lin_sys`
Triangular Systems of Linear Equations

`nag_tri_lin_sol`	Solves a real or complex triangular system of linear equations
`nag_tri_lin_cond`	Estimates the condition number of a real or complex triangular matrix
`nag_tri_mat_det`	Evaluates the determinant of a real or complex triangular matrix

Module 5.4: `nag_gen_bnd_lin_sys`
General Banded Systems of Linear Equations

`nag_gen_bnd_lin_sol`	Solves a general real or complex banded system of linear equations, with one or many right-hand sides
`nag_gen_bnd_lin_fac`	Performs an *LU* factorization of a general real or complex band matrix
`nag_gen_bnd_lin_sol_fac`	Solves a general real or complex banded system of linear equations, with coefficient matrix previously factorized by `nag_gen_bnd_lin_fac`

Module 5.5: `nag_sym_bnd_lin_sys`
Symmetric Banded Systems of Linear Equations

`nag_sym_bnd_lin_sol`	Solves a real symmetric or complex Hermitian positive definite banded system of linear equations, with one or many right-hand sides
`nag_sym_bnd_lin_fac`	Performs a Cholesky factorization of a real symmetric or complex Hermitian positive definite band matrix
`nag_sym_bnd_lin_sol_fac`	Solves a real symmetric or complex Hermitian positive definite banded system of linear equations, with coefficient matrix previously factorized by `nag_sym_bnd_lin_fac`

Module 5.6: `nag_sparse_prec`
Sparse Matrix Preconditioner Set-up and Solve

`nag_sparse_prec_init_jac`	Initializes sparse Jacobi preconditioner
`nag_sparse_prec_init_ssor`	Initializes sparse SSOR preconditioner
`nag_sparse_prec_init_ilu`	Initializes sparse ILU preconditioner for real non-symmetric or complex non-Hermitian matrices
`nag_sparse_prec_sol`	Sparse matrix preconditioned system solver

Module 5.7: nag_sparse_lin_sys
Sparse Linear System Iterative Solvers

nag_sparse_gen_lin_sol	General sparse linear system solver

Chapter 6: Eigenvalue and Least-squares Problems

Module 6.1: nag_sym_eig
Standard Symmetric Eigenvalue Problems

nag_sym_eig_all	All eigenvalues, and optionally eigenvectors, of a real symmetric or complex Hermitian matrix
nag_sym_eig_sel	Selected eigenvalues, and optionally the corresponding eigenvectors, of a real symmetric or complex Hermitian matrix
nag_sym_tridiag_reduc	Reduction of a real symmetric or complex Hermitian matrix to real symmetric tridiagonal form
nag_sym_tridiag_orth	Form or apply the transformation matrix determined by nag_sym_tridiag_reduc
nag_sym_tridiag_eig_all	All eigenvalues, and optionally eigenvectors, of a real symmetric tridiagonal matrix
nag_sym_tridiag_eig_val	Selected eigenvalues of a real symmetric tridiagonal matrix
nag_sym_tridiag_eig_vec	Selected eigenvectors of a real symmetric tridiagonal matrix

Module 6.2: nag_nsym_eig
Standard Nonsymmetric Eigenvalue Problems

nag_nsym_eig_all	All eigenvalues, and optionally eigenvectors, of a general real or complex matrix
nag_schur_fac	Schur factorization of a general real or complex matrix

Module 6.3: nag_svd
Singular Value Decomposition (SVD)

nag_gen_svd	Singular value decomposition of a general real or complex matrix
nag_gen_bidiag_reduc	Reduction of a general real or complex matrix to real bidiagonal form
nag_bidiag_svd	Singular value decomposition of a real bidiagonal matrix

Module 6.4: nag_lin_lsq
Linear Least-squares Problems

nag_lin_lsq_sol	Solves a real or complex linear least-squares problem
nag_lin_lsq_sol_svd	Solves a real or complex linear least-squares problem, assuming that a singular value decomposition of the coefficient matrix has already been computed
nag_qr_fac	QR factorization of a general real or complex matrix
nag_qr_orth	Form or apply the matrix Q determined by nag_qr_fac
nag_lin_lsq_sol_qr	Solves a real or complex linear least-squares problem, assuming that the QR factorization of the coefficient matrix has already been computed
nag_lin_lsq_sol_qr_svd	Solves a real or complex linear least-squares problem using the SVD, assuming that the QR factorization of the coefficient matrix has already been computed

