

# IMSL<sup>®</sup> Fortran Numerical Library

Function Catalog

Version 7.0





**CORPORATE HEADQUARTERS**

Rogue Wave Software  
5500 Flatiron Parkway Suite 200  
Boulder, CO 80301 USA  
[www.roguewave.com](http://www.roguewave.com)

**IMSL Libraries Contact Information**

[Contact Support](#)

**Products Overview**

[Products Overview](#)

© 1970-2012 Rogue Wave Software. The Rogue Wave Software name and logo, IMSL and PV-WAVE are registered trademarks of Rogue Wave Software, Inc. or its subsidiaries in the US and other countries. JMSL, JWAVE, TS-WAVE, and PyIMSL are trademarks of Rogue Wave Software, Inc. or its subsidiaries. All other company, product or brand names are the property of their respective owners.

**IMPORTANT NOTICE:** Information contained in this documentation is subject to change without notice. Use of this document is subject to the terms and conditions of a Rogue Wave Software License Agreement, including, without limitation, the Limited Warranty and Limitation of Liability. If you do not accept the terms of the license agreement, you may not use this documentation and should promptly return the product for a full refund. This documentation may not be copied or distributed in any form without the express written consent of Rogue Wave.

**IMSL<sup>®</sup>** C, C#, Java<sup>™</sup> and Fortran Application Development Tools

# Table of Contents

<b>IMSL® FORTRAN NUMERICAL LIBRARY VERSION 7.0</b>	<b>3</b>
Mathematical Functionality Overview	8
Mathematical Special Functions Overview	9
Statistical Functionality Overview	10
IMSL® Libraries are also available for C, Java, C# for .Net and Python	11
<b>IMSL MATH LIBRARY</b>	<b>12</b>
CHAPTER 1: LINEAR SYSTEMS .....	12
CHAPTER 2: EIGENSYSTEM ANALYSIS .....	22
CHAPTER 3: INTERPOLATION AND APPROXIMATION .....	26
CHAPTER 4: INTEGRATION AND DIFFERENTIATION .....	30
CHAPTER 5: DIFFERENTIAL EQUATIONS .....	32
CHAPTER 6: TRANSFORMS .....	34
CHAPTER 7: NONLINEAR EQUATIONS .....	36
CHAPTER 8: OPTIMIZATION .....	37
CHAPTER 9: BASIC MATRIX/VECTOR OPERATIONS .....	40
CHAPTER 10: LINEAR ALGEBRA OPERATORS AND GENERIC FUNCTIONS .....	49
CHAPTER 11: UTILITIES .....	51
<b>IMSL MATH SPECIAL FUNCTIONS LIBRARY</b>	<b>57</b>
CHAPTER 1: ELEMENTARY FUNCTIONS .....	57
CHAPTER 2: HYPERBOLIC FUNCTIONS .....	57
CHAPTER 3: EXPONENTIAL INTEGRALS AND RELATED FUNCTIONS .....	58
CHAPTER 4: GAMMA FUNCTION AND RELATED FUNCTIONS .....	59
CHAPTER 5: ERROR FUNCTIONS AND RELATED FUNCTIONS .....	61
CHAPTER 6: BESSEL FUNCTIONS .....	62
CHAPTER 7: KELVIN FUNCTIONS .....	63
CHAPTER 8: AIRY FUNCTIONS .....	64
CHAPTER 9: ELLIPTIC FUNCTIONS .....	65
CHAPTER 10: ELLIPTIC AND RELATED FUNCTIONS .....	66
CHAPTER 11: PROBABILITY DISTRIBUTIONS FUNCTIONS AND INVERSES .....	67
CHAPTER 12: MATHIEU FUNCTIONS .....	71
CHAPTER 13: MISCELLANEOUS FUNCTIONS .....	72
REFERENCE MATERIAL: LIBRARY ENVIRONMENTS UTILITIES .....	72
<b>IMSL STAT LIBRARY</b>	<b>73</b>
CHAPTER 1: BASIC STATISTICS .....	73
CHAPTER 2: REGRESSION .....	74
CHAPTER 3: CORRELATION .....	78
CHAPTER 4: ANALYSIS OF VARIANCE .....	79
CHAPTER 5: CATEGORICAL AND DISCRETE DATA ANALYSIS .....	80
CHAPTER 6: NONPARAMETRIC STATISTICS .....	81
CHAPTER 7: TESTS OF GOODNESS-OF-FIT AND RANDOMNESS .....	82
CHAPTER 8: TIME SERIES ANALYSIS AND FORECASTING .....	83
CHAPTER 9: COVARIANCE STRUCTURES AND FACTOR ANALYSIS .....	87
CHAPTER 10: DISCRIMINANT ANALYSIS .....	89
CHAPTER 11: CLUSTER ANALYSIS .....	89
CHAPTER 12: SAMPLING .....	90

<b>CHAPTER 13: SURVIVAL ANALYSIS, LIFE TESTING AND RELIABILITY .....</b>	<b>90</b>
<b>CHAPTER 14: MULTIDIMENSIONAL SCALING.....</b>	<b>91</b>
<b>CHAPTER 15: DENSITY AND HAZARD ESTIMATION.....</b>	<b>92</b>
<b>CHAPTER 16: LINE PRINTER GRAPHICS .....</b>	<b>92</b>
<b>CHAPTER 17: PROBABILITY DISTRIBUTIONS FUNCTIONS AND INVERSES....</b>	<b>93</b>
<b>CHAPTER 18: RANDOM NUMBER GENERATION .....</b>	<b>98</b>
<b>CHAPTER 19: UTILITIES.....</b>	<b>103</b>
<b>CHAPTER 20: MATHEMATICAL SUPPORT .....</b>	<b>106</b>

# IMSL® FORTRAN NUMERICAL LIBRARY VERSION 7.0

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 Numerical Library Version 7.0 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 the building blocks that eliminate the need to write code from scratch. These pre-written functions allow you to focus on your domain of expertise and reduce your development time.



