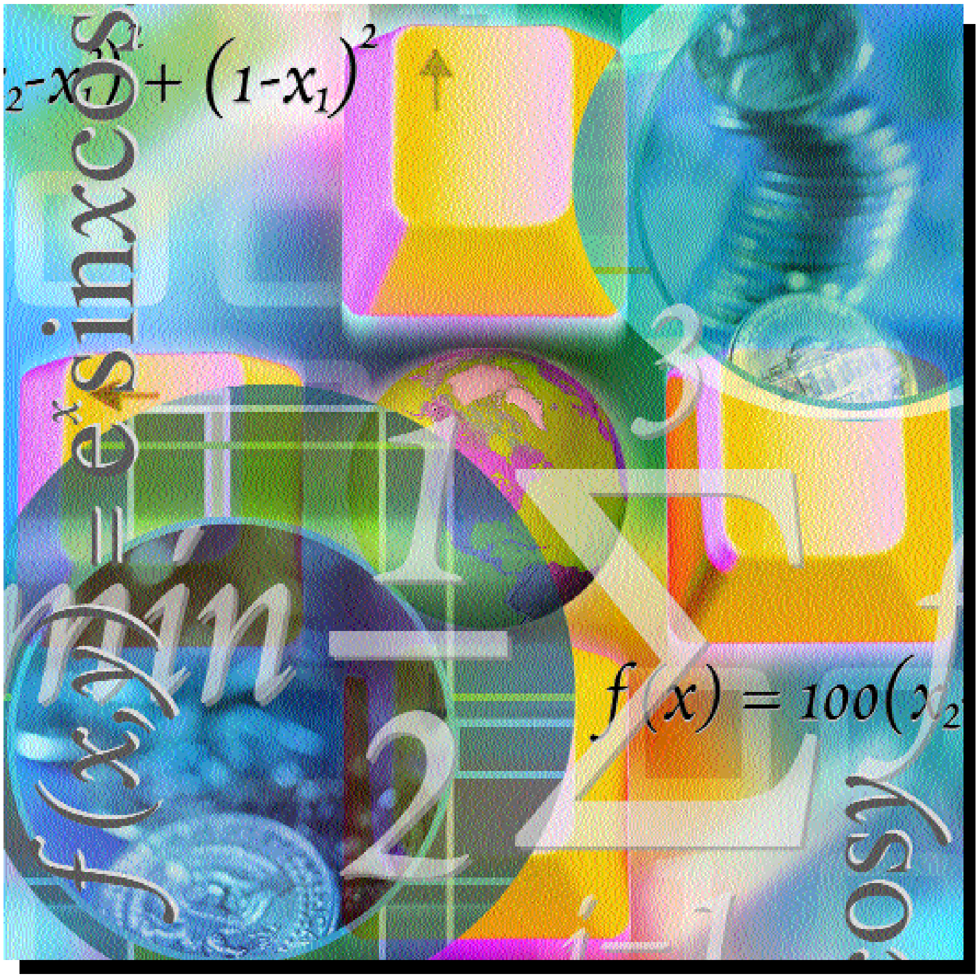
IMSL®

Fortran Library

V 5.0

Function Catalog



Trusted For Over 30 Years

Table Of Contents

IMSL Fortran Library, Version 5.0 2
IMSL – Also available for C and Java....................8
IMSL MATH/LIBRARY....................9
Chapter 1: Linear Systems....................9
Chapter 2: Eigensystem Analysis 17
Chapter 3: Interpolation and Approximation 19
Chapter 4: Integration and Differentiation 23
Chapter 5: Differential Equations 24
Chapter 6: Transforms 25
Chapter 7: Nonlinear Equations 26
Chapter 8: Optimization....................27
Chapter 9: Basic Matrix/Vector Operations....................29
Chapter 10: Linear Algebra Operators and Generic Functions 35
Chapter 11:Utilities 36
IMSL MATH/LIBRARY Special Functions 40
Chapter 1: Elementary Functions....................40
Chapter 2: Trigonometric and Hyperbolic Functions....................40
Chapter 3: Exponential Integrals and Related Functions....................41
Chapter 4: Gamma Function and Related Functions....................41
Chapter 5: Error Function and Related Functions 42
Chapter 6: Bessel Functions 42
Chapter 7: Kelvin Functions 44
Chapter 8: Airy Functions....................44
Chapter 9: Elliptic Integrals 45
Chapter 10: Elliptic and Related Functions....................45
Chapter 11: Probability Distribution Functions and Inverses 45
Chapter 12: Mathieu Functions 47
Chapter 13: Miscellaneous Functions 47
IMSL STAT/LIBRARY....................48
Chapter 1: Basic Statistics 48
Chapter 2: Regression 48
Chapter 3: Correlation 51
Chapter 4: Analysis of Variance....................51
Chapter 5: Categorical and Discrete Data Analysis....................52
Chapter 6: Nonparametric Statistics 53
Chapter 7: Tests of Goodness-of-Fit and Randomness 54
Chapter 8: Time Series Analysis and Forecasting....................54
Chapter 9: Covariance Structures and Factor Analysis 57
Chapter 10: Discriminant Analysis 58
Chapter 11: Cluster Analysis....................58
Chapter 12: Sampling....................58
Chapter 13: Survival Analysis, Life Testing and Reliability....................59
Chapter 14: Multidimensional Scaling....................60
Chapter 15: Density and Hazard Estimation....................60
Chapter 16: Line Printer Graphics....................60
Chapter 17: Probability Distribution Functions and Inverses 61
Chapter 18: Random Number Generation 63
Chapter 19: Utilities....................65
Chapter 20: Mathematical Support 67

IMSL FORTRAN LIBRARY, VERSION 5.0

Written for Fortran programmers and based on the world's most widely called numerical subroutines.

At the heart of the IMSL Libraries lies the comprehensive and trusted set of IMSL mathematical and statistical numerical algorithms. The IMSL Fortran Library Version 5.0 includes all of the algorithms from the IMSL family of Fortran libraries including the IMSL F90 Library, the IMSL FORTRAN 77 Library, and the IMSL parallel processing features. With IMSL, we provide "building blocks" which eliminate the need to write code from scratch. These pre-written functions allow you to focus on your expertise and reduce your development time.

ONE COMPREHENSIVE PACKAGE

All F77, F90 and parallel processing features are now contained within a single IMSL Fortran Library package

INTERFACE MODULES

The IMSL Fortran Library version 5.0 includes new powerful and flexible interface modules for all applicable routines. The Interface Modules accomplish the following:

- Allow for the use of advanced Fortran syntax and optional arguments throughout.
- Only require a short list of required arguments for each algorithm to facilitate development of simpler Fortran applications.
- Provide full depth and control via optional arguments for experienced programmers.
- Reduce development effort by checking data-type matches and array sizing at compile time.
- With operators and function modules, provide faster and more natural programming through an object-oriented approach.
- A simple and flexible interface to the library routines speeds programming and simplifies documentation.

The IMSL Fortran Library takes full advantage of the intrinsic characteristics and desirable features of the Fortran language.

BACKWARD COMPATIBILITY

The IMSL Fortran Library Version 5.0 maintains full backward compatibility with IMSL Fortran Libraries. No code modifications are required for existing applications that rely on previous versions of the IMSL Fortran Libraries. Calls to routines from the IMSL FORTRAN 77 Libraries with the F77 syntax continue to function.

SMP/OPENMP SUPPORT

The IMSL Fortran Library has also been designed to take advantage of symmetric multiprocessor (SMP) systems. Computationally intensive algorithms in areas such as linear algebra and fast Fourier transforms will leverage SMP capabilities on a variety of systems. By allowing you to replace the generic Basic Linear Algebra Subprograms ("BLAS") contained in the IMSL Fortran Libary with optimized BLAS from your hardware vendor, you can improve the performance of your numerical calculations.

MPI ENABLED

The IMSL Fortran Library provides a dynamic interface for computing mathematical solutions over a distributed system via Message Passing Interface (MPI). MPI enabled routines offer a simple, reliable user interface.

The IMSL Fortran library provides a number of MPI-enabled routines with an MPI-enhanced interface that provides:

- Computational control of the server node.
- Scalability of computational resources.
- Automatic processor prioritization.
- Self-scheduling algorithm to keep processors continuously active.
- Box data type application.
- Computational integrity.
- Dynamic error processing.
- Homogeneous and heterogeneous network functionality.
- Use of descriptive names and generic interfaces
- A suite of testing and benchmark software.

USER FRIENDLY NOMENCLATURE

The IMSL Fortran Library uses descriptive explanatory function names for intuitive programming.

ERROR HANDLING

Diagnostic error messages are clear and informative – designed not only to convey the error condition but also to suggest corrective action if appropriate. These error-handling features:

- Make it faster and easier for you to debug your programs.
- Provide for more productive programming and confidence that the algorithms are functioning properly in your application.

COST-EFFECTIVENESS AND VALUE

The IMSL Fortran Library significantly shortens program development time and promotes standardization. You'll find that using The IMSL Fortran Library saves time in your source code development and saves thousands of dollars in the design, development, documentation, testing and maintenance of your applications.

FULLY TESTED

Visual Numerics has developed over 30 years of experience in testing IMSL numerical algorithms for quality and performance across an extensive range of the latest compilers and environments. Visual Numerics works with compiler partners and hardware partners to ensure a high degree of reliability and performance optimization. This experience has allowed Visual Numerics to refine its test methods with painstaking detail. The result of this effort is a robust, sophisticated suite of test methods that allow the IMSL user to rely on the numerical analysis functionality and focus their bandwidth on their application development and testing.

WIDE COMPATIBILITY AND UNIFORM OPERATION

The IMSL Fortran Library is available for major UNIX computing environments, including Linux, as well as Windows NT/98/2000/XP. Visual Numerics performs extensive compatibility testing to ensure that the library is compatible with each supported computing environment.

COMPREHENSIVE DOCUMENTATION

Documentation for the IMSL Fortran Library is comprehensive, clearly written and standardized. Detailed information about each function is found in a single source within a chapter and consists of section name, purpose, synopsis, errors, return values and usage examples. Each manual's alphabetical index enables convenient cross-referencing.

IMSL documentation:

- Provides organized, easy-to-find information.
- Extensively documents, explains and provides references for algorithms.
- Online documentation provides powerful search capabilities with hundreds of code examples of function usage.

UNMATCHED PRODUCT SUPPORT

Behind every VNI license is a team of professionals ready to provide expert answers to questions about your IMSL software. Product support options include product maintenance and consultation, ensuring value and performance of your IMSL software.

Product support:

- Gives you direct access to VNI resident staff of expert product support specialists.
- Provides prompt, two-way communication with solutions to your programming needs.
- Includes product maintenance updates.
- Flexible licensing options

The IMSL Fortran Library can be licensed in a number of flexible ways: licenses may be node-locked to a specific CPU, or a specified number of licenses can be purchased to "float" throughout a heterogeneous network as they are needed. This allows you to cost-effectively acquire as many seats as you need today, adding more seats when it becomes necessary. Site licenses and campus licenses are also available.

Rely on the industry leader for software that is expertly developed, thoroughly tested, meticulously maintained and well documented. Get reliable results EVERY TIME!

MATHEMATICAL FUNCTIONS

The IMSL Fortran Library is a collection of the most commonly needed numerical functions customized for your programming needs. The mathematical functionality is organized into 11 sections. These capabilities range from solving systems of linear equations to optimization.

- **Linear Systems**, including real and complex full and sparse matrices, linear least squares, matrix decompositions, generalized inverses and vector-matrix operations.
- **Eigensystem Analysis**, including eigenvalues and eigenvectors of complex, real symmetric and complex Hermitian matrices.
- **Interpolation and Approximation**, including constrained curve-fitting splines, cubic splines, least-squares approximation and smoothing, and scattered data interpolation.
- **Integration and Differentiation**, including univariate, multivariate, Gauss quadrature and quasi-Monte Carlo.
- **Differential Equations**, using Adams-Gear and Runge-Kutta methods for stiff and nonstiff ordinary differential equations and support for partial differential equations.
- **Transforms**, including real and complex one- and two-dimensional fast Fourier transforms, as well as convolutions and correlations and Laplace transforms.
- **Nonlinear Equations**, including zeros and root finding of polynomials, zeros of a function and root of a system of equations.
- **Optimization**, including unconstrained, and linearly and nonlinearly constrained minimizations.
- **Basic Matrix/Vector Operations**, including Basic Linear Algebra Subprograms (BLAS) and matrix manipulation operations.
- **Linear Algebra Operators and Generic Functions**, including matrix algebra operations, and matrix and utility functionality.
- **Utilities**, including CPU time used, error handling and machine, mathematical, physical constants, retrieval of machine constants and changing error-handling.

MATHEMATICAL SPECIAL FUNCTIONS

The IMSL Fortran Library includes routines that evaluate the special mathematical functions that arise in applied mathematics, physics, engineering and other technical fields. The mathematical special functions are organized into 12 sections.

- **Elementary Functions**, including complex numbers, exponential functions and logarithmic functions.
- **Trigonometric and Hyperbolic Functions**, including trigonometric functions and hyperbolic functions.
- **Exponential Integrals and Related Functions**, including exponential integrals, logarithmic integrals and integrals of trigonometric and hyperbolic functions.
- **Gamma Functions and Related Functions**, including gamma functions, psi functions, Pochhammer's function and Beta functions.
- **Error Functions and Related Functions**, including error functions and Fresnel integrals.
- **Bessel Functions**, including real order complex valued Bessel functions.
- **Kelvin Functions**, including Kelvin functions and their derivatives.
- **Airy Functions**, including Airy functions and their derivatives.

- **Elliptic Integrals**, including complete and incomplete elliptic integrals.
- **Elliptic and Related Functions**, including Weierstrass P-functions and the Jacobi elliptic function**.**
- **Probability Distribution Functions and Inverses**, including statistical functions, such as chi-squared and inverse beta and many others.
- **Mathieu Functions**, including eigenvalues and sequence of Mathieu functions.

STATISTICAL FUNCTIONALITY

The statistical functionality is organized into 20 sections. These capabilities range from analysis of variance to random number generation.