Module 6.5: nag_sym_gen_eig
Symmetric-definite Generalized Eigenvalue Problems

nag_sym_gen_eig_all	All eigenvalues, and optionally eigenvectors, of a real symmetric-definite or complex Hermitian-definite generalized eigenvalue problem
nag_sym_gen_eig_sel	Selected eigenvalues, and optionally the corresponding eigenvectors, of a real symmetric-definite or complex Hermitian-definite generalized eigenvalue problem

Module 6.6: nag_nsym_gen_eig
Nonsymmetric Generalized Eigenvalue Problems

nag_nsym_gen_eig_all	All eigenvalues, and optionally eigenvectors, of a real or complex nonsymmetric generalized eigenvalue problem
nag_gen_schur_fac	Generalized Schur factorization of a real or complex matrix pencil

Chapter 7: Transforms

Module 7.1: nag_fft
Discrete Fourier Transforms

nag_fft_1d	Single or multiple 1-d complex discrete Fourier transform, or its inverse
nag_fft_1d_real	Single or multiple 1-d real or Hermitian discrete Fourier transform, or its inverse
nag_fft_1d_basic	Single or multiple 1-d real, Hermitian or complex discrete Fourier transform, which is overwritten on the input data
nag_fft_2d	2-d complex discrete Fourier transform, or its inverse
nag_fft_2d_basic	2-d complex discrete Fourier transform, which is overwritten on the input data
nag_fft_3d	3-d complex discrete Fourier transform, or its inverse
nag_fft_3d_basic	3-d complex discrete Fourier transform, which is overwritten on the input data
nag_fft_trig	Trigonometric coefficients for computing discrete Fourier transforms
nag_herm_to_cmplx	Convert Hermitian sequences to general complex sequences
nag_cmplx_to_herm	Convert Hermitian complex sequences to their compact real form
nag_conj_herm	Complex conjugates of Hermitian sequences

Module 7.2: nag_sym_fft
Symmetric Discrete Fourier Transforms

nag_fft_sin	Single or multiple 1-d discrete Fourier sine transform
nag_fft_cos	Single or multiple 1-d discrete Fourier cosine transform
nag_fft_qtr_sin	Single or multiple 1-d discrete quarter-wave Fourier sine transform, or its inverse
nag_fft_qtr_cos	Single or multiple 1-d discrete quarter-wave Fourier cosine transform, or its inverse

Module 7.3: nag_conv
Convolution and Correlation

nag_fft_conv	Computes the convolution or correlation of two real or complex vectors

Chapter 8: Curve and Surface Fitting

Module 8.1: nag_pch_interp
Piecewise Cubic Hermite Interpolation

nag_pch_monot_interp	Generates a monotonicity-preserving piecewise cubic Hermite interpolant
nag_pch_eval	Computes values and optionally derivatives of a piecewise cubic Hermite interpolant
nag_pch_intg	Computes the definite integral of a piecewise cubic Hermite interpolant
nag_pch_extract	Extracts details of a piecewise cubic Hermite interpolant from a structure of type nag_pch_comm_*wp*
nag_pch_comm_*wp*	Represents a piecewise cubic Hermite interpolant (type)

Module 8.2: nag_spline_1d
One-dimensional Spline Fitting

nag_spline_1d_auto_fit	Generates a cubic spline approximation to an arbitrary 1-d data set, with automatic knot selection
nag_spline_1d_lsq_fit	Generates a weighted least-squares cubic spline fit to an arbitrary 1-d data set, with given interior knots
nag_spline_1d_interp	Generates a cubic spline interpolant to an arbitrary 1-d data set
nag_spline_1d_eval	Computes values of a cubic spline and optionally its first three derivatives
nag_spline_1d_intg	Computes the definite integral of a cubic spline
nag_spline_1d_set	Initializes a cubic spline with given interior knots and B-spline coefficients
nag_spline_1d_extract	Extracts details of a cubic spline from a structure of type nag_spline_1d_comm_*wp*
nag_spline_1d_comm_*wp*	Represents a 1-d cubic spline in B-spline series form (type)