## ONE COMPREHENSIVE PACKAGE

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

### INTERFACE MODULES

The IMSL Fortran Numerical Library Version 7.0 includes 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.

This simple and flexible interface to the library routines speeds programming and simplifies documentation. The IMSL Fortran Numerical Library takes full advantage of the intrinsic characteristics and desirable features of the Fortran language.

### BACKWARD COMPATIBILITY

The IMSL Fortran Numerical Library Version 7.0 maintains full backward compatibility with earlier releases of the 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 as well as calls to the IMSL F90 Library.

## SMP/OPENMP SUPPORT

The IMSL Fortran Numerical Library has also been designed to take advantage of symmetric multiprocessor (SMP) systems. Computationally intensive algorithms in areas such as linear algebra 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 Numerical Library with optimized routines from your hardware vendor, you can improve the performance of your numerical calculations.

### MPI ENABLED

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

## LAPACK AND SCALAPACK

LAPACK was designed to make the linear solvers and eigensystem routines run more efficiently on high performance computers. For a number of IMSL routines, the user of the IMSL Fortran Numerical Library has the option of linking to code which is based on either the legacy routines or the more efficient LAPACK routines. To obtain improved performance we recommend linking with vendor High Performance versions of LAPACK and BLAS, if available.

ScaLAPACK includes a subset of LAPACK codes redesigned for use on distributed memory MIMD parallel computers. Use of the ScaLAPACK enhanced routines allows a user to solve large linear systems of algebraic equations at a performance level that might not be achievable on one computer by performing the work in parallel across multiple computers.

IMSL facilitates the use of parallel computing in these situations by providing interfaces to ScaLAPACK routines which accomplish the task. The IMSL Library solver interface has the same look and feel whether one is using the routine on a single computer or across multiple computers.

## USER FRIENDLY NOMENCLATURE

The IMSL Fortran Numerical 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:

- Allow faster and easier program debugging
- Provide more productive programming and confidence that the algorithms are functioning properly.

## COST-EFFECTIVE

The IMSL Fortran Numerical Library significantly shortens program development time and promotes standardization. Using the IMSL Fortran Numerical Library saves time in source code development and the design, development, documentation, testing and maintenance of applications.

## FULLY-TESTED

IMSL has almost four decades of experience in testing IMSL numerical algorithms for quality and performance across an extensive range of the latest compilers and environments. This experience has allowed IMSL to refine its test methods to a great level of detail. The result of this effort is a robust, sophisticated suite of test methods that allows the IMSL user to rely on the numerical analysis functionality and focus their bandwidth on application development and testing.

## WIDE COMPATIBILITY AND UNIFORM OPERATION

The IMSL Fortran Numerical Library is available for major UNIX computing environments, Linux, and Microsoft Windows. IMSL performs extensive compatibility testing to ensure that the library is compatible with each supported computing environment.

## COMPREHENSIVE DOCUMENTATION

Documentation for the IMSL Fortran Numerical Library is comprehensive, clearly written and standardized. Detailed information about each subroutine consists of the name, purpose, synopsis, exceptions, return values and usage examples.



## UNMATCHED PRODUCT SUPPORT

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

Product support:

- Gives users direct access to IMSL resident staff of expert product support specialists
- Provides prompt, two-way communication
- Includes product maintenance updates

## CONSULTING SERVICES

Rogue Wave Software offers expert consulting services for algorithm development as well as complete application development. Please contact Rogue Wave Software to learn more about its extensive experience in developing custom algorithms, building algorithms in scalable platforms, and full applications development.

# Mathematical Functionality Overview

The IMSL Fortran Numerical Library is a collection of the most commonly needed numerical functions customized for your programming needs. The mathematical functionality is organized into eleven 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.
- **Eigensystems 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** - including Adams-Gear and Runge-Kutta methods for stiff and non-stiff 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, 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 and the fastest linear programming algorithm available in a general math library.
- **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, machine, mathematical, physical constants, retrieval of machine constants and customizable error-handling.

# Mathematical Special Functions Overview

The IMSL Fortran Numerical 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 twelve 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 and integer order with both real and complex arguments
- **Kelvin Functions** - including Kelvin functions and their derivatives
- **Airy Functions** - including Airy functions, complex 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 Overview

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

- **Basic Statistics** - including univariate summary statistics, frequency tables, and rank and order statistics.
- **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.
- **Probability Distribution Functions and Inverses** - including binomial, hypergeometric, bivariate normal, gamma and many more.
- **Random Number Generation** - including the Mersenne Twister generator and a generator for multivariate normal distributions and pseudorandom numbers from several distributions, including gamma, Poisson, beta, and low discrepancy sequence.
- **Utilities** - including CPU time used, machine, mathematical, physical constants, retrieval of machine constants and customizable error-handling.
- **Mathematical Support** - including linear systems, special functions, and nearest neighbors.