- **Basic Statistics**, including univariate summary statistics, nonparametric tests, such as sign and Wilcoxon rank sum, and goodness-of-fit tests, such as chi-squared and Shapiro-Wilk.
- **Regression**, including stepwise regression, all best regression, multiple linear regression models, polynomial models and nonlinear models.
- **Correlation**, including sample variance-covariance, partial correlation and covariances, pooled variance-covariance and robust estimates of a covariance matrix and mean factor.
- **Analysis of Variance**, including one-way classification models, a balanced factorial design with fixed effects and the Student-Newman-Keuls multiple comparisons test.
- **Categorical and Discrete Data Analysis**, including chi-squared analysis of a two-way contingency table, exact probabilities in a two-way contingency table and analysis of categorical data using general linear models.
- **Nonparametric Statistics**, including sign tests, Wilcoxon sum tests and Cochran Q test for related observations.
- **Tests of Goodness-of-Fit and Randomness**, including chi-squared goodness-of-fit tests, Kolmogorov/Smirnov tests and tests for normality.
- **Time Series Analysis and Forecasting**, including analysis and forecasting of time series using a nonseasonal ARMA model, GARCH (Generalized Autoregressive Conditional Heteroskedasticity), Kalman filtering, Automatic Model Selection, Bayesian Seasonal Analysis and Prediction, Optimum Controller Design, Spectral Density Estimation, portmanteau lack of fit test and difference of a seasonal or nonseasonal time series.
- **Covariance Structures and Factor Analysis**, including principal components and factor analysis.
- **Discriminant Analysis**, including analysis of data using a generalized linear model and using various parametric models.
- **Cluster Analysis**, including hierarchical cluster analysis and k-means cluster analysis.
- **Sampling**, including analysis of data using a simple or stratified random sample.
- **Survival Analysis, Life Testing and Reliability**, including Kaplan-Meier estimates of survival probabilities.
- **Multidimensional Scaling**, including alternating least squares methods.
- **Density and Hazard Estimation**, including estimates for density and modified likelihood for hazards.
- **Line Printer Graphics**, including histograms, scatter plots, exploratory data analysis, empirical probability distribution, and other graphics routines.

- **Probability Distribution Functions and Inverses**, including binomial, hypergeometric, bivariate normal, gamma and many more.
- **Random Number Generation**, including a generator for multivariate normal distributions and pseudorandom numbers from several distributions, including gamma, Poisson and beta., and low discrepancy sequence.
- **Utilities**, including CPU time used, error handling and machine, mathematical, physical constants, retrieval of machine constants and changing error-handling.
- **Mathematical Support**, including linear systems, special functions, and nearest neighbors.

IMSL – ALSO AVAILABLE IN C AND JAVA

IMSL C NUMERICAL LIBRARY

The IMSL C Numerical Library ("CNL") is a comprehensive set of pre-built, thread-safe mathematical and statistical analysis functions that C or C++ programmers can embed directly into their numerical analysis applications. CNL's functions are based upon the same algorithms contained in the company's highly regarded IMSL Fortran Library. Visual Numerics, Inc. has been providing algorithms for mathematical and statistical computations under the IMSL name since 1970.

CNL significantly shortens program development time by taking full advantage of the intrinsic characteristics and desirable features of the C language. Variable argument lists simplify calling sequences. The concise set of required arguments contains only the information necessary for usage. Optional arguments provide added functionality and power to each function. You'll find that using CNL saves significant effort in your source code development and thousands of dollars in the design, development, testing and maintenance of your application.

JMSL: A NUMERICAL LIBRARY FOR JAVA

JMSL is a 100% Pure Java numerical library for the Java environment. The library extends core Java numerics and allows developers to seamlessly integrate advanced mathematical, statistical, financial, and charting functions into their Java applications.

JMSL is an object-oriented implementation of several important classes of mathematical and statistical functions drawn from the IMSL algorithm repository. Visual Numerics has taken individual algorithms and reimplemented them as object-oriented, Java methods. JMSL also adds financial functions and charting to the library, taking advantage of the collaboration and graphical benefits of Java. JMSL is designed with extensibility in mind; new classes may be derived from existing ones to add functionality to satisfy particular requirements.

Because JMSL is a 100% Pure Java class library, it can be deployed on any platform that supports Java. JMSL can be used to write client-side applets, server-side applications or even non-networked desktop applications. JMSL applets perform all processing on the Java client, whether it is a thin client, such as a network computer, a PC or workstation equipped with a Java Virtual Machine. Client-side processing reduces the number of "round trips" to a networked server, which in turn minimizes network traffic and system latency.

Chapter 1: Linear Systems

LINEAR SOLVERS

LIN_SOL_GEN

Solves a real general system of linear equations $Ax = b$.

LIN_SOL_SELF

Solves a system of linear equations $Ax = b$, where A is a self-adjoint matrix.

LIN_SOL_LSQ

Solves a rectangular system of linear equations $Ax \cong b$, in a least-squares sense.

LIN_SOL_SVD

Solves a rectangular least-squares system of linear equations $Ax \cong b$ using singular value decomposition.

LIN_SOL_TRI

Solves multiple systems of linear equations.

LIN_SVD

Computes the singular value decomposition (SVD) of a rectangular matrix, A.

LARGE-SCALE PARALLEL SOLVERS

PARALLEL_NONNEGATIVE_LSQ

Solves a linear, non-negative constrained least-squares system.

PARALLEL_BOUNDED_LSQ

Solves a linear least-squares system with bounds on the unknowns.

SOLUTION OF LINEAR SYSTEMS, MATRIX INVERSION, AND DETERMINANT EVALUATION

REAL GENERAL MATRICES

LSARG

Solves a real general system of linear equations with iterative refinement.

LSLRG

Solves a real general system of linear equations without iterative refinement.

LFCRG

Computes the LU factorization of a real general matrix and estimates its L_1 condition number.

LFTRG

Computes the LU factorization of a real general matrix.

LFSRG

Solves a real general system of linear equations given the LU factorization of the coefficient matrix.

LFIRG

Uses iterative refinement to improve the solution of a real general system of linear equations.

LFDRG

Computes the determinant of a real general matrix given the LU factorization of the matrix.

LINRG

Computes the inverse of a real general matrix.

COMPLEX GENERAL MATRICES

LSACG
Solves a complex general system of linear equations with iterative refinement.

LSLCG
Solves a complex general system of linear equations without iterative refinement.

LFCCG
Computes the *LU* factorization of a complex general matrix and estimates its L_1 condition number.

LFTCG
Computes the *LU* factorization of a complex general matrix.

LFSCG
Solves a complex general system of linear equations given the *LU* factorization of the coefficient matrix.

LFICG
Uses iterative refinement to improve the solution of a complex general system of linear equations.

LFDCG
Computes the determinant of a complex general matrix given the *LU* factorization of the matrix.

LINCG
Computes the inverse of a complex general matrix.

REAL TRIANGULAR MATRICES

LSLRT
Solves a real triangular system of linear equations.

LFCRT
Estimates the condition number of a real triangular matrix.

LFDRT
Computes the determinant of a real triangular matrix.

LINRT
Computes the inverse of a real triangular matrix.

COMPLEX TRIANGULAR MATRICES

LSLCT
Solves a complex triangular system of linear equations.

LFCCT
Estimates the condition number of a complex triangular matrix.

LFDCT
Computes the determinant of a complex triangular matrix.

LINCT
Computes the inverse of a complex triangular matrix.

REAL POSITIVE DEFINITE MATRICES

LSADS
Solves a real symmetric positive definite system of linear equations with iterative refinement.

LSLDS
Solves a real symmetric positive definite system of linear equations without iterative refinement.

LFCDS

Computes the $R^T R$ Cholesky factorization of a real symmetric positive definite matrix and estimates its L_1 condition number.

LFTDS

Computes the $R^T R$ Cholesky factorization of a real symmetric positive definite matrix.

LFSDS

Solves a real symmetric positive definite system of linear equations given the $R^T R$ Cholesky factorization of the coefficient matrix.

LFIDS

Uses iterative refinement to improve the solution of a real symmetric positive definite system of linear equations.

LFDDS

Computes the determinant of a real symmetric positive definite matrix given the $R^T R$ Cholesky factorization of the matrix.

LINDS

Computes the inverse of a real symmetric positive definite matrix.

REAL SYMMETRIC MATRICES

LSASF

Solves a real symmetric system of linear equations with iterative refinement.

LSLSF

Solves a real symmetric system of linear equations without iterative refinement.

LFCSF

Computes the $U D U^T$ factorization of a real symmetric matrix and estimates its L_1 condition number.

LFTSF

Computes the $U D U^T$ factorization of a real symmetric matrix.

LFSSF

Solves a real symmetric system of linear equations given the $U D U^T$ factorization of the coefficient matrix.

LFISF

Uses iterative refinement to improve the solution of a real symmetric system of linear equations.

LFDSF

Computes the determinant of a real symmetric matrix given the $U D U^T$ factorization of the matrix.

COMPLEX HERMITIAN POSITIVE DEFINITE MATRICES

LSADH

Solves a Hermitian positive definite system of linear equations with iterative refinement.

LSLDH

Solves a complex Hermitian positive definite system of linear equations without iterative refinement.

LFCDH

Computes the $R^H R$ factorization of a complex Hermitian positive definite matrix and estimates its L_1 condition number.

LFTDH

Computes the $R^H R$ factorization of a complex Hermitian positive definite matrix.

LFSDH

Solves a complex Hermitian positive definite system of linear equations given the $R^H R$ factorization of the coefficient matrix.

LFIDH

Uses iterative refinement to improve the solution of a complex Hermitian positive definite system of linear equations.

LFDDH

Computes the determinant of a complex Hermitian positive definite matrix given the $R^T R$ Cholesky factorization of the matrix.

COMPLEX HERMITIAN MATRICES

LSAHF

Solves a complex Hermitian system of linear equations with iterative refinement.

LSLHF

Solves a complex Hermitian system of linear equations without iterative refinement.

LFCHF

Computes the $U DU^H$ factorization of a complex Hermitian matrix and estimates its L_1 condition number.

LFTHF

Computes the $U DU^H$ factorization of a complex Hermitian matrix.

LFSHF

Solves a complex Hermitian system of linear equations given the $U DU^H$ factorization of the coefficient matrix.

LFIHF

Uses iterative refinement to improve the solution of a complex Hermitian system of linear equations.

LFDHF

Computes the determinant of a complex Hermitian matrix given the $U DU^H$ factorization of the matrix.

REAL BAND MATRICES IN BAND STORAGE MODE

LSLTR

Solves a real tridiagonal system of linear equations.

LSLCR

Computes the $L DU$ factorization of a real tridiagonal matrix A using a cyclic reduction algorithm.

LSARB

Solves a real system of linear equations in band storage mode with iterative refinement.

LSLRB

Solves a real system of linear equations in band storage mode without iterative refinement.

LFCRB

Computes the LU factorization of a real matrix in band storage mode and estimates its L_1 condition number.

LFTRB

Computes the LU factorization of a real matrix in band storage mode.

LFSRB

Solves a real system of linear equations given the LU factorization of the coefficient matrix in band storage mode.

LFIRB

Uses iterative refinement to improve the solution of a real system of linear equations in band storage mode.

LFDRB

Computes the determinant of a real matrix in band storage mode given the *LU* factorization of the matrix.

REAL BAND SYMMETRIC POSITIVE DEFINITE MATRICES IN BAND STORAGE MODE

LSAQS

Solves a real symmetric positive definite system of linear equations in band symmetric storage mode with iterative refinement.

LSLQS

Solves a real symmetric positive definite system of linear equations in band symmetric storage mode without iterative refinement.

LSLPB

Computes the R^TDR Cholesky factorization of a real symmetric positive definite matrix A in codiagonal band symmetric storage mode. Solves a system $Ax = b$.

LFCQS

Computes the $R^T R$ Cholesky factorization of a real symmetric positive definite matrix in band symmetric storage mode and estimates its L_1 condition number.

LFTQS

Computes the R^TR Cholesky factorization of a real symmetric positive definite matrix in band symmetric storage mode.

LFSQS

Solves a real symmetric positive definite system of linear equations given the factorization of the coefficient matrix in band symmetric storage mode.

LFIQS

Uses iterative refinement to improve the solution of a real symmetric positive definite system of linear equations in band symmetric storage mode.

LFDQS

Computes the determinant of a real symmetric positive definite matrix given the R^TR Cholesky factorization of the band symmetric storage mode.

COMPLEX BAND MATRICES IN BAND STORAGE MODE

LSLTQ

Solves a complex tridiagonal system of linear equations.

LSLCQ

Computes the *LDU* factorization of a complex tridiagonal matrix A using a cyclic reduction algorithm.

LSACB

Solves a complex system of linear equations in band storage mode with iterative refinement.

LSLCB

Solves a complex system of linear equations in band storage mode without iterative refinement.

LFCCB

Computes the *LU* factorization of a complex matrix in band storage mode and estimates its L_1 condition number.

LFTCB

Computes the LU factorization of a complex matrix in band storage mode.

LFSCB

Solves a complex system of linear equations given the LU factorization of the coefficient matrix in band storage mode.

LFICB

Uses iterative refinement to improve the solution of a complex system of linear equations in band storage mode.

LFDCB

Computes the determinant of a complex matrix given the LU factorization of the matrix in band storage mode.

COMPLEX BAND POSITIVE DEFINITE MATRICES IN BAND STORAGE MODE

LSAQH

Solves a complex Hermitian positive definite system of linear equations in band Hermitian storage mode with iterative refinement.

LSLQH

Solves a complex Hermitian positive definite system of linear equations in band Hermitian storage mode without iterative refinement.

LSLQB

Computes the $R^H DR$ Cholesky factorization of a complex Hermitian positive-definite matrix A in codiagonal band Hermitian storage mode. Solves a system $Ax = b$.

LFCQH

Computes the $R^H R$ factorization of a complex Hermitian positive definite matrix in band Hermitian storage mode and estimates its L_1 condition number.

LFTQH

Computes the $R^H R$ factorization of a complex Hermitian positive definite matrix in band Hermitian storage mode.

LFSQH

Solves a complex Hermitian positive definite system of linear equations given the factorization of the coefficient matrix in band Hermitian storage mode.

LFIQH

Uses iterative refinement to improve the solution of a complex Hermitian positive definite system of linear equations in band Hermitian storage mode.

LFDQH

Computes the determinant of a complex Hermitian positive definite matrix given the $R^T R$ Cholesky factorization in band Hermitian storage mode.

REAL SPARSE LINEAR EQUATION SOLVERS

LSLXG

Solves a sparse system of linear algebraic equations by Gaussian elimination.

LFTXG

Computes the LU factorization of a real general sparse matrix.

LFSXG

Solves a sparse system of linear equations given the LU factorization of the coefficient matrix.

COMPLEX SPARSE LINEAR EQUATION SOLVERS

LSLZG

Solves a complex sparse system of linear equations by Gaussian elimination.

LFTZG

Computes the *LU* factorization of a complex general sparse matrix.

LFSZG

Solves a complex sparse system of linear equations given the *LU* factorization of the coefficient matrix.

REAL SPARSE SYMMETRIC POSITIVE DEFINITE LINEAR EQUATIONS SOLVERS

LSLXD

Solves a sparse system of symmetric positive definite linear algebraic equations by Gaussian elimination.