Module 8.3: nag_spline_2d
Two-dimensional Spline Fitting

nag_spline_2d_auto_fit	Generates a bicubic spline approximation to a 2-d data set, with automatic knot selection
nag_spline_2d_lsq_fit	Generates a minimal, weighted least-squares bicubic spline surface fit to a given set of data points, with given interior knots
nag_spline_2d_interp	Generates a bicubic spline interpolating surface through a set of data values, given on a rectangular grid of the xy plane
nag_spline_2d_eval	Computes values of a bicubic spline
nag_spline_2d_intg	Computes the definite integral of a bicubic spline
nag_spline_2d_set	Initializes a bicubic spline with given interior knots and B-spline coefficients
nag_spline_2d_extract	Extracts details of a bicubic spline from a structure of type nag_spline_2d_comm_*wp*
nag_spline_2d_comm_*wp*	Represents a 2-d bicubic spline in B-spline series form (type)

Module 8.4: nag_scat_interp
Interpolation of Scattered Data

nag_scat_2d_interp	Generates a 2-d interpolating function using a modified Shepard method
nag_scat_2d_eval	Computes values of the interpolant generated by nag_scat_2d_interp and its partial derivatives
nag_scat_3d_interp	Generates a 3-d interpolating function using a modified Shepard method
nag_scat_3d_eval	Computes values of the interpolant generated by nag_scat_3d_interp and its partial derivatives
nag_scat_2d_set	Initializes a structure of type nag_scat_comm_*wp* to represent a 2-d scattered data interpolant
nag_scat_3d_set	Initializes a structure of type nag_scat_comm_*wp* to represent a 3-d scattered data interpolant
nag_scat_extract	Extracts details of a scattered data interpolant from a structure of derived type nag_scat_comm_*wp*
nag_scat_comm_*wp*	Represents a scattered data interpolant generated either by nag_scat_2d_interp or nag_scat_3d_interp (type)

Module 8.5: nag_cheb_1d
Chebyshev Series

nag_cheb_1d_fit	Finds the least-squares fit using arbitrary data points
nag_cheb_1d_interp	Generates the coefficients of the Chebyshev polynomial which interpolates (passes exactly through) data at a special set of points
nag_cheb_1d_fit_con	Finds the least-squares fit using arbitrary data points with constraints on some data points
nag_cheb_1d_eval	Evaluation of fitted polynomial in one variable, from Chebyshev series form
nag_cheb_1d_deriv	Derivatives of fitted polynomial in Chebyshev series form
nag_cheb_1d_intg	Integral of fitted polynomial in Chebyshev series form

Chapter 9: Optimization

Module 9.1: nag_qp
Linear and Quadratic Programming

nag_qp_sol	Solves a linear or quadratic programming problem
nag_qp_cntrl_init	Initialization procedure for nag_qp_cntrl_*wp*
nag_qp_cntrl_*wp*	Control parameters for nag_qp_sol (type)

Module 9.2: nag_nlin_lsq
Unconstrained Nonlinear Least-squares

nag_nlin_lsq_sol	Finds an unconstrained minimum of a sum of squares
nag_nlin_lsq_cov	Computes the variance-covariance matrix for a nonlinear least-squares problem
nag_nlin_lsq_cntrl_init	Initialization procedure for nag_nlin_lsq_cntrl_*wp*
nag_nlin_lsq_cntrl_*wp*	Control parameters for nag_nlin_lsq_sol (type)

Module 9.3: nag_nlp
Nonlinear Programming

nag_nlp_sol	Solves a dense nonlinear programming problem
nag_nlp_cntrl_init	Initialization procedure for nag_nlp_cntrl_*wp*
nag_nlp_cntrl_*wp*	Control parameters for nag_nlp_sol (type)

Module 9.4: `nag_con_nlin_lsq`
Constrained Nonlinear Least-squares

`nag_con_nlin_lsq_sol`	Finds a constrained minimum of a sum of squares
`nag_con_nlin_lsq_sol_1`	Finds a constrained minimum of a sum of squares
`nag_con_nlin_lsq_cntrl_init`	Initialization procedure for `nag_con_nlin_lsq_cntrl_`*wp*
`nag_con_nlin_lsq_cntrl_`*wp*	Control parameters for `nag_con_nlin_lsq_sol` and `nag_con_nlin_lsq_sol_1` (type)

Module 9.5: `nag_uv_min`
Univariate Minimization

`nag_uv_min_sol`	Finds the minimum of a continuous function of a single variable in a given finite interval