# IMSL® Libraries are also available for C, Java, C# for .Net and Python

## IMSL C Numerical Library

The IMSL C Numerical Library delivers advanced mathematical and statistical functionality for programmers to embed in C/C++ applications. This comprehensive set of functions is based upon the same algorithms contained in the highly regarded IMSL Fortran Library. The IMSL C Library is available on a wide range of development platforms and offers functions in key areas such as optimization, data mining, forecasting and design of experiments analysis. These pre-tested functions result in superior performance, increased scalability, ease of integration and greater reliability for software applications that require advanced mathematics and statistics. Dozens of algorithms take advantage of multi-core hardware using standard OpenMP directives.

## JMSL™ Numerical Library for Java Applications

The JMSL Numerical Library for Java applications is the broadest collection of mathematical, statistical, financial, data mining and charting classes available in 100% Java. It is the only Java programming solution that combines integrated charting with the reliable mathematical and statistical functionality of the industry-leading IMSL Numerical Library algorithms. This blend of advanced numerical analysis and visualization on the Java platform allows organizations to gain insight into valuable data and share analysis results across the enterprise quickly. The JMSL Library continues to be the leader, providing robust data analysis and visualization technology for the Java platform and a fast, scalable framework for tailored analytical applications.

## IMSL C# Numerical Library for Microsoft® .Net Applications

The IMSL C# Numerical Library for Microsoft® .NET Applications is a numerical analysis library written in 100% C#, providing broad coverage of advanced mathematics and statistics for the .NET Framework. This offers .NET developers seamless accessibility to analytics capabilities in the most integrated language for the .NET environment with the highest degree of programming productivity and ease of use with Visual Studio®.

The IMSL C# Library is the only numerical library to offer industry standard numerical analysis and charting for .NET languages. This Library provides unprecedented analytic capabilities and the most comprehensive and accessible mathematical, statistical and finance algorithms for .NET languages. With the IMSL C# Library, IMSL has brought all of the benefits inherent in the .NET Framework to a new level by adding robust analytics to its broad set of capabilities. Written in C#, the library is easily integrated into any .NET language such as Visual Basic .NET, F# and IronPython among others.

## PyIMSL™ Studio Transforming Analytic Application Data

PyIMSL Studio is the only commercially-available numerical analysis application development environment designed for deploying mathematical and statistical prototype models into production applications. PyIMSL Studio closes the prototype to production gap by providing modelers and implementation teams with a common set of tested and supported high-quality development tools as well as the same underlying numerical algorithms. Using PyIMSL Studio, prototype work is transformed into production applications faster, with less complexity, cost and risk to the project.

# IMSL MATH LIBRARY

## CHAPTER 1: LINEAR SYSTEMS

### LINEAR SOLVERS

ROUTINE	DESCRIPTION
<b>LIN_SOL_GEN</b>	Solves a real general system of linear equations $Ax = b$ .
<b>LIN_SOL_SELF</b>	Solves a system of linear equations $Ax = b$ , where $A$ is a self-adjoint matrix.
<b>LIN_SOL_LSQ</b>	Solves a rectangular system of linear equations $Ax \cong b$ , in a least-squares sense.
<b>LIN_SOL_SVD</b>	Solves a rectangular least-squares system of linear equations $Ax \cong b$ using singular value decomposition.
<b>LIN_SOL_TRI</b>	Solves multiple systems of linear equations.
<b>LIN_SVD</b>	Computes the singular value decomposition (SVD) of a rectangular matrix, $A$ .

### LARGE-SCALE PARALLEL SOLVERS

ROUTINE	DESCRIPTION
<b>PARALLEL_NONNEGATIVE_LSQ</b>	Solves a linear, non-negative constrained least-squares system.
<b>PARALLEL_BOUNDED_LSQ</b>	Solves a linear least-squares system with bounds on the unknowns

### SOLUTION OF LINEAR SYSTEMS, MATRIX INVERSION, AND DETERMINANT EVALUATION

#### REAL GENERAL MATRICES

ROUTINE	DESCRIPTION
<b>LSARG</b>	Solves a real general system of linear equations with iterative refinement.
<b>LSLRG</b>	Solves a real general system of linear equations without iterative refinement.

## REAL GENERAL MATRICES

ROUTINE	DESCRIPTION
<b>LFCRG</b>	Computes the $LU$ factorization of a real general matrix and estimates its $L_1$ condition number.
<b>LFTRG</b>	Computes the $LU$ factorization of a real general matrix.
<b>LFSRG</b>	Solves a real general system of linear equations given the $LU$ factorization of the coefficient matrix.
<b>LFIRG</b>	Uses iterative refinement to improve the solution of a real general system of linear equations.
<b>LFDRG</b>	Computes the determinant of a real general matrix given the $LU$ factorization of the matrix.
<b>LINRG</b>	Computes the inverse of a real general matrix.

## COMPLEX GENERAL MATRICES

ROUTINE	DESCRIPTION
<b>LSACG</b>	Solves a complex general system of linear equations with iterative refinement.
<b>LSLCG</b>	Solves a complex general system of linear equations without iterative refinement.
<b>LFCCG</b>	Computes the $LU$ factorization of a complex general matrix and estimates its $L_1$ condition number.
<b>LFTCG</b>	Computes the $LU$ factorization of a complex general matrix.
<b>LFSCG</b>	Solves a complex general system of linear equations given the $LU$ factorization of the coefficient matrix.
<b>LFICG</b>	Uses iterative refinement to improve the solution of a complex general system of linear equations.
<b>LFDCG</b>	Computes the determinant of a complex general matrix given the $LU$ factorization of the matrix.
<b>LINCG</b>	Computes the inverse of a complex general matrix.