LSCXD

Performs the symbolic Cholesky factorization for a sparse symmetric matrix using a minimum degree ordering or a user-specified ordering, and set up the data structure for the numerical Cholesky factorization.

LNFXD

Computes the numerical Cholesky factorization of a sparse symmetrical matrix A.

LFSXD

Solves a real sparse symmetric positive definite system of linear equations, given the Cholesky factorization of the coefficient matrix.

COMPLEX SPARSE HERMITIAN POSITIVE DEFINITE LINEAR EQUATIONS SOLVERS

LSLZD

Solves a complex sparse Hermitian positive definite system of linear equations by Gaussian elimination.

LNFZD

Computes the numerical Cholesky factorization of a sparse Hermitian matrix A.

LFSZD

Solves a complex sparse Hermitian positive definite system of linear equations, given the Cholesky factorization of the coefficient matrix.

REAL TOEPLITZ MATRICES IN TOEPLITZ STORAGE MODE

LSLTO

Solves a real Toeplitz linear system.

COMPLEX TOEPLITZ MATRICES IN TOEPLITZ STORAGE MODE

LSLTC

Solves a complex Toeplitz linear system.

COMPLEX CICRCULAR MATRICES IN CIRCULANT STORAGE MODE

LSLCC

Solves a complex circulant linear system.

ITERATIVE METHODS

PCGRC

Solves a real symmetric definite linear system using a preconditioned conjugate gradient method with reverse communication.

JCGRC

Solves a real symmetric definite linear system using the Jacobi-preconditioned conjugate gradient method with reverse communication.

GMRES

Uses `GMRES` with reverse communication to generate an approximate solution of $Ax = b$.

LINEAR LEAST SQUARES AND MATRIX FACTORIZATION

LEAST SQUARES, QR DECOMPOSITION AND GENERALIZED INVERSE LEAST SQUARES

LSQRR

Solves a linear least-squares problem without iterative refinement.

LQRRV

Computes the least-squares solution using Householder transformations applied in blocked form.

LSBRR

Solves a linear least-squares problem with iterative refinement.

LCLSQ

Solves a linear least-squares problem with linear constraints.

LQRRR

Computes the QR decomposition, $AP = QR$, using Householder transformations.

LQERR

Accumulate the orthogonal matrix Q from its factored form given the QR factorization of a rectangular matrix A.

LQRSL

Computes the coordinate transformation, projection, and complete the solution of the least-squares problem $Ax = b$.

LUPQR

Computes an updated QR factorization after the rank-one matrix αxy^T is added.

CHOLESKY FACTORIZATION

LCHRG

Computes the Cholesky decomposition of a symmetric positive semidefinite matrix with optional column pivoting.

LUPCH

Updates the $R^T R$ Cholesky factorization of a real symmetric positive definite matrix after a rank-one matrix is added.

LDNCH

Downdates the $R^T R$ Cholesky factorization of a real symmetric positive definite matrix after a rank-one matrix is removed.

SINGULAR VALUE DECOMPOSITIONS

LSVRR

Computes the singular value decomposition of a real matrix.

LSVCR

Computes the singular value decomposition of a complex matrix.

LSGRR

Computes the generalized inverse of a real matrix.

Chapter 2: Eigensystem Analysis

EIGENVALUE DECOMPOSITION

LIN_EIG_SELF

Computes the eigenvalues of a self-adjoint matrix, A.

LIN_EIG_GEN

Computes the eigenvalues of an $n \times n$ matrix, A.

LIN_GEIG_GEN

Computes the generalized eigenvalues of an $n \times n$ matrix pencil, $Av = \lambda Bv$.

EIGENVALUES AND (OPTIONALLY) EIGENVECTORS OF $AX = \lambda X$

REAL GENERAL PROBLEM $AX = \lambda X$

EVLRG

Computes all of the eigenvalues of a real matrix.

EVCRG

Computes all of the eigenvalues and eigenvectors of a real matrix.

EPIRG

Computes the performance index for a real eigensystem.

COMPLEX GENERAL PROBLEM $AX = \lambda X$

EVLCG

Computes all of the eigenvalues of a complex matrix.

EVCCG

Computes all of the eigenvalues and eigenvectors of a complex matrix.

EPICG

Computes the performance index for a complex eigensystem.

REAL SYMMETRIC PROBLEM $AX = \lambda X$

EVLSF

Computes all of the eigenvalues of a real symmetric matrix.

EVCSF

Computes all of the eigenvalues and eigenvectors of a real symmetric matrix.

EVASF

Computes the largest or smallest eigenvalues of a real symmetric matrix.

EVESF

Computes the largest or smallest eigenvalues and the corresponding eigenvectors of a real symmetric matrix.

EVBSF

Computes selected eigenvalues of a real symmetric matrix.

EVFSF

Computes selected eigenvalues and eigenvectors of a real symmetric matrix.

EPISF

Computes the performance index for a real symmetric eigensystem.

REAL BAND SYMMETRIC MATRICES IN BAND STROAGE MODE

EVLSB

Computes all of the eigenvalues of a real symmetric matrix in band symmetric storage mode.

EVCSB

Computes all of the eigenvalues and eigenvectors of a real symmetric matrix in band symmetric storage mode.

EVASB

Computes the largest or smallest eigenvalues of a real symmetric matrix in band symmetric storage mode.

EVESB

Computes the largest or smallest eigenvalues and the corresponding eigenvectors of a real symmetric matrix in band symmetric storage mode.

EVBSB

Computes the eigenvalues in a given interval of a real symmetric matrix stored in band symmetric storage mode.

EVFSB

Computes the eigenvalues in a given interval and the corresponding eigenvectors of a real symmetric matrix stored in band symmetric storage mode.

EPISB

Computes the performance index for a real symmetric eigensystem in band symmetric storage mode.

COMPLEX HERMITIAN MATRICES

EVLHF

Computes all of the eigenvalues of a complex Hermitian matrix.

EVCHF

Computes all of the eigenvalues and eigenvectors of a complex Hermitian matrix.

EVAHF

Computes the largest or smallest eigenvalues of a complex Hermitian matrix.

EVEHF

Computes the largest or smallest eigenvalues and the corresponding eigenvectors of a complex Hermitian matrix.

EVBHF

Computes the eigenvalues in a given range of a complex Hermitian matrix.

EVFHF

Computes the eigenvalues in a given range and the corresponding eigenvectors of a complex Hermitian matrix.

EPIHF

Computes the performance index for a complex Hermitian eigensystem.

REAL UPPER HESSENBERG MATRICES

EVLRH

Computes all of the eigenvalues of a real upper Hessenberg matrix.

EVCRH

Computes all of the eigenvalues and eigenvectors of a real upper Hessenberg matrix.

COMPLEX UPPER HESSENBERG MATRICES

EVLCH

Computes all of the eigenvalues of a complex upper Hessenberg matrix.

EVCCH

Computes all of the eigenvalues and eigenvectors of a complex upper Hessenberg matrix.

EIGENVALUES AND (OPTIONALLY) EIGENVECTORS OF $AX = \lambda BX$

REAL GENERAL PROBLEM $AX = \lambda X$

GVLRG
Computes all of the eigenvalues of a generalized real eigensystem $Az = \lambda Bz$.

GVCRG
Computes all of the eigenvalues and eigenvectors of a generalized real eigensystem $Az = \lambda Bz$.

GPIRG
Computes the performance index for a generalized real eigensystem $Az = \lambda Bz$.

COMPLEX GENERAL PROBLEM $AX = \lambda X$

GVLCG
Computes all of the eigenvalues of a generalized complex eigensystem $Az = \lambda Bz$.

GVCCG
Computes all of the eigenvalues and eigenvectors of a generalized complex eigensystem $Az = \lambda Bz$.

GPICG
Computes the performance index for a generalized complex eigensystem $Az = \lambda Bz$.

REAL SYMMETRIC PROBLEM $AX = \lambda X$

GVLSP
Computes all of the eigenvalues of the generalized real symmetric eigenvalue problem $Az = \lambda Bz$, with B symmetric positive definite.

GVCSP
Computes all of the eigenvalues and eigenvectors of the generalized real symmetric eigenvalue problem $Az = \lambda Bz$, with B symmetric positive definite.

GPISP
Computes the performance index for a generalized real symmetric eigensystem problem.

Chapter 3: Interpolation and Approximation

CURVE AND SURFACE FITTING WITH SPLINES

SPLINE_CONSTRAINTS
Returns the derived type array result.

SPLINE_VALUES
Returns an array result, given an array of input.

SPLINE_FITTING
Weighted least-squares fitting by B-splines to discrete One-Dimensional data is performed.

SURFACE_CONSTRAINTS
Returns the derived type array result given optional input.

SURFACE_VALUES
Returns a tensor product array result, given two arrays of independent variable values

SURFACE_FITTING
Weighted least-squares fitting by tensor product B-splines to discrete two-dimensional data is performed.

CUBIC SPLINE INTERPOLATION

CSIEZ

Computes the cubic spline interpolant with the ‘not-a-knot’ condition and returns values of the interpolant at specified points.

CSINT

Computes the cubic spline interpolant with the ‘not-a-knot’ condition.

CSDEC

Computes the cubic spline interpolant with specified derivative endpoint conditions.

CSHER

Computes the Hermite cubic spline interpolant.

CSAKM

Computes the Akima cubic spline interpolant.

CSCON

Computes a cubic spline interpolant that is consistent with the concavity of the data.

CSPER

Computes the cubic spline interpolant with periodic boundary conditions.

CUBIC SPLINE EVALUATION AND INTEGRATION

CSVAL

Evaluates a cubic spline.

CSDER

Evaluates the derivative of a cubic spline.

CS1GD

Evaluates the derivative of a cubic spline on a grid.

CSITG

Evaluates the integral of a cubic spline.

B-SPLINE INTERPOLATION

SPLEZ

Computes the values of a spline that either interpolates or fits user-supplied data.

BSINT

Computes the spline interpolant, returning the B-spline coefficients.

BSNAK

Computes the “not-a-knot” spline knot sequence.

BSOPK

Computes the “optimal” spline knot sequence.

BS2IN

Computes a two-dimensional tensor-product spline interpolant, returning the tensor-product B-spline coefficients.

BS3IN

Computes a three-dimensional tensor-product spline interpolant, returning the tensor-product B-spline coefficients.

SPLINE EVALUATION, INTEGRATION, AND CONVERSION TO PIECEWISE POLYNOMIAL GIVEN THE B-SPLINE REPRESENTATION

BSVAL

Evaluates a spline, given its B-spline representation.

BSDER

Evaluates the derivative of a spline, given its B-spline representation.

BS1GD

Evaluates the derivative of a spline on a grid, given its B-spline representation.

BSITG

Evaluates the integral of a spline, given its B-spline representation.

BS2VL

Evaluates a two-dimensional tensor-product spline, given its tensor-product B-spline representation.

BS2DR

Evaluates the derivative of a two-dimensional tensor-product spline, given its tensor-product B-spline representation.

BS2GD

Evaluates the derivative of a two-dimensional tensor-product spline, given its tensor-product B-spline representation on a grid.

BS2IG

Evaluates the integral of a tensor-product spline on a rectangular domain, given its tensor-product B-spline representation.

BS3VL

Evaluates a three-dimensional tensor-product spline, given its tensor-product B-spline representation.

BS3DR

Evaluates the derivative of a three-dimensional tensor-product spline, given its tensor-product B-spline representation.

BS3GD

Evaluates the derivative of a three-dimensional tensor-product spline, given its tensor-product B-spline representation on a grid.

BS3IG

Evaluates the integral of a tensor-product spline in three dimensions over a three-dimensional rectangle, given its tensor-product B-spline representation.

BSCPP

Converts a spline in B-spline representation to piecewise polynomial representation.

PIECEWISE POLYNOMIAL

PPVAL

Evaluates a piecewise polynomial.

PPDER

Evaluates the derivative of a piecewise polynominal.

PP1GD

Evaluates the derivative of a piecewise polynomial on a grid.

PPITG

Evaluates the integral of a piecewise polynomial.

QUADRATIC POLYNOMIAL INTERPOLATION ROUTINES FOR GRIDDED DATA

QDVAL

Evaluates a function defined on a set of points using quadratic interpolation.

QDDER

Evaluates the derivative of a function defined on a set of points using quadratic interpolation.

QD2VL

Evaluates a function defined on a rectangular grid using quadratic interpolation.

QD2DR

Evaluates the derivative of a function defined on a rectangular grid using quadratic interpolation.

QD3VL

Evaluates a function defined on a rectangular three-dimensional grid using quadratic interpolation.

QD3DR

Evaluates the derivative of a function defined on a rectangular three-dimensional grid using quadratic interpolation.

SCATTERED DATA INTERPOLATION

SURF

Computes a smooth bivariate interpolant to scattered data that is locally a quintic polynomial in two variables.

LEAST-SQUARES APPROXIMATION

RLINE

Fits a line to a set of data points using least squares.

RCURV

Fits a polynomial curve using least squares.

FNLSQ

Computes a least-squares approximation with user-supplied basis functions.

BSLSQ

Computes the least-squares spline approximation, and returns the B-spline coefficients.

BSVLS

Computes the variable knot B-spline least squares approximation to given data.

CONFT

Computes the least-squares constrained spline approximation, returning the B-spline coefficients.

BSLS2

Computes a two-dimensional tensor-product spline approximant using least squares, returning the tensor-product B-spline coefficients.

BSLS3

Computes a three-dimensional tensor-product spline approximant using least squares, returning the tensor-product B-spline coefficients.

CUBIC SPLINE SMOOTHING

CSSED

Smooth one-dimensional data by error detection.

CSSMH

Computes a smooth cubic spline approximation to noisy data.

CSSCV

Computes a smooth cubic spline approximation to noisy data using cross-validation to estimate the smoothing parameter.

RATIONAL L_∞ APPROXIMATION

RATCH

Computes a rational weighted Chebyshev approximation to a continuous function on an interval.

Chapter 4: Integration and Differentiation

UNIVARIATE QUADRATURE

QDAGS

Integrates a function (which may have endpoint singularities).

QDAG

Integrates a function using a globally adaptive scheme based on Gauss-Kronrod rules.

QDAGP

Integrates a function with singularity points given.

QDAGI

Integrates a function over an infinite or semi-infinite interval.

QDAWO

Integrates a function containing a sine or a cosine.

QDAWF

Computes a Fourier integral.

QDAWS

Integrates a function with algebraic-logarithmic singularities.

QDAWC

Integrates a function F(X)/(X – C) in the Cauchy principal value sense.

QDNG

Integrates a smooth function using a nonadaptive rule.

MULTIDIMENSIONAL QUADRATURE

TWODQ

Computes a two-dimensional iterated integral.