Module 9.6: `nag_nlp_sparse`
Sparse Nonlinear Programming

`nag_nlp_sparse_sol`	Solves a sparse nonlinear programming problem
`nag_nlp_sparse_cntrl_init`	Initialization procedure for `nag_nlp_sparse_cntrl_`*wp*
`nag_nlp_sparse_cntrl_`*wp*	Control parameters for `nag_nlp_sparse_sol` (type)

Chapter 10: Nonlinear Equations

Module 10.1: `nag_polynom_eqn`
Roots of Polynomials

`nag_polynom_roots`	Calculates the roots of a polynomial

Module 10.2: `nag_nlin_eqn`
Roots of a Single Nonlinear Equation

`nag_nlin_eqn_sol`	Finds a solution of a single nonlinear equation

Module 10.3: `nag_nlin_sys`
Roots of a System of Nonlinear Equations

`nag_nlin_sys_sol`	Finds a solution of a system of nonlinear equations

Chapter 11: Quadrature

Module 11.1: `nag_quad_1d`
Numerical Integration over a Finite Interval

`nag_quad_1d_gen`	1-d quadrature, adaptive, finite interval, allowing for badly behaved integrand, allowing for singularities at user-specified break-points, suitable for oscillatory integrands
`nag_quad_1d_wt_trig`	1-d quadrature, adaptive, finite interval, weight function $\cos(\omega x)$ or $\sin(\omega x)$
`nag_quad_1d_wt_end_sing`	1-d quadrature, adaptive, finite interval, weight function with end-point singularities of algebraico-logarithmic type
`nag_quad_1d_wt_hilb`	1-d quadrature, adaptive, finite interval, weight function $1/(x-c)$, Cauchy principal value (Hilbert transform)
`nag_quad_1d_data`	1-d quadrature, integration of function defined by data values, Gill–Miller method

Module 11.2: `nag_quad_1d_inf`
Numerical Integration over an Infinite Interval

`nag_quad_1d_inf_gen`	1-d quadrature, adaptive, semi-infinite or infinite interval
`nag_quad_1d_inf_wt_trig`	1-d quadrature, adaptive, semi-infinite interval, weight function $\cos(\omega x)$ or $\sin(\omega x)$

Module 11.3: `nag_quad_md`
Multi-dimensional Integrals

`nag_quad_md_rect`	Multi-dimensional adaptive quadrature over a hyper-rectangle
`nag_quad_md_rect_mintg`	Multi-dimensional adaptive quadrature over a hyper-rectangle, multiple integrands
`nag_quad_2d`	2-d quadrature, finite region
`nag_quad_monte_carlo`	Multi-dimensional quadrature over hyper-rectangle, Monte-Carlo method

Module 11.4: `nag_quad_util`
Numerical Integration Utilities

`nag_quad_gs_wt_absc`	Calculation of weights and abscissae for Gaussian quadrature rules, general choice of rule

Chapter 12: Ordinary Differential Equations

Module 12.1: `nag_ivp_ode_rk`
Solution of Initial Value Problems for Ordinary Differential Equations by Runge–Kutta Methods

`nag_rk_setup`	Sets up the integration
`nag_rk_interval`	Integrates across an interval and provides the solution at user-specified points
`nag_rk_info`	Provides statistics about the integration
`nag_rk_global_err`	Provides information about global error assessment
`nag_rk_step`	Integrates one step at a time
`nag_rk_interp`	Interpolates the solution
`nag_rk_reset_end`	Resets the end point of integration
`nag_rk_comm_`*wp*	Communicating structure for `nag_ivp_ode_rk` (type)

Chapter 13: Partial Differential Equations

Module 13.1: `nag_pde_helm`
Helmholtz Equations

`nag_pde_helm_3d`	Solves the 3-d Helmholtz equation using a standard seven-point finite difference scheme and a fast Fourier transform method

Module 13.2: `nag_pde_ell_mg`
Multigrid Solution of Elliptic Partial Differential Equations

`nag_pde_ell_rect`	Generates a seven-diagonal system of linear equations which arises from the discretization of a two-dimensional elliptic partial differential equation on a rectangle
`nag_pde_ell_mg_sol`	Solves a seven-diagonal system of linear equations using a multigrid iteration