## REAL TRIANGULAR MATRICES

ROUTINE	DESCRIPTION
<b>LSLRT</b>	Solves a real triangular system of linear equations.
<b>LFCRT</b>	Estimates the condition number of a real triangular matrix.
<b>LFDRT</b>	Computes the determinant of a real triangular matrix.
<b>LINRT</b>	Computes the inverse of a real triangular matrix.

## COMPLEX TRIANGULAR MATRICES

ROUTINE	DESCRIPTION
<b>LSLCT</b>	Solves a complex triangular system of linear equations.
<b>LFCCT</b>	Estimates the condition number of a complex triangular matrix.
<b>LFDCT</b>	Computes the determinant of a complex triangular matrix.
<b>LINCT</b>	Computes the inverse of a complex triangular matrix.

## REAL POSITIVE DEFINITE MATRICES

ROUTINE	DESCRIPTION
<b>LSADS</b>	Solves a real symmetric positive definite system of linear equations with iterative refinement.
<b>LSLDS</b>	Solves a real symmetric positive definite system of linear equations without iterative refinement.
<b>LFCDS</b>	Computes the $R^T R$ Cholesky factorization of a real symmetric positive definite matrix and estimates its $L_1$ condition number.
<b>LFTDS</b>	Computes the $R^T R$ Cholesky factorization of a real symmetric positive definite matrix.
<b>LFSDS</b>	Solves a real symmetric positive definite system of linear equations given the $R^T R$ Cholesky factorization of the coefficient matrix.
<b>LFIDS</b>	Uses the iterative refinement to improve the solution of a real symmetric positive definite system of linear equations.



## REAL POSITIVE DEFINITE MATRICES

ROUTINE	DESCRIPTION
<b>LFDDS</b>	Computes the determinant of a real symmetric positive matrix given the $R^T R$ Cholesky factorization of the matrix.
<b>LINDS</b>	Computes the inverse of a real symmetric positive definite matrix.

## REAL SYMMETRIC MATRICES

ROUTINE	DESCRIPTION
<b>LSASF</b>	Solves a real symmetric system of linear equations with iterative refinement.
<b>LLSF</b>	Solves a real symmetric system of linear equations without iterative refinement.
<b>LFCSF</b>	Computes the $U D U^T$ factorization of a real symmetric matrix and estimates its $L_1$ condition number.
<b>LFTSF</b>	Computes the $U D U^T$ factorization of a real symmetric matrix.
<b>LFSSF</b>	Solves a real symmetric system of linear equations given the $U D U^T$ factorization of the coefficient matrix.
<b>LFISF</b>	Uses iterative refinement to improve the solution of a real symmetric system of linear equations.
<b>LFDSF</b>	Computes the determinant of a real symmetric matrix given the $U D U^T$ factorization of the matrix.

## COMPLEX HERMITIAN POSITIVE DEFINITE MATRICES

ROUTINE	DESCRIPTION
<b>LSADH</b>	Solves a complex Hermitian positive definite system of linear equations with iterative refinement.
<b>LSLDH</b>	Solves a complex Hermitian positive definite system of linear equations without iterative refinement.
<b>LFCDH</b>	Computes the $R^H R$ factorization of a complex Hermitian positive definite matrix and estimates its $L_1$ condition number.
<b>LFTDH</b>	Computes the $R^H R$ factorization of a complex Hermitian positive definite matrix.
<b>LFSDH</b>	Solves a complex Hermitian positive definite system of linear equations given the $R^H R$ factorization of the coefficient matrix.

## COMPLEX HERMITIAN POSITIVE DEFINITE MATRICES

ROUTINE	DESCRIPTION
<b>LFIDH</b>	Uses the iterative refinement to improve the solution of a complex Hermitian positive definite system of linear equations.
<b>LFDDH</b>	Computes the determinant of a complex Hermitian positive definite matrix given the $R^H R$ Cholesky factorization of the matrix.

## COMPLEX HERMITIAN MATRICES

ROUTINE	DESCRIPTION
<b>LSAHF</b>	Solves a complex Hermitian system of linear equations with iterative refinement.
<b>LSLHF</b>	Solves a complex Hermitian system of linear equations without iterative refinement.
<b>LFCHF</b>	Computes the $U D U^H$ factorization of a complex Hermitian matrix and estimates its $L_1$ condition number.
<b>LFTHF</b>	Computes the $U D U^H$ factorization of a complex Hermitian matrix.
<b>LFSHF</b>	Solves a complex Hermitian system of linear equations given the $U D U^H$ factorization of the coefficient matrix.
<b>LFIHF</b>	Uses iterative refinement to improve the solution of a complex Hermitian system of linear equations.
<b>LFDFH</b>	Computes the determinant of a complex Hermitian matrix given the $U D U^H$ factorization of the matrix.

## REAL BAND MATRICES IN BAND STORAGE MODE

ROUTINE	DESCRIPTION
<b>LSLTR</b>	Solves a real tridiagonal system of linear equations.
<b>LSLCR</b>	Computes the $L D U$ factorization of a real tridiagonal matrix $A$ using a cyclic reduction algorithm.
<b>LSARB</b>	Solves a real system of linear equations in band storage mode with iterative refinement.
<b>LSLRB</b>	Solves a real system of linear equations in band storage mode without iterative refinement.