QAND

Integrates a function on a hyper-rectangle.

QMC

Integrates a function over a hyperrectangle using a quasi-Monte Carlo method.

GAUSS RULES AND THREE-TERM RECURRENCES

GQRUL

Computes a Gauss, Gauss-Radau, or Gauss-Lobatto quadrature rule with various classical weight functions.

GQRCF

Computes a Gauss, Gauss-Radau or Gauss-Lobatto quadrature rule given the recurrence coefficients for the monic polynomials orthogonal with respect to the weight function.

RECCF

Computes recurrence coefficients for various monic polynomials.

RECQR

Computes recurrence coefficients for monic polynomials given a quadrature rule.

FQRUL

Computes a Fejér quadrature rule with various classical weight functions.

DIFFERENTIATION

DERIV

Computes the first, second or third derivative of a user-supplied function.

Chapter 5: Differential Equations

FIRST-ORDER ORDINARY DIFFERENTIAL EQUATIONS

SOLUTION OF THE INITIAL VALUE PROBLEM FOR ODES

IVPRK

Solves an initial-value problem for ordinary differential equations using the Runge-Kutta-Verner fifth-order and sixth-order method.

IVMRK

Solves an initial-value problem $y' = f(t, y)$ for ordinary differential equations using Runge-Kutta pairs of various orders.

IVPAG

Solves an initial-value problem for ordinary differential equations using either Adams-Moulton's or Gear's BDF method.

SOLUTION OF THE BOUNDARY VALUE PROBLEM FOR ODES

BVPFD

Solves a (parameterized) system of differential equations with boundary conditions at two points, using a variable order, variable step size finite difference method with deferred corrections.

BVPMS

Solves a (parameterized) system of differential equations with boundary conditions at two points, using a multiple-shooting method.

SOLUTION OF DIFFERENTIAL-ALGEBRAIC SYSTEMS

DASPG

Solves a first order differential-algebraic system of equations, $g(t, y, y') = 0$, using the Petzold–Gear BDF method.

PARTIAL DIFFERENTIAL EQUATIONS

SOLUTION OF SYSTEMS OF PDES IN ONE DIMENSION

PDE_1D_MG

Method of lines with Variable Griddings.

MOLCH

Solves a system of partial differential equations of the form $u_t = f(x, t, u, u_x, u_{xx})$ using the method of lines.
The solution is represented with cubic Hermite polynomials.

SOLUTION OF A PDE IN TWO AND THREE DIMENSIONS

FPS2H

Solves Poisson's or Helmholtz's equation on a two-dimensional rectangle using a fast Poisson solver based on the `HODIE` finite-difference scheme on a uniform mesh.

FPS3H

Solves Poisson's or Helmholtz's equation on a three-dimensional box using a fast Poisson solver based on the `HODIE` finite-difference scheme on a uniform mesh.

STURM-LIOUVILLE PROBLEMS

SLEIG

Determines eigenvalues, eigenfunctions and/or spectral density functions for Sturm-Liouville problems in the form.

SLCNT

Calculates the indices of eigenvalues of a Sturm-Liouville problem of the form for

$$-\frac{d}{dx}(p(x)\frac{du}{dx})+q(x)u=\lambda r(x)u \text{ for } x \text{ in } [a,b]$$

with boundary conditions (at regular points)

$$a_1u-a_2(pu')=\lambda(a_1'u-a_2'(pu')) \text{ at } a$$
$$b_1u+b_2(pu')=0 \text{ at } b$$

in a specified subinterval of the real line, [α, β].

Chapter 6: Transforms

REAL TRIGONOMETRIC FFT

FAST_DFT

Computes the Discrete Fourier Transform of a rank-1 complex array, *x*.

FAST_2DFT

Computes the Discrete Fourier Transform (2DFT) of a rank-2 complex array, *x*.

FAST_3DFT

Computes the Discrete Fourier Transform (2DFT) of a rank-3 complex array, *x*.

FFTRF

Computes the Fourier coefficients of a real periodic sequence.

FFTRB

Computes the real periodic sequence from its Fourier coefficients.

FFTRI

Computes parameters needed by `FFTRF` and `FFTRB`.

COMPLEX EXPONENTIAL FFT

FFTCF

Computes the Fourier coefficients of a complex periodic sequence.

FFTCB

Computes the complex periodic sequence from its Fourier coefficients.

FFTCI

Computes parameters needed by `FFTCF` and `FFTCB`.

REAL SINE AND COSINE FFTS

FSINT

Computes the discrete Fourier sine transformation of an odd sequence.

FSINI

Computes parameters needed by `FSINT`.

FCOST

Computes the discrete Fourier cosine transformation of an even sequence.

FCOSI

Computes parameters needed by `FCOST`.

REAL QUARTER SINE AND QUARTER COSINE FFTS

QSINF

Computes the coefficients of the sine Fourier transform with only odd wave numbers.

QSINB

Computes a sequence from its sine Fourier coefficients with only odd wave numbers.

QSINI

Computes parameters needed by `QSINF` and `QSINB`.

QCOSF

Computes the coefficients of the cosine Fourier transform with only odd wave numbers.

QCOSB

Computes a sequence from its cosine Fourier coefficients with only odd wave numbers.

QCOSI

Computes parameters needed by `QCOSF` and `QCOSB`.

TWO- AND THREE- DIMENSIONAL COMPLEX FFTS

FFT2D

Computes Fourier coefficients of a complex periodic two-dimensional array.

FFT2B

Computes the inverse Fourier transform of a complex periodic two-dimensional array.

FFT3F

Computes Fourier coefficients of a complex periodic three-dimensional array.

FFT3B

Computes the inverse Fourier transform of a complex periodic three-dimensional array.

CONVOLUTIONS AND CORRELATIONS

RCONV

Computes the convolution of two real vectors.

CCONV

Computes the convolution of two complex vectors.

RCORL

Computes the correlation of two real vectors.

CCORL

Computes the correlation of two complex vectors.

LAPLACE TRANSFORM

INLAP

Computes the inverse Laplace transform of a complex function.

SINLP

Computes the inverse Laplace transform of a complex function.

Chapter 7: Nonlinear Equations

ZEROS OF A POLYNOMIAL

ZPLRC

Finds the zeros of a polynomial with real coefficients using Laguerre's method.

ZPORC

Finds the zeros of a polynomial with real coefficients using the Jenkins-Traub three-stage algorithm.

ZPOCC

Finds the zeros of a polynomial with complex coefficients using the Jenkins-Traub three-stage algorithm.

ZERO(S) OF A FUNCTION

ZANLY

Finds the zeros of a univariate complex function using Müller's method.

ZBREN
Finds a zero of a real function that changes sign in a given interval.

ZREAL
Finds the real zeros of a real function using Müller's method.

ROOT OF A SYSTEM OF EQUATIONS

NEQNF
Solves a system of nonlinear equations using a modified Powell hybrid algorithm and a finite-difference approximation to the Jacobian.

NEQNJ
Solves a system of nonlinear equations using a modified Powell hybrid algorithm with a user-supplied Jacobian.

NEQBF
Solves a system of nonlinear equations using factored secant update with a finite-difference approximation to the Jacobian.

NEQBJ
Solves a system of nonlinear equations using factored secant update with a user-supplied Jacobian.

Chapter 8: Optimization

UNCONSTRAINED MINIMIZATION

UNIVARIATE FUNCTION

UVMIF
Finds the minimum point of a smooth function of a single variable using only function evaluations.

UVMID
Finds the minimum point of a smooth function of a single variable using both function evaluations and first derivative evaluations.

UVMGS
Finds the minimum point of a nonsmooth function of a single variable.

MULTIVARIATE FUNCTION

UMINF
Minimizes a function of N variables using a quasi-Newton method and a finite-difference gradient.

UMING
Minimizes a function of N variables using a quasi-Newton method and a user-supplied gradient.

UMIDH
Minimizes a function of N variables using a modified Newton method and a finite-difference Hessian.

UMIAH
Minimizes a function of N variables using a modified Newton method and a user-supplied Hessian.

UMCGF
Minimizes a function of N variables using a conjugate gradient algorithm and a finite-difference gradient.

UMCGG
Minimizes a function of N variables using a conjugate gradient algorithm and a user-supplied gradient.

UMPOL
Minimizes a function of N variables using a direct search polytope algorithm.

NONLINEAR LEAST SQUARES

UNLSF

Solves a nonlinear least-squares problem using a modified Levenberg-Marquardt algorithm and a finite-difference Jacobian.

UNLSJ

Solves a nonlinear least squares problem using a modified Levenberg-Marquardt algorithm and a user-supplied Jacobian.

MINIMIZATION WITH SIMPLE BOUNDS

BCONF

Minimizes a function of N variables subject to bounds on the variables using a quasi-Newton method and a finite-difference gradient.

BCONG

Minimizes a function of N variables subject to bounds on the variables using a quasi-Newton method and a user-supplied gradient.

BCODH

Minimizes a function of N variables subject to bounds on the variables using a modified Newton method and a finite-difference Hessian.

BCOAH

Minimizes a function of N variables subject to bounds on the variables using a modified Newton method and a user-supplied Hessian.

BCPOL

Minimizes a function of N variables subject to bounds on the variables using a direct search complex algorithm.

BCLSF

Solves a nonlinear least squares problem subject to bounds on the variables using a modified Levenberg-Marquardt algorithm and a finite-difference Jacobian.

BCLSJ

Solves a nonlinear least squares problem subject to bounds on the variables using a modified Levenberg-Marquardt algorithm and a user-supplied Jacobian.

BCNLS

Solves a nonlinear least-squares problem subject to bounds on the variables and general linear constraints.

LINEARLY CONSTRAINED MINIMIZATION

DLPRS

Solves a linear programming problem via the revised simplex algorithm.

SLPRS

Solves a sparse linear programming problem via the revised simplex algorithm.

QPROG

Solves a quadratic programming problem subject to linear equality/inequality constraints.

LCONF

Minimizes a general objective function subject to linear equality/inequality constraints.

LCONG

Minimizes a general objective function subject to linear equality/inequality constraints.

NONLINEARLY CONSTRAINED MINIMIZATION

NNLPF

Using a sequential equality constrained QP method.

NNLPG

Using a sequential equality constrained QP method and a user supplied gradient.

SERVICE ROUTINES

CDGRD

Approximates the gradient using central differences.

FDGRD

Approximates the gradient using forward differences.

FDHES

Approximates the Hessian using forward differences and function values.

GDHES

Approximates the Hessian using forward differences and a user-supplied gradient.

FDJAC

Approximate the Jacobian of M functions in N unknowns using forward differences.

CHGRD

Checks a user-supplied gradient of a function.

CHHES

Checks a user-supplied Hessian of an analytic function.

CHJAC

Checks a user-supplied Jacobian of a system of equations with M functions in N unknowns.

GGUES

Generates points in an N-dimensional space.

Chapter 9: Basic Matrix/Vector Operations

BASIC LINEAR ALGEBRA SUBPROGRAMS (BLAS)

SSET

Sets the components of a vector to a scalar.

SCOPY

Copies a vector x to a vector y, both single precision.

SSCAL

Multiplies a vector by a scalar, $y \leftarrow ay$, both single precision.

SVCAL

Multiplies a vector by a scalar and stores the result in another vector, $y \leftarrow ax$, all single precision.

SADD

Adds a scalar to each component of a vector, $x \leftarrow x + a$, all single precision.

SSUB

Subtract each component of a vector from a scalar, $x \leftarrow a - x$, all single precision.

SAXPY

Computes the scalar times a vector plus a vector, $y \leftarrow ax + y$, all single precision.

SSWAP

Interchange vectors x and y, both single precision.

SDOT

Computes the single-precision dot product $x^T y$.

DSDOT

Computes the single-precision dot product $x^T y$ using a double precision accumulator.

SDSDOT

Computes the sum of a single-precision scalar and a single precision dot product, $a + x^T y$, using a double-precision accumulator.

SDDOTI

Computes the sum of a single-precision scalar plus a single precision dot product using a double-precision accumulator, which is set to the result ACC $\leftarrow a + x^T y$.

SHPROD

Computes the Hadamard product of two single-precision vectors.

SXYZ

Computes a single-precision *xyz* product.

SSUM

Sums the values of a single-precision vector.

SASUM

Sums the absolute values of the components of a single-precision vector.

SNRM2

Computes the Euclidean length or L_2 norm of a single-precision vector.

SPRDCT

Multiplies the components of a single-precision vector.

ISMIN

Finds the smallest index of the component of a single-precision vector having minimum value.

ISMAX

Finds the smallest index of the component of a single-precision vector having maximum value.

ISAMIN

Finds the smallest index of the component of a single-precision vector having minimum absolute value.

ISAMAX

Finds the smallest index of the component of a single-precision vector having maximum absolute value.

SROTG

Constructs a Givens plane rotation in single precision.

SROT

Applies a Givens plane rotation in single precision.

SROTMG

Constructs a modified Givens plane rotation in single precision.

SROTM

Applies a modified Givens plane rotation in single precision.

SGEMV

Computes one of the matrix-vector operations: $y \leftarrow \alpha Ax + \beta y$, or $y \leftarrow \alpha A^T x + \beta y$.

SGBMV

Computes one of the matrix-vector operations:
$y \leftarrow \alpha Ax + \beta y$, or $y \leftarrow \alpha A^T x + \beta y$,
where A is a matrix stored in band storage mode.

CHEMV

Compute the matrix-vector operation $y \leftarrow \alpha Ax + \beta y$, where A is an Hermitian matrix.

CHBMV

Computes the matrix-vector operation $y \leftarrow \alpha Ax + \beta y$, where A is an Hermitian band matrix in band Hermitian storage.

SSYMV

Computes the matrix-vector operation $y \leftarrow \alpha Ax + \beta y$, where A is a symmetric matrix.

SSBMV

Computes the matrix-vector operation $y \leftarrow \alpha Ax + \beta y$, where A is a symmetric matrix in band symmetric storage mode.

STRMV

Computes one of the matrix-vector operations: $x \leftarrow Ax$ or $x \leftarrow A^T x$, where A is a triangular matrix.

STBMV

Computes one of the matrix-vector operations: $x \leftarrow Ax$ or $x \leftarrow A^T x$, where A is a triangular matrix in band storage mode.

STRSV

Solves one of the triangular linear systems: $x \leftarrow A^{-1}x$ or $x \leftarrow \left(A^{-1}\right)^T x$ where A is a triangular matrix.

STBSV

Solves one of the triangular systems: $x \leftarrow A^{-1}x$ or $x \leftarrow \left(A^{-1}\right)^T x$, where A is a triangular matrix in band storage mode.