Module 13.3: `nag_pde_parab_1d`

Parabolic Partial Differential Equations in One Space Variable

`nag_pde_parab_1d_fd`	Integrates a system of parabolic PDEs in one space variable, coupled with ODEs; using finite differences for the spatial discretisation with optional automatic adaptive spatial remeshing
`nag_pde_interp_1d_fd`	Interpolates the solution and first derivative of a system of partial differential equations solved using finite differences, at a set of user-specified points
`nag_pde_parab_1d_coll`	Integrates a system of parabolic PDEs in one space variable, coupled with ODEs; using a Chebyshev C^0 collocation method for the spatial discretisation
`nag_pde_interp_1d_coll`	Interpolates the solution and first derivative of a system of partial differential equations solved using a Chebyshev C^0 collocation method, at a set of user-specified points
`nag_pde_parab_1d_cntrl_init`	Initialization procedure for type `nag_pde_parab_1d_cntrl_`*wp*
`nag_pde_parab_1d_comm_`*wp*	Communicates arrays for the underlying ODE solver between calls to the procedures in `nag_pde_parab_1d` (type)
`nag_pde_parab_1d_cntrl_`*wp*	Control parameters for procedures `nag_pde_parab_1d_fd` and `nag_pde_parab_1d_coll` (type)

Chapter 19: Operations Research

Module 19.1: `nag_ip`

Integer Programming

`nag_ip_sol`	Solves 'zero-one', 'general', 'mixed' or 'all' integer linear programming problems
`nag_ip_cntrl_init`	Initialization procedure for `nag_ip_cntrl_`*wp*
`nag_ip_cntrl_`*wp*	Control parameters for `nag_ip_sol` (type)

Module 19.2: `nag_short_path`

Shortest Path Problems

`nag_short_path_find`	Finds the shortest path through a directed or undirected acyclic network

Chapter 20: Statistical Distribution Functions

Module 20.1: `nag_normal_dist`

Probabilities and Deviate for a Normal Distribution

`nag_normal_prob`	Computes probabilities for various parts of a univariate Normal distribution
`nag_normal_deviate`	Computes the deviate associated with a given probability of a standard Normal distribution
`nag_bivar_normal_prob`	Computes the lower tail probability for a bivariate Normal distribution
`nag_mv_normal_prob`	Computes probabilities for various parts of a multivariate Normal distribution

Module 20.2: `nag_t_dist`

Probabilities and Deviate for a Student's t-distribution

`nag_t_prob`	Computes probabilities for various parts of a Student's t-distribution with ν degrees of freedom
`nag_t_deviate`	Computes the deviate associated with a given probability of a Student's t-distribution

Module 20.3: `nag_chisq_dist`

Probabilities and Deviate for a χ^2-distribution

`nag_chisq_prob` Computes lower or upper tail probability for a χ^2-distribution with ν degrees of freedom

`nag_chisq_deviate` Computes the deviate associated with a given lower tail probability of a χ^2-distribution with ν degrees of freedom

Module 20.4: `nag_f_dist`

Probabilities and Deviate for an F-distribution

`nag_f_prob` Computes lower or upper tail probability for an F-distribution with ν_1 and ν_2 degrees of freedom

`nag_f_deviate` Computes the deviate associated with a given lower tail probability of an F-distribution with ν_1 and ν_2 degrees of freedom

Module 20.5: `nag_beta_dist`

Probabilities and Deviate for a Beta Distribution

`nag_beta_prob` Computes lower or upper tail probability for a beta distribution with parameters a and b

`nag_beta_deviate` Computes the deviate associated with a given lower tail probability of a beta distribution with parameters a and b

Module 20.6: `nag_gamma_dist`

Probabilities and Deviate for a Gamma Distribution

`nag_gamma_prob` Computes lower or upper tail probability for a gamma distribution with shape parameter a and scale parameter b

`nag_gamma_deviate` Computes the deviate associated with a given lower tail probability of a gamma distribution with shape parameter a and scale parameter b

Module 20.7: `nag_discrete_dist`

Probabilities for Discrete Distributions

`nag_binom_prob` Computes lower tail, upper tail or point probability for a binomial distribution with parameters n and p

`nag_poisson_prob` Computes lower tail, upper tail or point probability for a Poisson distribution with parameter λ

`nag_hypergeo_prob` Computes lower tail, upper tail or point probability for a hypergeometric distribution with parameters n, l, and m