## REAL BAND MATRICES IN BAND STORAGE MODE

ROUTINE	DESCRIPTION
<b>LFCRB</b>	Computes the $LU$ factorization of a real matrix in band storage mode and estimates its $L_1$ condition number.
<b>LFTRB</b>	Computes the $LU$ factorization of a real matrix in band storage mode.
<b>LFSRB</b>	Solves a real system of linear equations given the $LU$ factorization of the coefficient matrix in band storage mode.
<b>LFIRB</b>	Uses iterative refinement to improve the solution of a real system of linear equations in band storage mode.
<b>LFDRB</b>	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

ROUTINE	DESCRIPTION
<b>LSAQS</b>	Solves a real symmetric positive definite system of linear equations in band symmetric storage mode with iterative refinement.
<b>LSLQS</b>	Solves a real symmetric positive definite system of linear equations in band symmetric storage mode without iterative refinement.
<b>LSLPB</b>	Computes the $R^TDR$ Cholesky factorization of a real symmetric positive definite matrix $A$ in codiagonal band symmetric storage mode. Solves a system $Ax = b$ .
<b>LFCQS</b>	Computes the $R^TR$ Cholesky factorization of a real symmetric positive definite matrix in band symmetric storage mode and estimates its $L_1$ condition number.
<b>LFTQS</b>	Computes the $R^TR$ Cholesky factorization of a real symmetric positive definite matrix in band symmetric storage mode.
<b>LFSQS</b>	Solves a real symmetric positive definite system of linear equations given the factorization of the coefficient matrix in band symmetric storage mode.
<b>LFIQS</b>	Uses iterative refinement to improve the solution of a real symmetric positive definite system of linear equations in band symmetric storage mode.
<b>LFDQS</b>	Computes the determinant of a real symmetric positive definite matrix given the $R^TR$ Cholesky factorization of the matrix in band symmetric storage mode.

### COMPLEX BAND MATRICES IN BAND STORAGE MODE

ROUTINE	DESCRIPTION
<b>LSLTQ</b>	Solves a complex tridiagonal system of linear equations.
<b>LSLCQ</b>	Computes the $LDU$ factorization of a complex tridiagonal matrix $A$ using a cyclic reduction algorithm.
<b>LSACB</b>	Solves a complex system of linear equations in band storage mode with iterative refinement.
<b>LSLCB</b>	Solves a complex system of linear equations in band storage mode without iterative refinement.
<b>LFCCB</b>	Computes the $LU$ factorization of a complex matrix in band storage mode and estimates its $L_1$ condition number. the $UDU^H$ factorization of the coefficient matrix.
<b>LFTCB</b>	Computes the $LU$ factorization of a complex matrix in band storage mode given the coefficient matrix in band storage mode.
<b>LFSCB</b>	Solves a complex system of linear equations given the $LU$ factorization of the coefficient matrix in band storage mode.
<b>LFICB</b>	Uses iterative refinement to improve the solution of a complex system of linear equations in band storage mode.
<b>LFDCB</b>	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

ROUTINE	DESCRIPTION
<b>LSAQH</b>	Solves a complex Hermitian positive definite system of linear equations in band Hermitian storage mode with iterative refinement.
<b>LSLQH</b>	Solves a complex Hermitian positive definite system of linear equations in band Hermitian storage mode without iterative refinement.
<b>LSLQB</b>	Computes the $R^H DR$ Cholesky factorization of a complex Hermitian positive-definite matrix $A$ in codiagonal band Hermitian storage mode. Solves a system $Ax = b$ .
<b>LFCQH</b>	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.
<b>LFTQH</b>	Computes the $R^H R$ factorization of a complex Hermitian positive definite matrix in band Hermitian storage mode.

## COMPLEX BAND POSITIVE DEFINITE MATRICES IN BAND STORAGE MODE

ROUTINE	DESCRIPTION
<b>LFSQH</b>	Solves a complex Hermitian positive definite system of linear equations given the factorization of the coefficient matrix in band Hermitian storage mode.
<b>LFIQH</b>	Uses iterative refinement to improve the solution of a complex Hermitian positive definite system of linear equations in band Hermitian storage mode.
<b>LFDQH</b>	Computes the determinant of a complex Hermitian positive definite matrix given the $R^H R$ Cholesky factorization in band Hermitian storage mode.

## REAL SPARSE LINEAR EQUATION SOLVERS

ROUTINE	DESCRIPTION
<b>LSLXG</b>	Solves a sparse system of linear algebraic equations by Gaussian elimination.
<b>LFTXG</b>	Computes the $LU$ factorization of a real general sparse matrix.
<b>LFSXG</b>	Solves a sparse system of linear equations given the $LU$ factorization of the coefficient matrix.

## COMPLEX SPARSE LINEAR EQUATION SOLVERS

ROUTINE	DESCRIPTION
<b>LSLZG</b>	Solves a complex sparse system of linear equations by Gaussian elimination.
<b>LSTZG</b>	Computes the $LU$ factorization of a complex general sparse matrix.
<b>LFSZG</b>	Solves a complex sparse system of linear equations given the $LU$ factorization of the coefficient matrix.

## REAL SPARSE SYMMETRIC POSITIVE DEFINITE LINEAR EQUATIONS SOLVERS

ROUTINE	DESCRIPTION
<b>LSLXD</b>	Solves a sparse system of symmetric positive definite linear algebraic equations by Gaussian elimination.
<b>LSCXD</b>	Performs the symbolic Cholesky factorization for a sparse symmetric matrix using a minimum degree ordering or a user-specified ordering, and sets up the data structure for the numerical Cholesky factorization.
<b>LNFXD</b>	Computes the numerical Cholesky factorization of a sparse symmetrical matrix $A$ .