SGER

Computes the rank-one update of a real general matrix: $A \leftarrow A + \alpha xy^T$

CGERU

Computes the rank-one update of a complex general matrix: $A \leftarrow A + \alpha xy^T$.

CGERC

Computes the rank-one update of a complex general matrix: $A \leftarrow A + \alpha x\overline{y}^T$.

CHER

Computes the rank-one update of an Hermitian matrix: $A \leftarrow A + \alpha x\overline{x}^T$ with x complex and α real.

CHER2

Computes a rank-two update of an Hermitian matrix: $A \leftarrow A + \alpha x\overline{y}^T + \overline{\alpha} y\overline{x}^T$.

SSYR

Computes the rank-one update of a real symmetric matrix: $A \leftarrow A + \alpha xx^T$.

SSYR2

Computes the rank-two update of a real symmetric matrix: $A \leftarrow A + \alpha xy^T + \alpha yx^T$.

SGEMM

Computes one of the matrix-matrix operations:

$$C \leftarrow \alpha AB + \beta C, C \leftarrow \alpha A^T B + \beta C, C \leftarrow \alpha AB^T + \beta C, \text{ or } C \leftarrow \alpha A^T B^T + \beta C.$$

SSYMM

Computes one of the matrix-matrix operations: $C \leftarrow \alpha AB + \beta C$ or $C \leftarrow \alpha BA + \beta C$, where A is a symmetric matrix and B and C are m by n matrices.

CHEMM

Computes one of the matrix-matrix operations: $C \leftarrow \alpha AB + \beta C$ or $C \leftarrow \alpha BA + \beta C$, where A is an Hermitian matrix and B and C are m by n matrices.

SSYRK

Computes one of the symmetric rank k operations:
$C \leftarrow \alpha AA^T + \beta C$ or $C \leftarrow \alpha A^T A + \beta C$,
where C is an n by n symmetric matrix and A is an n by k matrix in the first case and a k by n matrix in the second case.

CHERK

Computes one of the Hermitian rank k operations:
$C \leftarrow \alpha A\overline{A}^T + \beta C$ or $C \leftarrow \alpha \overline{A}^T A + \beta C$,
where C is an n by n Hermitian matrix and A is an n by k matrix in the first case and a k by n matrix in the second case.

SSYR2K

Computes one of the symmetric rank $2k$ operations: $C \leftarrow \alpha AB^T + \alpha BA^T + \beta C$ or $C \leftarrow \alpha A^T B + \alpha B^T A + \beta C$
where C is an n by n symmetric matrix and A and B are n by k matrices in the first case and k by n matrices in the second case.

CHER2K

Computes one of the Hermitian rank $2k$ operations:
$C \leftarrow \alpha A\overline{B}^T + \overline{\alpha} B\overline{A}^T + \beta C$ or $C \leftarrow \alpha \overline{A}^T B + \overline{\alpha} \overline{B}^T A + \beta C$, where C is an n by n Hermitian matrix and case and k by n matrices in the second case.

STRMM

Computes one of the matrix-matrix operations:
$B \leftarrow \alpha AB, B \leftarrow \alpha A^T B$ or $B \leftarrow \alpha BA, B \leftarrow \alpha BA^T$,
where B is an m by n matrix and A is a triangular matrix.

STRSM

Solves one of the matrix equations:
$B \leftarrow \alpha A^{-1} B, B \leftarrow \alpha BA^{-1}, B \leftarrow \alpha \left(A^{-1}\right)^T B$,
or $B \leftarrow \alpha B\left(A^{-1}\right)^T$
where B is an m by n matrix and A is a triangular matrix.

CTRSM

Solves one of the complex matrix equations:
$B \leftarrow \alpha A^{-1} B, B \leftarrow \alpha BA^{-1}, B \leftarrow \alpha \left(A^{-1}\right)^T B, B \leftarrow \alpha B\left(A^{-1}\right)^T$,
$B \leftarrow \alpha \left(\overline{A}^T\right)^{-1} B$, or $B \leftarrow \alpha B\left(\overline{A}^T\right)^{-1}$
where A is a traiangular matrix.

OTHER MATRIX/VECTOR OPERATIONS

MATRIX COPY

CRGRG

Copies a real general matrix.

CCGCG

Copies a complex general matrix.

CRBRB

Copies a real band matrix stored in band storage mode.

CCBCB

Copies a complex band matrix stored in complex band storage mode.

MATRIX CONVERSION

CRGRB

Converts a real general matrix to a matrix in band storage mode.

CRBRG

Converts a real matrix in band storage mode to a real general matrix.

CCGCB

Converts a complex general matrix to a matrix in complex band storage mode.

CCBCG

Converts a complex matrix in band storage mode to a complex matrix in full storage mode.

CRGCG

Copies a real general matrix to a complex general matrix.

CRRCR

Copies a real rectangular matrix to a complex rectangular matrix.

CRBCB

Converts a real matrix in band storage mode to a complex matrix in band storage mode.

CSFRG

Extends a real symmetric matrix defined in its upper triangle to its lower triangle.

CHFCG

Extends a complex Hermitian matrix defined in its upper triangle to its lower triangle.

CSBRB

Copies a real symmetric band matrix stored in band symmetric storage mode to a real band matrix stored in band storage mode.

CHBCB

Copies a complex Hermitian band matrix stored in band Hermitian storage mode to a complex band matrix stored in band storage mode.

TRNRR

Transposes a rectangular matrix.

MATRIX MULTIPLICATION

MXTXF

Computes the transpose product of a matrix, $A^T A$.

MXTYF

Multiplies the transpose of matrix A by matrix B, $A^T B$.

MXYTF

Multiplies a matrix A by the transpose of a matrix B, AB^T.

MRRRR

Multiplies two real rectangular matrices, AB.

MCRCR

Multiplies two complex rectangular matrices, AB.

HRRRR

Computes the Hadamard product of two real rectangular matrices.

BLINF

Computes the bilinear form $x^T Ay$.

POLRG

Evaluates a real general matrix polynomial.

MATRIX-VECTOR MULTIPLICATION

MURRV

Multiplies a real rectangular matrix by a vector.

MURBV

Multiplies a real band matrix in band storage mode by a real vector.

MUCRV

Multiplies a complex rectangular matrix by a complex vector.

MUCBV

Multiplies a complex band matrix in band storage mode by a complex vector.

MATRIX ADDITION

ARBRB

Adds two band matrices, both in band storage mode.

ACBCB

Adds two complex band matrices, both in band storage mode.

MATRIX NORM

NRIRR

Computes the infinity norm of a real matrix.

NR1RR

Computes the 1-norm of a real matrix.

NR2RR

Computes the Frobenius norm of a real rectangular matrix.

NR1RB

Computes the 1-norm of a real band matrix in band storage mode.

NR1CB

Computes the 1-norm of a complex band matrix in band storage mode.

DISTANCE BETWEEN TWO POINTS

DISL2

Computes the Euclidean (2-norm) distance between two points.

DISL1

Computes the 1-norm distance between two points.

DISLI

Computes the infinity norm distance between two points.

VECTOR CONVOLUTIONS

VCONR

Computes the convolution of two real vectors.

VCONC

Computes the convolution of two complex vectors.

EXTENDED PRECISION ARITHMETIC

DQINI

Initializes an extended-precision accumulator with a double-precision scalar.

DQSTO

Stores a double-precision approximation to an extended-precision scalar.

DQADD

Adds a double-precision scalar to the accumulator in extended precision.

DQMUL

Multiplies double-precision scalars in extended precision.

ZQINI

Initializes an extended-precision complex accumulator to a double complex scalar.

ZQSTO

Stores a double complex approximation to an extended-precision complex scalar.

ZQADD

Adds a double complex scalar to the accumulator in extended precision.

ZQMUL

Multiplies double complex scalars using extended precision.

Chapter 10: Linear Algebra Operators and Generic Functions

OPERATORS: .X., .TX., .XT., .HX., .XH

Computes matrix-vector and matrix-matrix products.

OPERATORS: .T., .H.

Computes transpose and conjugate transpose of a matrix.

OPERATOR: .I.

Computes the inverse matrix, for square non-singular matrices, or the Moore-Penrose generalized inverse matrix for singular square matrices or rectangular matrices.

OPERATORS: .IX., .XI.

Computes the inverse matrix times a vector or matrix for square non-singular matrices or the corresponding Moore-Penrose generalized inverse matrix for singular square matrices or rectangular matrices.

CHOL

Computes the Cholesky factorization of a positive-definite, symmetric or self-adjoint matrix, A.

COND

Computes the condition number of a rectangular matrix, A.

DET

Computes the determinant of a rectangular matrix, A.

DIAG

Constructs a square diagonal matrix from a rank-1 array or several diagonal matrices from a rank-2 array.

DIAGONALS

Extracts a rank-1 array whose values are the diagonal terms of a rank-2 array argument.

EIG

Computes the eigenvalue-eigenvector decomposition of an ordinary or generalized eigenvalue problem.

EYE

Creates a rank-2 square array whose diagonals are all the value one.

FFT

The Discrete Fourier Transform of a complex sequence and its inverse transform.

FFT_BOX

The Discrete Fourier Transform of several complex or real sequences.

IFFT

The inverse of the Discrete Fourier Transform of a complex sequence.

IFFT_BOX

The inverse Discrete Fourier Transform of several complex or real sequences.

ISNAN

This is a generic logical function used to test scalars or arrays for occurrence of an IEEE 754 Standard format of floating point (ANSI/IEEE 1985) NaN, or not-a-number.

NAN

Returns, as a scalar function, a value corresponding to the IEEE 754 Standard format of floating point (ANSI/IEEE 1985) for NaN.

NORM

Computes the norm of a rank-1 or rank-2 array.

ORTH
Orthogonalizes the columns of a rank-2 or rank-3 array.

RAND
Computes a scalar, rank-1, rank-2 or rank-3 array of random numbers.

RANK
Computes the mathematical rank of a rank-2 or rank-3 array.

SVD
Computes the singular value decomposition of a rank-2 or rank-3 array, $A = USV^T$.

UNIT
Normalizes the columns of a rank-2 or rank-3 array so each has Euclidean length of value one.

Chapter 11:Utilities

SCALAPACK UTILITIES

SCALAPACK_READ
Reads matrix data from a file and transmits it into the two-dimensional block-cyclic form.

SCALAPACK_WRITE
Writes the matrix data to a file.

PRINT

ERROR_POST
Prints error messages.

SHOW
Prints rank-1 or rank-2 arrays of numbers in a readable format.

WRRRN
Prints a real rectangular matrix with integer row and column labels.

WRRRL
Prints a real rectangular matrix with a given format and labels.

WRIRN
Prints an integer rectangular matrix with integer row and column labels

WRIRL
Prints an integer rectangular matrix with a given format and labels.

WRCRN
Prints a complex rectangular matrix with integer row and column labels.

WRCRL
Prints a complex rectangular matrix with a given format and labels.

WROPT
Sets or Retrieves an option for printing a matrix.

PGOPT
Sets or Retrieves page width and length for printing.

PERMUTE

PERMU
Rearranges the elements of an array as specified by a permutation.

PERMA
Permutes the rows or columns of a matrix.

SORT

SORT_REAL
Sorts a rank-1 array of real numbers x so the y results are algebraically nondecreasing, $y_1 \le y_2 \le \dots y_n$.

SVRGN
Sorts a real array by algebraically increasing value.

SVRGP
Sorts a real array by algebraically increasing value and returns the permutation that rearranges the array.

SVIGN
Sorts an integer array by algebraically increasing value.

SVIGP
Sorts an integer array by algebraically increasing value and returns the permutation that rearranges the array.

SVRBN
Sorts a real array by nondecreasing absolute value.

SVRBP
Sorts a real array by nondecreasing absolute value and returns the permutation that rearranges the array.

SVIBN
Sorts an integer array by nondecreasing absolute value.

SVIBP
Sorts an integer array by nondecreasing absolute value and returns the permutation that rearranges the array.

SEARCH

SRCH
Searches a sorted vector for a given scalar and returns its index.

ISRCH
Searches a sorted integer vector for a given integer and returns its index.

SSRCH
Searches a character vector, sorted in ascending ASCII order, for a given string and returns its index.

CHARACTER STRING MANIPULATION

ACHAR
Returns a character given its ASCII value.

IACHAR
Returns the integer ASCII value of a character argument.

ICASE
Returns the ASCII value of a character converted to uppercase.

IICSR
Compares two character strings using the ASCII collating sequence but without regard to case.

IIDEX
Determines the position in a string at which a given character sequence begins without regard to case.

CVTSI
Converts a character string containing an integer number into the corresponding integer form.

TIME, DATE AND VERSION

CPSEC
Returns CPU time used in seconds.

TIMDY
Gets time of day.

TDATE
Gets today’s date.

NDAYS
Computes the number of days from January 1, 1900, to the given date.

NDYIN
Gives the date corresponding to the number of days since January 1, 1900

IDYWK
Computes the day of the week for a given date.

VERML
Obtains IMSL MATH/LIBRARY-related version, system and serial numbers.

RANDOM NUMBER GENERATION

RAND_GEN
Generates a rank-1 array of random numbers.

RNGET
Retrieves the current value of the seed used in the IMSL random number generators.

RNSET
Initializes a random seed for use in the IMSL random number generators.

RNOPT
Selects the uniform (0, 1) multiplicative congruential pseudorandom number generator.

RNUNF
Generates a pseudorandom number from a uniform (0, 1) distribution.

RNUN
Generates pseudorandom numbers from a uniform (0, 1) distribution.

LOW DISCREPANCY SEQUENCES

FAURE_INIT
Generates pseudorandom numbers from a uniform (0, 1) distribution.

FAURE_FREE
Frees the structure containing information about the Faure sequence

FAURE_NEXT
Computes a shuffled Faure sequence.

OPTIONS MANAGER

IUMAG
This routine handles MATH/LIBRARY and STAT/LIBRARY type `INTEGER` options.

UMAG
Gets and puts type `REAL` options.

SUMAG
This routine handles MATH/LIBRARY and STAT/LIBRARY type `SINGLE PRECISION` options.

DUMAG
This routine handles MATH/LIBRARY and STAT/LIBRARY type `DOUBLE PRECISION` options.

LINE PRINTER GRAPHICS

PLOTP
Prints a plot of up to 10 sets of points.

MISCELLANEOUS

PRIME
Decomposes an integer into its prime factors.

CONST
Returns the value of various mathematical and physical constants.

CUNIT

Converts X in units XUNITS to Y in units YUNITS.

HYPOT

Computes $\sqrt{a^2 + b^2}$ without underflow or overflow.

IMSL MATH/LIBRARY SPECIAL FUNCTIONS

Chapter 1: Elementary Functions

CARG

Evaluates the argument of a complex number.

CBRT

Evaluates the cube root.