Chapter 21: Random Number Generation

Module 21.1: `nag_rand_util`

Utilities for Random Number Generation

`nag_rand_seed_set` Sets the seed used by random number generating procedures to give a repeatable or non-repeatable sequence of random numbers

`nag_seed_`*wp* Stores data required to generate successive random numbers from a given stream (type)

Module 21.2: nag_rand_contin
Random Numbers from Continuous Distributions

nag_rand_uniform — Generates random numbers from a uniform distribution over (a, b)

nag_rand_normal — Generates random numbers from a Normal distribution with mean a and standard deviation b

nag_rand_mv_normal — Generates a vector of n random numbers from a multivariate Normal distribution with mean vector a and covariance matrix C

nag_rand_beta — Generates random numbers from a beta distribution with parameters a and b

nag_rand_neg_exp — Generates random numbers from a (negative) exponential distribution with mean a

nag_rand_gamma — Generates random numbers from a gamma distribution with parameters a and b

Module 21.3: nag_rand_discrete
Random Numbers from Discrete Distributions

nag_rand_binom — Generates random integers from a binomial distribution and/or returns a reference vector for the distribution

nag_rand_neg_binom — Generates random integers from a negative binomial distribution and/or returns a reference vector for the distribution

nag_rand_hypergeo — Generates random integers from an hypergeometric distribution and/or returns a reference vector for the distribution

nag_rand_user_dist — Generates random integers and/or returns a reference vector from a discrete distribution defined in terms of its PDF or CDF

nag_rand_ref_vec — Generates random integers from a discrete distribution, using a reference vector

nag_ref_vec_*wp* — Stores a reference vector which is used to generate random integers from a discrete distribution (type)

Chapter 22: Basic Descriptive Statistics

Module 22.1: nag_basic_stats
Basic Descriptive Statistics for Univariate Data

nag_summary_stats_1v — Computes basic descriptive statistics for univariate data

Chapter 25: Correlation and Regression Analysis

Module 25.1: nag_lin_reg
Regression Analysis

nag_simple_lin_reg — Performs a simple linear regression analysis for a pair of related variables

nag_mult_lin_reg — Performs a general multiple linear regression analysis for any given predictor variables and a response variable

Module 25.2: nag_correl
Correlation Analysis

nag_prod_mom_correl — Calculates the variance-covariance matrix and the Pearson product-moment correlation coefficients for a set of data

nag_part_correl — Calculates the partial variance-covariance matrix and the partial correlation matrix from a correlation or variance covariance matrix

Chapter 28: Multivariate Analysis

Module 28.1: `nag_fac_analysis`

Factor Analysis and Principal Component

`nag_prin_comp`	Performs principal component analysis

Module 28.2: `nag_canon_analysis`

Canonical Analysis

`nag_canon_var`	Performs canonical variate analysis

Module 28.3: `nag_mv_rotation`

Rotations

`nag_orthomax`	Computes orthogonal rotation, using a generalized orthomax rotations

Chapter 29: Time Series Analysis

Module 29.1: `nag_tsa_identify`

Time Series Analysis – Identification

`nag_tsa_acf`	Calculates the sample autocorrelation function of a univariate time series
`nag_tsa_pacf`	Calculates the sample partial autocorrelation function of a univariate time series

Module 29.2: `nag_tsa_kalman`

Kalman Filtering

`nag_kalman_init`	Provides an initial estimate of the Kalman filter state covariance matrix
`nag_kalman_predict`	Calculates a one step prediction for the square root covariance Kalman filter
`nag_kalman_sqrt_cov_var`	Calculates a time-varying square root covariance Kalman filter
`nag_kalman_sqrt_cov_invar`	Calculates a time-invariant square root covariance Kalman filter

Module 29.3: `nag_tsa_spectral`

Time Series Spectral Analysis

`nag_spectral_data`	Calculates the smoothed sample spectrum of a univariate time series
`nag_spectral_cov`	Calculates the smoothed sample spectrum of a univariate time series using autocovariances data
`nag_bivar_spectral_data`	Calculates the smoothed sample cross spectrum of a bivariate time series
`nag_bivar_spectral_cov`	Calculates the smoothed sample cross spectrum of a bivariate time series using autocovariances data
`nag_bivar_spectral_coh`	Calculates the squared coherency, the cross amplitude, the gain and the phase spectra
`nag_bivar_spectral_lin_sys`	Calculates the noise spectrum and the impulse response function from a linear system