## REAL SPARSE SYMMETRIC POSITIVE DEFINITE LINEAR EQUATIONS SOLVERS

ROUTINE	DESCRIPTION
<b>LFSXD</b>	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

ROUTINE	DESCRIPTION
<b>LSLZD</b>	Solves a complex sparse Hermitian positive definite system of linear equations by Gaussian elimination.
<b>LNFDZD</b>	Computes the numerical Cholesky factorization of a sparse Hermitian matrix <b>A</b> .
<b>LFSZD</b>	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

ROUTINE	DESCRIPTION
<b>LSLTO</b>	Solves a real Toeplitz linear system.

## COMPLEX TOEPLITZ MATRICES IN TOEPLITZ STORAGE MODE

ROUTINE	DESCRIPTION
<b>LSLTC</b>	Solves a complex Toeplitz linear system.

## COMPLEX CIRCULAR MATRICES IN CIRCULANT STORAGE MODE

ROUTINE	DESCRIPTION
<b>LSLCC</b>	Solves a complex circulant linear system.

## ITERATIVE METHODS

ROUTINE	DESCRIPTION
<b>PCGRC</b>	Solves a real symmetric definite linear system using a preconditioned conjugate gradient method with reverse communication.
<b>JCGRC</b>	Solves a real symmetric definite linear system using the Jacobi-preconditioned conjugate gradient method with reverse communication.
<b>GMRES</b>	Uses <b>GMRES</b> 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

ROUTINE	DESCRIPTION
<b>LSQRR</b>	Solves a linear least-squares problem without iterative refinement.
<b>LQRRV</b>	Computes the least-squares solution using Householder transformations applied in blocked form.
<b>LSBRR</b>	Solves a linear least-squares problem with iterative refinement.
<b>LCLSQ</b>	Solves a linear least-squares problem with linear constraints.
<b>LQRRR</b>	Computes the $QR$ decomposition, $AP = QR$ , using Householder transformations.
<b>LQERR</b>	Accumulate the orthogonal matrix $Q$ from its factored form given the $QR$ factorization of a rectangular matrix $A$ .
<b>LQRSL</b>	Computes the coordinate transformation, projection, and complete the solution of the least-squares problem $Ax = b$ .
<b>LUPQR</b>	Computes an updated QR factorization after the rank-one matrix $axy^T$ is added.

### CHOLESKY FACTORIZATION

ROUTINE	DESCRIPTION
<b>LCHRG</b>	Computes the Cholesky decomposition of a symmetric positive semidefinite matrix with optional column pivoting.
<b>LUPCH</b>	Updates the $R^T R$ Cholesky factorization of a real symmetric positive definite matrix after a rank-one matrix is added.
<b>LDNCH</b>	Downdates the $R^T R$ Cholesky factorization of a real symmetric positive definite matrix after a rank-one matrix is removed.

### SINGULAR VALUE DECOMPOSITIONS

ROUTINE	DESCRIPTION
<b>LSVRR</b>	Computes the singular value decomposition of a real matrix.
<b>LSVCR</b>	Computes the singular value decomposition of a complex matrix.

## SINGULAR VALUE DECOMPOSITIONS

ROUTINE

DESCRIPTION

**LSGRR**

Computes the generalized inverse of a real matrix.

## CHAPTER 2: EIGENSYSTEM ANALYSIS

### EIGENVALUE DECOMPOSITION

ROUTINE

DESCRIPTION

**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 $AX = \lambda X$

ROUTINE

DESCRIPTION

**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 $AX = \lambda X$

ROUTINE

DESCRIPTION

**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 GENERAL PROBLEM $AX = \lambda X$

ROUTINE	DESCRIPTION
<b>EVLSF</b>	Computes all of the eigenvalues of a real symmetric matrix.
<b>EVCSF</b>	Computes all of the eigenvalues and eigenvectors of a real symmetric matrix.
<b>EVASF</b>	Computes the largest or smallest eigenvalues of a real symmetric matrix.
<b>EVESF</b>	Computes the largest or smallest eigenvalues and the corresponding eigenvectors of a real symmetric matrix.
<b>EVBSF</b>	Computes selected eigenvalues of a real symmetric matrix.
<b>EVFSF</b>	Computes selected eigenvalues and eigenvectors of a real symmetric matrix.
<b>EPISF</b>	Computes the performance index for a real symmetric eigensystem.

## REAL BAND SYMMETRIC MATRICIES IN BAND STORAGE MODE

ROUTINE	DESCRIPTION
<b>EVLSB</b>	Computes all of the eigenvalues of a real symmetric matrix in band symmetric storage mode.
<b>EVCSB</b>	Computes all of the eigenvalues and eigenvectors of a real symmetric matrix in band symmetric storage mode.
<b>EVASB</b>	Computes the largest or smallest eigenvalues of a real symmetric matrix in band symmetric storage mode.
<b>EVESB</b>	Computes the largest or smallest eigenvalues and the corresponding eigenvectors of a real symmetric matrix in band symmetric storage mode.
<b>EVBSB</b>	Computes the eigenvalues in a given interval of a real symmetric matrix stored in band symmetric storage mode.
<b>EVFSB</b>	Computes the eigenvalues in a given interval and the corresponding eigenvectors of a real symmetric matrix stored in band symmetric storage mode.
<b>EPISB</b>	Computes the performance index for a real symmetric eigensystem in band symmetric storage mode.