EXPRL

Evaluates the exponential function factored from first order, (`EXP`(`X`) – 1.0)/`X`.

LOG10

Extends FORTRAN's generic log10 function to evaluate the principal value of the complex common logarithm.

ALNREL

Evaluates the natural logarithm of one plus the argument.

Chapter 2: Trigonometric and Hyperbolic Functions

TRIGONOMETRIC FUNCTIONS

TAN

Extends FORTRAN's generic tan to evaluate the complex tangent.

COT

Evaluates the cotangent.

SINDG

Evaluates the sine for the argument in degrees.

COSDG

Evaluates the cosine for the argument in degrees.

ASIN

Extends FORTRAN's generic `ASIN` function to evaluate the complex arc sine.

ACOS

Extends FORTRAN's generic `ACOS` function evaluate the complex arc cosine.

ATAN

Extends FORTRAN's generic function `ATAN` to evaluate the complex arc tangent.

ATAN2

This function extends FORTRAN's generic function `ATAN2` to evaluate the complex arc tangent of a ratio.

HYPERBOLIC FUNCTIONS

SINH

Extends FORTRAN's generic function `SINH` to evaluate the complex hyperbolic sine.

COSH

Extends FORTRAN's generic function `COSH` to evaluate the complex hyperbolic cosine.

TANH

Extends FORTRAN's generic function `TANH` to evaluate the complex hyperbolic tangent.

INVERSE HYPERBOLIC FUNCTIONS

ASINH

Evaluates the arc hyperbolic sine.

ACOSH
Evaluates the arc hyperbolic cosine.

ATANH
Evaluates the arc hyperbolic tangent.

Chapter 3: Exponential Integrals and Related Functions

EI
Evaluates the exponential integral for arguments greater than zero and the Cauchy principal value for arguments less than zero.

E1
Evaluates the exponential integral for arguments greater than zero and the Cauchy principal value of the integral for arguments less than zero.

ENE
Evaluates the exponential integral of integer order for arguments greater than zero scaled by `EXP(X)`.

ALI
Evaluates the logarithmic integral.

SI
Evaluates the sine integral.

CI
Evaluates the cosine integral.

CIN
Evaluates a function closely related to the cosine integral.

SHI
Evaluates the hyperbolic sine integral..

CHI
Evaluates the hyperbolic cosine integral.

CINH
Evaluates a function closely related to the hyperbolic cosine integral.

Chapter 4: Gamma Function and Related Functions

FACTORIAL FUNCTION

FAC
Evaluates the factorial of the argument.

BINOM
Evaluates the binomial coefficient.

GAMMA FUNCTION

GAMMA
Evaluates the complete gamma function.

GAMR
Evaluates the reciprocal gamma function.

ALNGAM
Evaluates the logarithm of the absolute value of the gamma function.

ALGAMS
Returns the logarithm of the absolute value of the gamma function and the sign of gamma.

INCOMPLETE GAMMA FUNCTION

GAMI
Evaluates the incomplete gamma function.

GAMIC
Evaluates the complementary incomplete gamma function.

GAMIT
Evaluates the Tricomi form of the incomplete gamma function.

PSI FUNCTION

PSI

Evaluates the logarithmic derivative of the gamma function.

POCHHAMMER'S FUNCTION

POCH

Evaluates a generalization of Pochhammer's symbol.

POCH1

Evaluates a generalization of Pochhammer's symbol starting from the first order.

BETA FUNCTION

BETA

Evaluates the complete beta function.

ALBETA

Evaluates the natural logarithm of the complete beta function for positive arguments.

BETAI

Evaluates the incomplete beta function ratio.

Chapter 5: Error Function and Related Functions

ERROR FUNCTIONS

ERF

Evaluates the error function.

ERFC

Evaluates the complementary error function.

ERFCE

Evaluates the exponentially scaled complementary error function.

CERFE

Evaluates the complex scaled complemented error function.

ERFI

Evaluates the inverse error function.

ERFCI

Evaluates the inverse complementary error function.

DAWS

Evaluates Dawson's function.

FRESNEL INTEGRALS

FRESC

Evaluates the cosine Fresnel integral.

FRESS

Evaluates the sine Fresnel integral.

Chapter 6: Bessel Functions

BESSEL FUNCTIONS OF ORDERS 0 AND 1

BSJ0

Evaluates the Bessel function of the first kind of order zero.

BSJ1

Evaluates the Bessel function of the first kind of order one.

BSY0

Evaluates the Bessel function of the second kind of order zero.

BSY1

Evaluates the Bessel function of the second kind of order one.

BS10

Evaluates the modified Bessel function of the first kind of order zero.

BSI1
Evaluates the modified Bessel function of the first kind of order one.

BSKO
Evaluates the modified Bessel function of the third kind of order zero.

BSK1
Evaluates the modified Bessel function of the third kind of order one.

BSI0E
Evaluates the exponentially scaled modified Bessel function of the first kind of order zero.

BSI1E
Evaluates the exponentially scaled modified Bessel function of the first kind of order one.

BSK0E
Evaluates the exponentially scaled modified Bessel function of the third kind of order zero.

BSK1E
Evaluates the exponentially scaled modified Bessel function of the third kind of order one.

SERIES OF BESSEL FUNCTIONS, INTEGER ORDER

BSJNS
Evaluates a sequence of Bessel functions of the first kind with integer order and real arguments.

BSINS
Evaluates a sequence of modified Bessel functions of the first kind with integer order and real arguments.

SERIES OF BESSEL FUNCTIONS, REAL ORDER AND ARGUMENT

BSJS
Evaluates a sequence of Bessel functions of the first kind with real order and real positive arguments

BSYS
Evaluates a sequence of Bessel functions of the second kind with real nonnegative order and real positive arguments.

BSIS
Evaluates a sequence of modified Bessel functions of the first kind with real order and real positive arguments.

BSIES
Evaluates a sequence of exponentially scaled modified Bessel functions of the first kind with nonnegative real order and real positive arguments.

BSKS
Evaluates a sequence of modified Bessel functions of the third kind of fractional order.

BSKES
Evaluates a sequence of exponentially scaled modified Bessel functions of the third kind of fractional order.

SERIES OF BESSEL FUNCTIONS, REAL ORDER AND COMPLEX ARGUMENT

CBJS
Evaluates a sequence of Bessel functions of the first kind with real order and complex arguments.

CBYS
Evaluates a sequence of Bessel functions of the second kind with real order and complex arguments.

CBIS

Evaluates a sequence of modified Bessel functions of the first kind with real order and complex arguments.

CBKS

Evaluates a sequence of modified Bessel functions of the second kind with real order and complex arguments.

Chapter 7: Kelvin Functions

BER0

Evaluates the Kelvin function of the first kind, ber, of order zero.

BE10

Evaluates the Kelvin function of the first kind, bei, of order zero.

AKER0

Evaluates the Kelvin function of the second kind, ker, of order zero.

AKEI0

Evaluates the Kelvin function of the second kind, kei, of order zero.

BERP0

Evaluates the derivative of the Kelvin function of the first kind, ber, of order zero.

BEIP0

Evaluates the derivative of the Kelvin function of the first kind, bei, of order zero.

AKERP0

Evaluates the derivative of the Kelvin function of the second kind, ker, of order zero.

AKEIP0

Evaluates the Kelvin function of the second kind, kei, of order zero.

BER1

Evaluates the Kelvin function of the first kind, ber, of order one.

BEI1

Evaluates the Kelvin function of the first kind, bei, of order one.

AKER1

Evaluates the Kelvin function of the second kind, ker, of order one.

AKEI1

Evaluates the Kelvin function of the second kind, kei, of order one.

Chapter 8: Airy Functions

AI

Evaluates the Airy function.

BI

Evaluates the Airy function of the second kind.

AID

Evaluates the derivative of the Airy function.

BID

Evaluates the derivative of the Airy function of the second kind.

AIE

Evaluates the exponentially scaled Airy function.

BIE

Evaluates the exponentially scaled Airy function of the second kind.

AIDE

Evaluates the exponentially scaled derivative of the Airy function.

BIDE

Evaluates the exponentially scaled derivative of the Airy function of the second kind.

Chapter 9: Elliptic Integrals

ELK

Evaluates the complete elliptic integral of the kind K(x).

ELE

Evaluates the complete elliptic integral of the second kind $E(x)$.

ELRF

Evaluates Carlson's incomplete elliptic integral of the first kind R_F(X, Y, Z).

ELRD

Evaluates Carlson's incomplete elliptic integral of the second kind R_D(X, Y, Z).

ELRJ

Evaluates Carlson's incomplete elliptic integral of the third kind R_J(X, Y, Z, RHO).

ELRC

Evaluates an elementary integral from which inverse circular functions, logarithms and inverse hyperbolic functions can be computed.

Chapter 10: Elliptic and Related Functions

WEIERSTRASS ELLIPTIC AND RELATED FUNCTIONS

CWPL

Evaluates the Weierstrass' $\wp$ function in the lemniscatic case for complex argument with unit period parallelogram.

CWPLD

Evaluates the first derivative of the Weierstrass' $\wp$ function in the lemniscatic case for complex argument with unit period parallelogram.

CWPQ

Evaluates the Weierstrass' $\wp$ function in the equianharmonic case for complex argument with unit period parallelogram.

CWPQD

Evaluates the first derivative of the Weierstrass' $\wp$ function in the equianharmonic case for complex argument with unit period parallelogram.

JACOBI ELLIPTIC FUNCTIONS

EJSN

Evaluates the Jacobi elliptic function $\mathrm{sn}(x, m)$.

EJCN

Evaluates the Jacobi elliptic function $\mathrm{cn}(x, m)$.

EJDN

Evaluates the Jacobi elliptic function $\mathrm{dn}(x, m)$.

Chapter 11: Probability Distribution Functions and Inverses

DISCRETE RANDOM VARIABLES: DISTRIBUTION FUNCTIONS AND PROBABILITY FUNCTIONS

BINDF

Evaluates the binomial distribution function.

BINPR

Evaluates the binomial probability function.

HYPDF

Evaluates the hypergeometric distribution function.

HYPPR

Evaluates the hypergeometric probability function.

POIDF

Evaluates the Poisson distribution function.

POIPR

Evaluates the Poisson probability function.

CONTINUOUS RANDOM VARIABLES: DISTRIBUTION FUNCTIONS AND THEIR INVERSES

AKS1DF

Evaluates the distribution function of the one-sided Kolmogorov-Smirnov goodness of fit D^+ or D^- test statistic based on continuous data for one sample.

AKS2DF

Evaluates the distribution function of the Kolmogorov-Smirnov goodness of fit *D* test statistic based on continuous data for two samples.

ANORDF

Evaluates the standard normal (Gaussian) distribution function.

ANORIN

Evaluates the inverse of the standard normal (Gaussian) distribution function.

BETDF

Evaluates the beta probability distribution function.

BETIN

Evaluates the inverse of the beta distribution function.

BNRDF

Evaluates the bivariate normal distribution function.

CHIDF

Evaluates the chi-squared distribution function.

CHIIN

Evaluates the inverse of the chi-squared distribution function.

CSNDF

Evaluates the noncentral chi-squared distribution function.

FDF

Evaluates the *F* distribution function.

FIN

Evaluates the inverse of the *F* distribution function

GAMDF

Evaluates the gamma distribution function

TDF

Evaluates the Student's *t* distribution function.

TIN

Evaluates the inverse of the Student's *t* distribution function.

TNDF

Evaluates the noncentral Student's *t* distribution function.

GENERAL CONTINUOUS RANDOM VARIABLES

GCDF

Evaluates a general continuous cumulative distribution function given ordinates of the density.

GCIN

Evaluates the inverse of a general continuous cumulative distribution function given ordinates of the density.

Chapter 12: Mathieu Functions

MATEE

Evaluates the eigenvalues for the periodic Mathieu functions

MATCE

Evaluates a sequence of even, periodic, integer order, real Mathieu functions.

MATSE

Evaluates a sequence of odd, periodic, integer order, real Mathieu functions.

Chapter 13: Miscellaneous Functions

SPENC

Evaluates a form of Spence's integral.

INITS

Initializes the orthogonal series so the function value is the number of terms needed to insure the error is no larger than the requested accuracy.

CSEVL

Evaluates the *N*-term Chebyshev series.

Library Environments Utilities

The following routines are documented in the Reference Material sections of the IMSL® MATH/LIBRARY® and IMSL® STAT/LIBRARY® User's Manuals.

ERSET

Sets error handler default print and stop actions.

IERCD

Retrieves the code for an informational error.

N1RTY

Retrieves an error type for the most recently called IMSL routine.

IMACH

Retrieves interger machine constants.

AMACH

Retrieves machine constants.

DMACH

See AMACH.

IFNAN(X)

Checks if a floating-point number is NaN (not a number).

UMACH

Sets or Retrieves input or output device unit numbers.

IMSL STAT/LIBRARY

Chapter 1: Basic Statistics

FREQUENCY TABULATIONS

OWFRQ

Tallies observations into a one-way frequency table.

TWFRQ

Tallies observations into a two-way frequency table.

FREQ

Tallies multivariate observations into a multiway frequency table.

UNIVARIATE SUMMARY STATISTICS

UVSTA

Computes basic univariate statistics

RANKS AND ORDER STATISTICS

RANKS

Computes the ranks, normal scores, or exponential scores for a vector of observations.

LETTR

Produces a letter value summary.

ORDST

Determines order statistics.

EQTIL

Computes empirical quantiles.

PARAMETRIC ESTIMATES AND TESTS

TWOMV

Computes statistics for mean and variance inferences using samples from two normal populations.

BINES

Estimates the parameter *p* of the binomial distribution.

POIES

Estimates the parameter of the Poisson distribution.

NRCES

Computes maximum likelihood estimates of the mean and variance from grouped and/or censored normal data.

GROUPED DATA

GRPES

Computes basic statistics from grouped data.

CONTINOUS DATA IN A TABLE

CSTAT

Computes cell frequencies, cell means, and cell sums of squares for multivariate data.

MEDPL

Computes a median polish of a two-way table.

Chapter 2: Regression

SIMPLE LINEAR REGRESSION

RLINE

Fits a line to a set of data points using least squares.

RONE

Analyzes a simple linear regression model.

RINCF

Performs response control given a fitted simple linear regression model.

RINPF

Performs inverse prediction given a fitted simple linear regression model.

MULTIVARIATE GENERAL LINEAR MODEL ANALYSIS

MODEL FITTING

RLSE

Fits a multiple linear regression model using least squares.

RCOV

Fits a multivariate linear regression model given the variance-covariance matrix.

RGIVN

Fits a multivariate linear regression model via fast Givens transformations.

RGLM

Fits a multivariate general linear model.