## COMPLEX HERMITIAN MATRICES

ROUTINE	DESCRIPTION
<b>EVLHF</b>	Computes all of the eigenvalues of a complex Hermitian matrix.
<b>EVCHF</b>	Computes all of the eigenvalues and eigenvectors of a complex Hermitian matrix.
<b>EVAHF</b>	Computes the largest or smallest eigenvalues of a complex Hermitian matrix.
<b>EVEHF</b>	Computes the largest or smallest eigenvalues and the corresponding eigenvectors of a complex Hermitian matrix.
<b>EVBFH</b>	Computes the eigenvalues in a given range of a complex Hermitian matrix.
<b>EVFHF</b>	Computes the eigenvalues in a given range and the corresponding eigenvectors of a complex Hermitian matrix.
<b>EPIHF</b>	Computes the performance index for a complex Hermitian eigensystem.

## REAL UPPER HESSENBERG MATRICES

ROUTINE	DESCRIPTION
<b>EVLRH</b>	Computes all of the eigenvalues of a real upper Hessenberg matrix.
<b>EVCRH</b>	Computes all of the eigenvalues and eigenvectors of a real upper Hessenberg matrix.

## COMPLEX UPPER HESSENBERG MATRICES

ROUTINE	DESCRIPTION
<b>EVLCH</b>	Computes all of the eigenvalues of a complex upper Hessenberg matrix.
<b>EVCH</b>	Computes all of the eigenvalues and eigenvectors of a complex upper.

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

### REAL GENERAL PROBLEM $AX = \lambda X$

ROUTINE	DESCRIPTION
<b>GVLRG</b>	Computes all of the eigenvalues of a generalized real eigensystem $Az = \lambda Bz$

### REAL GENERAL PROBLEM $AX = \lambda X$

ROUTINE	DESCRIPTION
<b>GVCRG</b>	Computes all of the eigenvalues and eigenvectors of a generalized real eigensystem $Az = \lambda Bz$ .
<b>GPIRG</b>	Computes the performance index for a generalized real eigensystem $Az = \lambda Bz$ .

### COMPLEX GENERAL PROBLEM $AX = \lambda BX$

ROUTINE	DESCRIPTION
<b>GVLCG</b>	Computes all of the eigenvalues of a generalized complex eigensystem $Az = \lambda Bz$ .
<b>GVCCG</b>	Computes all of the eigenvalues and eigenvectors of a generalized complex eigensystem $Az = \lambda Bz$ .
<b>GPICG</b>	Computes the performance index for a generalized complex eigensystem $Az = \lambda Bz$ .

### REAL SYMMETRIC PROBLEM $AX = \lambda BX$

ROUTINE	DESCRIPTION
<b>GVLSP</b>	Computes all of the eigenvalues of the generalized real symmetric eigenvalue problem $Az = \lambda Bz$ , with $B$ symmetric positive definite.
<b>GV CSP</b>	Computes all of the eigenvalues and eigenvectors of the generalized real symmetric eigenvalue problem $Az = \lambda Bz$ , with $B$ symmetric positive definite.
<b>GPISP</b>	Computes the performance index for a generalized real symmetric eigensystem problem.

### EIGENVALUES AND EIGENVECTORS COMPUTED WITH ARPACK

ROUTINE	DESCRIPTION
<b>ARPACK_SYMMETRIC</b>	Computes some eigenvalues and eigenvectors of the generalized real symmetric eigenvalue problem $Ax = \lambda Bx$ .
<b>ARPACK_SVD</b>	Computes some singular values and left and right singular vectors of a real rectangular matrix $A_{M \times N} = USV^T$ .
<b>ARPACK_NONSYMMETRIC</b>	Compute some eigenvalues and eigenvectors of the generalized eigenvalue problem $Ax = \lambda Bx$ . This can be used for the case $B = I$ .
<b>ARPACK_COMPLEX</b>	Compute some eigenvalues and eigenvectors of the generalized eigenvalue problem $Ax = \lambda Bx$ .

## CHAPTER 3: INTERPOLATION AND APPROXIMATION

### CURVE AND SURFACE FITTING WITH SPLINES

ROUTINE	DESCRIPTION
<b>SPLINE_CONSTRAINTS</b>	Returns the derived type array result.
<b>SPLINE_VALUES</b>	Returns an array result, given an array of input.
<b>SPLINE_FITTING</b>	Weighted least-squares fitting by <i>B</i> -splines to discrete One-Dimensional data is performed.
<b>SURFACE_CONSTRAINTS</b>	Returns the derived type array result given optional input.
<b>SURFACE_VALUES</b>	Returns a tensor product array result, given two arrays of independent variable values.
<b>SURFACE_FITTING</b>	Weighted least-squares fitting by tensor product <i>B</i> -splines to discrete two-dimensional data is performed.

### CUBIC SPLINE INTERPOLATION

ROUTINE	DESCRIPTION
<b>CSIEZ</b>	Computes the cubic spline interpolant with the 'not-a-knot' condition and returns values of the interpolant at specified points.
<b>CSINT</b>	Computes the cubic spline interpolant with the 'not-a-knot' condition.
<b>CSDEC</b>	Computes the cubic spline interpolant with specified derivative endpoint conditions.
<b>CSHER</b>	Computes the Hermite cubic spline interpolant.
<b>CSAKM</b>	Computes the Akima cubic spline interpolant.
<b>CSCON</b>	Computes a cubic spline interpolant that is consistent with the concavity of the data.
<b>CSPER</b>	Computes the cubic spline interpolant with periodic boundary conditions.