RLEQU

Fits a multivariate linear regression model with linear equality restrictions $H\,B = G$ imposed on the regression parameters given results from routine `RGIVN` after `IDO` = 1 and `IDO` = 2 and prior to `IDO` = 3.

STATISTICAL INFERENCE AND DIAGNOSTICS

RSTAT

Computes statistics related to a regression fit given the coefficient estimates.

RCOVB

Computes the estimated variance-covariance matrix of the estimated regression coefficients given the *R* matrix.

CESTI

Constructs an equivalent completely testable multivariate general linear hypothesis $H\,BU = G$ from a partially testable hypothesis $H_pBU = G_p$.

RHPSS

Computes the matrix of sums of squares and crossproducts for the multivariate general linear hypothesis $H\,BU = G$ given the coefficient estimates.

RHPTE

Performs tests for a multivariate general linear hypothesis $H\,BU = G$ given the hypothesis sums of squares and crossproducts matrix S_H and the error sums of squares and crossproducts matrix S_E.

RLOFE

Computes a lack of fit test based on exact replicates for a fitted regression model.

RLOFN

Computes a lack of fit test based on near replicates for a fitted regression model.

RCASE

Computes case statistics and diagnostics given data points, coefficient estimates.

ROTIN

Computes diagnostics for detection of outliers and influential data points given residuals and the *R* matrix for a fitted general linear model.

UTILITIES FOR CLASSIFICATION VARIABLES

GCLAS

Gets the unique values of each classification variable.

GRGLM

Generates regressors for a general linear model.

VARIABLES SELECTION

RBEST

Selects the best multiple linear regression models.

RSTEP

Builds multiple linear regression models using forward selection, backward selection, or stepwise selection.

GSWEP

Performs a generalized sweep of a row of a nonnegative definite matrix.

RSUBM

Retrieves a symmetric submatrix from a symmetric matrix.

POLYNOMINAL REGRESSION AND SECOND-ORDER MODELS

POLYNOMINAL REGRESSION ANALYSIS

RCURV

Fits a polynomial curve using least squares.

RPOLY

Analyzes a polynomial regression model.

SECOND-ORDER MODEL DESIGN

RCOMP

Generates an orthogonal central composite design.

UTILITY ROUTINES FOR POLYNOMIAL MODELS AND SECOND-ORDER MODELS

RFORP

Fits an orthogonal polynomial regression model.

RSTAP

Computes summary statistics for a polynomial regression model given the fit based on orthogonal polynomials.

RCASP

Computes case statistics for a polynomial regression model given the fit based on orthogonal polynomials.

OPOLY

Generates orthogonal polynomials with respect to *x*-values and specified weights.

GCSCP

Generates centered variables, squares, and crossproducts.

TCSCP

Transforms coefficients from a second order response surface model generated from squares and crossproducts of centered variables to a model using uncentered variables.

NONLINEAR REGRESSION ANALYSIS

RNLIN

Fits a nonlinear regression model.

FITTING LINEAR MODELS BASED ON CRITERIA OTHER THAN LEAST SQUARES

RLAV

Fits a multiple linear regression model using the least absolute values criterion.

RLLP

Fits a multiple linear regression model using the L_p norm criterion.

RLMV

Fits a multiple linear regression model using the minimax criterion.

Chapter 3: Correlation

THE CORRELATION MATRIX

CORVC

Computes the variance-covariance or correlation matrix.

COVPL

Computes a pooled variance-covariance matrix from the observations.

PCORR

Computes partial correlations or covariances from the covariance or correlation matrix.

RBCOV

Computes a robust estimate of a covariance matrix and mean vector.

CORRELATION MEASURES FOR A CONTINGENCY TABLE

CTRHO

Estimates the bivariate normal correlation coefficient using a contingency table.

TETCC

Categorizes bivariate data and computes the tetrachoric correlation coefficient.

A DICHOTOMOUS VARIABLE WITH A CLASSIFICATION VARIABLE

BSPBS

Computes the biserial and point-biserial correlation coefficients for a dichotomous variable and a numerically measurable classification variable.

BSCAT

Computes the biserial correlation coefficient for a dichotomous variable and a classification variable.

MEASURES BASED UPON RANKS

CNCRD

Calculates and test the significance of the Kendall coefficient of concordance.

KENDL

Computes and test Kendall's rank correlation coefficient.

KENDP

Computes the frequency distribution of the total score in Kendall's rank correlation coefficient.

Chapter 4: Analysis of Variance

GENERAL ANALYSIS

AONEW

Analyzes a one-way classification model.

AONEC

Analyzes a one-way classification model with covariates.

ATWOB

Analyzes a randomized block design or a two-way balanced design.

ABIBD

Analyzes a balanced incomplete block design or a balanced lattice design.

ALATN

Analyzes a Latin square design.

ANWAY

Analyzes a balanced *n*-way classification model with fixed effects.

ABALD

Analyzes a balanced complete experimental design for a fixed, random, or mixed model.

ANEST

Analyzes a completely nested random model with possibly unequal numbers in the subgroups.

INFERENCE ON MEANS AND VARIANCE COMPONENTS

CTRST

Computes contrast estimates and sums of squares.

SCIPM

Computes simultaneous confidence intervals on all pairwise differences of means.

SNKMC

Performs Student-Newman-Keuls multiple comparison test.

CIDMS

Computes a confidence interval on a variance component estimated as proportional to the difference in two mean squares in a balanced complete experimental design.

SERVICE ROUTINE

ROREX

Reorders the responses from a balanced complete experimental design.

Chapter 5: Categorical and Discrete Data Analysis

STATISTICS IN THE TWO-WAY CONTINGENCY TABLE

CTTWO

Performs a chi-squared analysis of a 2 by 2 contingency table.

CTCHI

Performs a chi-squared analysis of a two-way contingency table.

CTPRB

Computes exact probabilities in a two-way contingency table.

CTEPR

Computes Fisher's exact test probability and a hybrid approximation to the Fisher exact test probability for a contingency table using the network algorithm.

LOG-LINEAR MODELS

PRPFT

Performs iterative proportional fitting of a contingency table using a loglinear model.

CTLLN

Computes model estimates and associated statistics for a hierarchical log-linear model.

CTPAR

Computes model estimates and covariances in a fitted log-linear model.

CTASC

Computes partial association statistics for log-linear models in a multidimensional contingency table.

CTSTP

Builds hierarchical log-linear models using forward selection, backward selection, or stepwise selection.

RANDOMIZATION TESTS

CTRAN

Performs generalized Mantel-Haenszel tests in a stratified contingency table.

GENERALIZED CATEGORICAL MODELS

CTGLM

Analyzes categorical data using logistic, Probit, Poisson, and other generalized linear models.

WEIGHTED LEAST-SQUARES ANALYSIS

CTWLS

Performs a generalized linear least-squares analysis of transformed probabilities in a two-dimensional contingency table.

Chapter 6: Nonparametric Statistics

ONE SAMPLE OR MATCHED SAMPLES

TESTS OF LOCATION

SIGNT

Performs a sign test of the hypothesis that a given value is a specified quantile of a distribution.

SNRNK

Performs a Wilcoxon signed rank test.

TESTS FOR TREND

NCTRD

Performs the Noether test for cyclical trend.

SDPLC

Performs the Cox and Stuart sign test for trends in dispersion and location

TIES

NTIES

Computes tie statistics for a sample of observations.

TWO INDEPENDENT SAMPLES

RNKSM

Performs the Wilcoxon rank sum test.

INCLD

Performs an includance test.

MORE THAN TWO SAMPLES

ONE WAY TESTS OF LOCATION

KRSKL

Performs a Kruskal-Wallis test for identical population medians.

BHAKV

Performs a Bhapkar *V* test.

TWO-WAY TESTS OF LOCATION

FRDMN

Performs Friedman's test for a randomized complete block design.

QTEST

Performs a Cochran *Q* test for related observations.

TESTS FOR TRENDS

KTRND

Performs *k*-sample trends test against ordered alternatives.

Chapter 7: Tests of Goodness-of-Fit and Randomness

GENERAL GOODNESS-OF-FIT TESTS FOR A SPECIFIED DISTRIBUTION

KSONE

Performs a Kolmogorov-Smirnov one-sample test for continuous distributions.

CHIGF

Performs a chi-squared goodness-of-fit test.

SPWLK

Performs a Shapiro-Wilk *W*-test for normality.

LILLF

Performs Lilliefors test for an exponential or normal distribution.

MVMMT

Computes Mardia's multivariate measures of skewness and kurtosis and test for multivariate normality.

TWO SAMPLE TESTS

KSTWO

Performs a Kolmogorov-Smirnov two-sample test.

TESTS FOR RANDOMNESS

RUNS

Performs a runs up test.

PAIRS

Performs a pairs test.

DSQAR

Performs a d^2 test.

DCUBE

Performs a triplets test.

Chapter 8: Time Series Analysis and Forecasting

GENERAL METHODOLOGY

TIME SERIES TRANSFORMATION

BCTR

Performs a forward or an inverse Box-Cox (power) transformation.

DIFF

Differences a time series.

SAMPLE CORRELATION FUNCTION

ACF

Computes the sample autocorrelation function of a stationary time series.

PACF

Computes the sample partial autocorrelation function of a stationary time series.

CCF

Computes the sample cross-correlation function of two stationary time series.

MCCF

Computes the multichannel cross-correlation function of two mutually stationary multichannel time series.

TIME DOMAIN METHODOLOGY

NONSEASONAL AUTOREGRESSIVE MOVING AVERAGE MODEL

ARMME

Computes method of moments estimates of the autoregressive parameters of an ARMA model.

MAMME

Computes method of moments estimates of the moving average parameters of an ARMA model.

NSPE

Computes preliminary estimates of the autoregressive and moving average parameters of an ARMA model.

NSLSE

Computes least-squares estimates of parameters for a nonseasonal ARMA model.

MAX_ARMA

Exact maximum likelihood estimation of the parameters in a univariate ARMA (auto-regressive, moving average) time series model.

GARCH

Computes estimates of the parameters of a GARCH(p,q) model.

SPWF

Computes the Wiener forecast operator for a stationary stochastic process.

NSBJF

Computes Box-Jenkins forecasts and their associated probability limits for a nonseasonal ARMA model.

TRANSFER FUNCTION MODEL

IRNSE

Computes estimates of the impulse response weights and noise series of a univariate transfer function model.

TFPE

Computes preliminary estimates of parameters for a univariate transfer function model.

MULTICHANNEL TIME SERIES

MLSE

Computes least-squares estimates of a linear regression model for a multichannel time series with a specified base channel.

MWFE

Computes least-squares estimates of the multichannel Wiener filter coefficients for two mutually stationary multichannel time series.

KALMN

Performs Kalman filtering and evaluates the likelihood function for the state-space model.

AUTOMATIC MODEL SELECTION FITTING

AUTO_UNI_AR

Automatic selection and fitting of a univariate autoregressive time series model.

AUTO_FPE_UNI_AR

Automatic selection and fitting of a univariate autoregressive time series model using Akaike's Final Prediction Error (FPE) criteria.

AUTO_MUL_AR
Automatic selection and fitting of a multivariate autoregressive time series model.

AUTO_FPE_MUL_AR
Automatic selection and fitting of a multivariate autoregressive time series model using Akaike's Multivariate Final Prediction Error (MFPE) criteria.

BAYESIAN TIME SERIES ESTIMATION

BAY_SEA
Allows for a decomposition of a time series into trend, seasonal, and an error component.

CONTROLLER DESIGN

OPT_DES
Allows for multiple channels for both the controlled and manipulated variables.

DIAGNOSTICS

LOFCF
Performs lack-of-fit test for a univariate time series or transfer function given the appropriate correlation function.

FREQUENCY DOMAIN METHODOLOGY

SMOOTHING FUNCTIONS

DIRIC
Computes the Dirichlet kernel.

FEJER
Computes the Fejér kernel.

SPECTRAL DENSITY ESTIMATION

ARMA_SPEC
Calculates the rational power spectrum for an ARMA model.

PFFT
Computes the periodogram of a stationary time series using a fast Fourier transform.

SSWD
Estimates the nonnormalized spectral density of a stationary time series using a spectral window given the time series data.

SSWP
Estimates the nonnormalized spectral density of a stationary time series using a spectral window given the periodogram.

SWED
Estimates the nonnormalized spectral density of a stationary time series based on specified periodogram weights given the time series data.

SWEP
Estimates the nonnormalized spectral density of a stationary time series based on specified periodogram weights given the periodogram.

CROSS-SPECTRAL DENSITY ESTIMATION

CPFFT
Computes the cross periodogram of two stationary time series using a fast Fourier transform.

CSSWD
Estimates the nonnormalized cross-spectral density of two stationary time series using a spectral window given the time series data.

CSSWP

Estimates the nonnormalized cross-spectral density of two stationary time series using a spectral window given the spectral densities and cross periodogram.

CSWED

Estimates the nonnormalized cross-spectral density of two stationary time series using a weighted cross periodogram given the time series data.

CSWEP

Estimates the nonnormalized cross-spectral density of two stationary time series using a weighted cross periodogram given the spectral densities and cross periodogram.

Chapter 9: Covariance Structures and Factor Analysis

PRINCIPAL COMPONENTS

PRINC

Computes principal components from a variance-covariance matrix or a correlation matrix.

KPRIN

Maximum likelihood or least-squares estimates for principal components from one or more matrices.

FACTOR ANALYSIS

FACTOR EXTRACTION

FACTR

Extracts initial factor loading estimates in factor analysis.

FACTOR ROTATION AND SUMMARIZATION

FROTA

Computes an orthogonal rotation of a factor loading matrix using a generalized orthomax criterion, including quartimax, varimax, and equamax rotations.

FOPCS

Computes an orthogonal Procrustes rotation of a factor-loading matrix using a target matrix.

FDOBL

Computes a direct oblimin rotation of a factor loading matrix.

FPRMX

Computes an oblique Promax or Procrustes rotation of a factor loading matrix using a target matrix, including pivot and power vector options.

FHARR

Computes an oblique rotation of an unrotated factor loading matrix using the Harris-Kaiser method.

FGCRF

Computes direct oblique rotation according to a generalized fourth-degree polynomial criterion.

FIMAG

Computes the image transformation matrix.

FRVAR

Computes the factor structure and the variance explained by each factor.

FACTOR SCORES

FCOEF

Computes a matrix of factor score coefficients for input to the routine `FSCOR`.

FSCOR

Computes a set of factor scores given the factor score coefficient matrix.

RESIDUAL CORRELATION

FRESI

Computes communalities and the standardized factor residual correlation matrix.

INDEPENDENCE OF SETS OF VARIABLES AND CANONICAL CORRELATION ANALYSIS

MVIND

Computes a test for the independence of *k* sets of multivariate normal variables.

CANCR

Performs canonical correlation analysis from a data matrix.

CANVC

Performs canonical correlation analysis from a variance-covariance matrix or a correlation matrix.

Chapter 10: Discriminant Analysis

PARAMETRIC DISCRIMINATION

DSCRM

Performs a linear or a quadratic discriminant function analysis among several known groups.

DMSCR

Uses Fisher's linear discriminant analysis method to reduce the number of variables.

NONPARAMETRIC DISCRIMINATION

NNBRD

Performs *k* nearest neighbor discrimination.

Chapter 11: Cluster Analysis

HIERARCHICAL CLUSTER ANALYSIS

CDIST

Computes a matrix of dissimilarities (or similarities) between the columns (or rows) of a matrix.

CLINK

Performs a hierarchical cluster analysis given a distance matrix.

CNUMB

Computes cluster membership for a hierarchical cluster tree.

K-MEANS CLUSTER ANALYSIS

KMEAN

Performs a *K*-means (centroid) cluster analysis.

Chapter 12: Sampling

SMPPR

Computes statistics for inferences regarding the population proportion and total given proportion data from a simple random sample.

SMPPS

Computes statistics for inferences regarding the population proportion and total given proportion data from a stratified random sample.

SMPRR

Computes statistics for inferences regarding the population mean and total using ratio or regression estimation, or inferences regarding the population ratio given a simple random sample.

SMPRS

Computes statistics for inferences regarding the population mean and total using ratio or regression estimation given continuous data from a stratified random sample.

SMPSC

Computes statistics for inferences regarding the population mean and total using single stage cluster sampling with continuous data.

SMPSR

Computes statistics for inferences regarding the population mean and total, given data from a simple random sample.

SMPSS

Computes statistics for inferences regarding the population mean and total, given data from a stratified random sample.

SMPST

Computes statistics for inferences regarding the population mean and total given continuous data from a two-stage sample with equisized primary units.

Chapter 13: Survival Analysis, Life Testing and Reliability

SURVIVAL ANALYSIS

KAPMR

Computes Kaplan-Meier estimates of survival probabilities in stratified samples.

KTBLE

Prints Kaplan-Meier estimates of survival probabilities in stratified samples.

TRNBL

Computes Turnbull's generalized Kaplan-Meier estimates of survival probabilities in samples with interval censoring.

PHGLM

Analyzes time event data via the proportional hazards model.

SVGLM

Analyzes censored survival data using a generalized linear model.

STBLE

Estimates survival probabilities and hazard rates for various parametric models.

ACTUARIAL TABLES

ACTBL

Produces population and cohort life tables.

Chapter 14: Multidimensional Scaling

MULTIDIMENSIONAL SCALING ROUTINES

MSIDV

Performs individual-differences multidimensional scaling for metric data using alternating least squares.

UTILITY ROUTINES

MSDST

Computes distances in a multidimensional scaling model.

MSSTN

Transforms dissimilarity/similarity matrices and replace missing values by estimates to obtain standardized dissimilarity matrices.

MSDBL

Obtains normalized product-moment (double centered) matrices from dissimilarity matrices.

MSINI

Computes initial estimates in multidimensional scaling models.

MSTRS

Computes various stress criteria in multidimensional scaling.

Chapter 15: Density and Hazard Estimation

ESTIMATES FOR A DENSITY

DESPL

Performs nonparametric probability density function estimation by the penalized likelihood method.

DESKN

Performs nonparametric probability density function estimation by the kernel method.

DNFFT

Computes Gaussian kernel estimates of a univariate density via the fast Fourier transform over a fixed interval.

DESPT

Estimates a probability density function at specified points using linear or cubic interpolation.

MODIFIED LIKELIHOOD ESTIMATES FOR HAZARDS

HAZRD

Performs nonparametric hazard rate estimation using kernel functions and quasi-likelihoods.

HAZEZ

Performs nonparametric hazard rate estimation using kernel functions. Easy-to-use version of HAZRD.

HAZST

Performs hazard rate estimation over a grid of points using a kernel function.

Chapter 16: Line Printer Graphics

HISTOGRAMS

VHSTP

Prints a vertical histogram.

VHS2P

Prints a vertical histogram with every bar subdivided into two parts.

HHSTP

Prints a horizontal histogram.

SCATTERPLOTS

SCTP

Prints a scatter plot of several groups of data

EXPLORATORY DATA ANALYSIS

BOXP

Prints boxplots for one or more samples.

STMLP

Prints a stem-and-leaf plot.

EMPIRICAL PROBABILITY DISTRIBUTION

CDFP

Prints a sample cumulative distribution function (CDF), a theoretical CDF, and confidence band information.

CDF2P

Prints a plot of two sample cumulative distribution functions.

PROBP

Prints a probability plot.

OTHER GRAPHICS ROUTINES

PLOTP

Prints a plot of up to 10 sets of points.

TREEP

Prints a binary tree.

Chapter 17: Probability Distribution Functions and Inverses

DISCRETE RANDOM VARIABLES: DISTRIBUTION FUNCTIONS AND PROBABILITY FUNCTIONS

BINDF

Evaluates the binomial distribution function.

BINPR

Evaluates the binomial probability function.

HYPDF

Evaluates the hypergeometric distribution function.

HYPPR

Evaluates the hypergeometric probability function.

POIDF

Evaluates the Poisson distribution function.

POIPR

Evaluates the Poisson probability function.

CONTINUOUS RANDOM VARIABLES: DISTRIBUTION FUNCTIONS AND THEIR INVERSES

AKS1DF

Evaluates the distribution function of the one-sided Kolmogorov-Smirnov goodness of fit D^+ or D^- test statistic based on continuous data for one sample.

AKS2DF

Evaluates the distribution function of the Kolmogorov-Smirnov goodness of fit *D* test statistic based on continuous data for two samples.

ANORDF

Evaluates the standard normal (Gaussian) distribution function.

ANORIN

Evaluates the standard normal (Gaussian) distribution function.

BETDF

Evaluates the beta probability distribution function.

BETIN

Evaluates the inverse of the beta distribution function.

BNRDF

Evaluates the bivariate normal distribution function.

CHIDF

Evaluates the chi-squared distribution function.

CHIIN

Evaluates the inverse of the chi-squared distribution function.

CSNDF

Evaluates the noncentral chi-squared distribution function.

CSNIN

Evaluates the inverse of the noncentral chi-squared function.

FDF

Evaluates the *F* distribution function.

FIN

Evaluates the inverse of the *F* distribution function.

GAMDF

Evaluates the gamma distribution function.

GAMIN

Evaluates the inverse of the gamma distribution function.

TDF

Evaluates the Student's *t* distribution function.

TIN

Evaluates the inverse of the Student's *t* distribution function.

TNDF

Evaluates the noncentral Student's *t* distribution function.

TNIN

Evaluates the inverse of the noncentral Student's *t* distribution function.

GENERAL CONTINUOUS RANDOM VARIABLES

GCDF

Evaluates a general continuous cumulative distribution function given ordinates of the density.

GCIN

Evaluates the inverse of a general continuous cumulative distribution function given ordinates of the density.

GFNIN

Evaluates the inverse of a general continuous cumulative distribution function given in a subprogram.

Chapter 18: Random Number Generation

UTILITY ROUTINES FOR RANDOM NUMBER GENERATORS

RNOPT

Selects the uniform (0,1) multiplicative congruential pseudorandom number generator.

RNOPG

Retrieves the indicator of the type of uniform random number generator.

RNSET

Initializes a random seed for use in the IMSL random number generators.

RNGET

Retrieves the current value of the seed used in the IMSL random number generators.

RNSES

Initializes the table in the IMSL random number generators that use shuffling.

RNGES

Retrieves the current value of the table in the IMSL random number generators that use shuffling.

RNSEF

Retrieves the array used in the IMSL `GFSR` random number generator.

RNGEF

Retrieves the current value of the array used in the IMSL `GFSR` random number generator.

RNISD

Determines a seed that yields a stream beginning 100,000 numbers beyond the beginning of the stream yielded by a given seed used in IMSL multiplicative congruential generators (with no shufflings).

BASIC UNIFORM DISTRIBUTION

RNUN

Generates pseudorandom numbers from a uniform (0, 1) distribution.

RNUNF

Generates a pseudorandom number from a uniform (0, 1) distribution.

UNIVARIATE DISCRETE DISTRIBUTIONS

RNBIN

Generates pseudorandom numbers from a binomial distribution.

RNGDA

Generates pseudorandom numbers from a general discrete distribution using an alias method.

RNGDS

Sets up table to generate pseudorandom numbers from a general discrete distribution.

RNGDT

Generates pseudorandom numbers from a general discrete distribution using a table lookup method.

RNGEO

Generates pseudorandom numbers from a geometric distribution.

RNHYP

Generates pseudorandom numbers from a hypergeometric distribution.

RNLGR

Generates pseudorandom numbers from a logarithmic distribution.

RNNBN

Generates pseudorandom numbers from a negative binomial distribution.

RNPOI

Generates pseudorandom numbers from a Poisson distribution.

RNUND

Generates pseudorandom numbers from a discrete uniform distribution.

UNIVARIATE CONTINUOUS DISTRIBUTIONS

RNBET

Generates pseudorandom numbers from a beta distribution.

RNCHI

Generates pseudorandom numbers from a chi-squared distribution.

RNCHY

Generates pseudorandom numbers from a Cauchy distribution.

RNEXP

Generates pseudorandom numbers from a standard exponential distribution.

RNEXT

Generates pseudorandom numbers from a mixture of two exponential distributions.

RNGAM

Generates pseudorandom numbers from a standard gamma distribution.

RNGCS

Sets up table to generate pseudorandom numbers from a general continuous distribution.

RNGCT

Generates pseudorandom numbers from a general continuous distribution.

RNLNL

Generates pseudorandom numbers from a lognormal distribution.

RNNOA

Generates pseudorandom numbers from a standard normal distribution using an acceptance/rejection method.

RNNOF

Generates a pseudorandom number from a standard normal distribution.

RNNOR

Generates pseudorandom numbers from a standard normal distribution using an inverse CDF method.

RNSTA

Generates pseudorandom numbers from a stable distribution.

RNSTT

Generates pseudorandom numbers from a Student's *t* distribution.

RNTRI

Generates pseudorandom numbers from a triangular distribution on the interval (0, 1).

RNVMS

Generates pseudorandom numbers from a von Mises distribution.

RNWIB

Generates pseudorandom numbers from a Weibull distribution.

MULTIVARIATE DISTRIBUTIONS

RNCOR

Generates a pseudorandom orthogonal matrix or a correlation matrix.

RNDAT

Generates pseudorandom numbers from a multivariate distribution determined from a given sample.

RNMTN

Generates pseudorandom numbers from a multinomial distribution.

RNMVN

Generates pseudorandom numbers from a multivariate normal distribution.

RNSPH

Generates pseudorandom points on a unit circle or K-dimensional sphere.

RNTAB

Generates a pseudorandom two-way table.

ORDER STATISTICS

RNNOS

Generates pseudorandom order statistics from a standard normal distribution.

RNUNO

Generates pseudorandom order statistics from a uniform (0, 1) distribution.

STOCHASTIC PROCESSES

RNARM

Generates a time series from a specified ARMA model.

RNNPP

Generates pseudorandom numbers from a nonhomogenous Poisson process.

SAMPLES AND PERMUTATIONS

RNPER

Generates a pseudorandom permutation.

RNSRI

Generates a simple pseudorandom sample of indices.

RNSRS

Generates a simple pseudorandom sample from a finite population.

LOW DISCREPANCY SEQUENCES

FAURE_INIT

Generates pseudorandom numbers from a uniform (0, 1) distribution.

FAURE_FREE

Frees the structure containing information about the Faure sequence

FAURE_NEXT

Computes a shuffled Faure sequence.

Chapter 19: Utilities

PRINT

WRRRN

Prints a real rectangular matrix with integer row and column labels.

WRRRL

Prints a real rectangular matrix with a given format and labels.

WRIRN

Prints an integer rectangular matrix with integer row and column labels.

WRIRL

Prints an integer rectangular matrix with a given format and labels.

WROPT
Sets or retrieves an option for printing a matrix.

PGOPT
Sets or retrieves page width and length for printing.

PERMUTE

PERMU
Rearranges the elements of an array as specified by a permutation.

PERMA
Permutes the rows or columns of a matrix.

RORDM
Reorders rows and columns of a symmetric matrix.

MVNAN
Moves any rows of a matrix with the IMSL missing value code NaN (not a number) in the specified columns to the last rows of the matrix.

SORT

SVRGN
Sorts a real array by algebraically increasing value.

SVRGP
Sorts a real array by algebraically increasing value and returns the permutation that rearranges the array.

SVIGN
Sorts an integer array by algebraically increasing value.

SVIGP
Sorts an integer array by algebraically increasing value and returns the permutation that rearranges the array.

SCOLR
Sorts columns of a real rectangular matrix using keys in rows.

SROWR
Sorts rows of a real rectangular matrix using keys in columns.

SEARCH

SRCH
Searches a sorted vector for a given scalar and returns its index.

ISRCH
Searches a sorted integer vector for a given integer and returns its index.

SSRCH
Searches a character vector, sorted in ascending ASCII order, for a given string and returns its index.

CHARACTER STRING MANIPULATION

ACHAR
Returns a character given its ASCII value.

IACHAR
Returns the integer ASCII value of a character argument.

ICASE
Returns the ASCII value of a character converted to uppercase.

IICSR
Compares two character strings using the ASCII collating sequence but without regard to case.

IIDEX
Determines the position in a string at which a given character sequence begins without regard to case.

CVTSI
Converts a character string containing an integer number into the corresponding integer form.

TIME, DATE AND VERSION

CPSEC
Returns CPU time used in seconds.

TIMDY
Gets time of day.

TDATE
Gets today's date.

NDAYS
Computes the number of days from January 1, 1900, to the given date.

NDYIN
Gives the date corresponding to the number of days since January 1, 1900.

IDYWK
Computes the day of the week for a given date.

VERSL
Obtains STAT/LIBRARY-related version, system and serial numbers.

RETRIEVAL OF DATA SETS

GDATA
Retrieves a commonly analyzed data set.

Chapter 20: Mathematical Support

LINEAR SYSTEMS

GIRTS
Solves a triangular linear system given R.

CHFAC
Cholesky factorization R^TR of a nonnegative definite matrix

MCHOL
Modified Cholesky factorization

SPECIAL FUNCTIONS

ENOS
Expected value of a normal order statistic

AMILLR
Mill's ratio

NEAREST NEIGHBORS

QUADT
Forms a *k*-*d* tree

NGHBR
Searches a *k*-*d* tree for the *m* nearest neighbors