## CUBIC SPLINE EVALUATION AND INTEGRATION

ROUTINE	DESCRIPTION
<b>CSVAL</b>	Evaluates a cubic spline.
<b>CSDER</b>	Evaluates the derivative of a cubic spline.
<b>CS1GD</b>	Evaluates the derivative of a cubic spline on a grid.
<b>CSITG</b>	Evaluates the integral of a cubic spline.

## B-SPLINE INTERPOLATION

ROUTINE	DESCRIPTION
<b>SPLEZ</b>	Computes the values of a spline that either interpolates or fits user-supplied data.
<b>BSINT</b>	Computes the spline interpolant, returning the <i>B</i> -spline coefficients.
<b>BSNAK</b>	Computes the “not-a-knot” spline knot sequence.
<b>BSOPK</b>	Computes the “optimal” spline knot sequence.
<b>BS2IN</b>	Computes a two-dimensional tensor-product spline interpolant, returning the tensor-product <i>B</i> -spline coefficients.
<b>BS3IN</b>	Computes a three-dimensional tensor-product spline interpolant, returning the tensor-product <i>B</i> -spline coefficients.

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

ROUTINE	DESCRIPTION
<b>BSVAL</b>	Evaluates a spline, given its <i>B</i> -spline representation.
<b>BSDER</b>	Evaluates the derivative of a spline, given its <i>B</i> -spline representation.
<b>BS1GD</b>	Evaluates the derivative of a spline on a grid, given its <i>B</i> -spline representation.

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

ROUTINE	DESCRIPTION
<b>BSITG</b>	Evaluates the integral of a spline, given its <i>B</i> -spline representation.
<b>BS2VL</b>	Evaluates a two-dimensional tensor-product spline, given its tensor-product <i>B</i> -spline representation.
<b>BS2DR</b>	Evaluates the derivative of a two-dimensional tensor-product spline, given its tensor-product <i>B</i> -spline representation.
<b>BS2GD</b>	Evaluates the derivative of a two-dimensional tensor-product spline, given its tensor-product <i>B</i> -spline representation on a grid.
<b>BS2IG</b>	Evaluates the integral of a tensor-product spline on a rectangular domain, given its tensor-product <i>B</i> -spline representation.
<b>BS3VL</b>	Evaluates a three-dimensional tensor-product spline, given its tensor-product <i>B</i> -spline representation.
<b>BS3DR</b>	Evaluates the derivative of a three-dimensional tensor-product spline, given its tensor-product <i>B</i> -spline representation.
<b>BS3GD</b>	Evaluates the derivative of a three-dimensional tensor-product spline, given its tensor-product <i>B</i> -spline representation on a grid.
<b>BS3IG</b>	Evaluates the integral of a tensor-product spline in three dimensions over a three-dimensional rectangle, given its tensor-product <i>B</i> -spline representation.
<b>BSCPP</b>	Converts a spline in <i>B</i> -spline representation to piecewise polynomial representation.

## PIECEWISE POLYNOMIAL

ROUTINE	DESCRIPTION
<b>PPVAL</b>	Evaluates a piecewise polynomial.
<b>PPDER</b>	Evaluates the derivative of a piecewise polynomial.
<b>PP1GD</b>	Evaluates the derivative of a piecewise polynomial on a grid.
<b>PPITG</b>	Evaluates the integral of a piecewise polynomial.

## QUADRATIC POLYNOMIAL INTERPOLATION ROUTINES FOR GRIDDED DATA

ROUTINE	DESCRIPTION
<b>QDVAL</b>	Evaluates a function defined on a set of points using quadratic interpolation.
<b>QDDER</b>	Evaluates the derivative of a function defined on a set of points using quadratic interpolation.
<b>QD2VL</b>	Evaluates a function defined on a rectangular grid using quadratic interpolation.
<b>QD2DR</b>	Evaluates the derivative of a function defined on a rectangular grid using quadratic interpolation.
<b>QD3VL</b>	Evaluates a function defined on a rectangular three-dimensional grid using quadratic interpolation.
<b>DQ3DR</b>	Evaluates the derivative of a function defined on a rectangular three-dimensional grid using quadratic interpolation.

## MULTI-DIMENSIONAL INTERPOLATION

ROUTINE	DESCRIPTION
<b>SURF</b>	Computes a smooth bivariate interpolant to scattered data that is locally a quintic polynomial in two variables.
<b>SURFND</b>	Performs multidimensional interpolation and differentiation for up to 7 dimensions.

## LEAST-SQUARES APPROXIMATION

ROUTINE	DESCRIPTION
<b>RLINE</b>	Fits a line to a set of data points using least squares.
<b>RCURV</b>	Fits a polynomial curve using least squares.
<b>FNLSQ</b>	Computes a least-squares approximation with user-supplied basis functions.
<b>BLSQ</b>	Computes the least-squares spline approximation, and returns the B-spline coefficients.
<b>BSVLS</b>	Computes the variable knot B-spline least squares approximation to given data.

## LEAST-SQUARES APPROXIMATION

ROUTINE	DESCRIPTION
<b>CONFIT</b>	Computes the least-squares constrained spline approximation, returning the B-spline coefficients.
<b>BSLS2</b>	Computes a two-dimensional tensor-product spline approximant using least squares, returning the tensor-product B-spline coefficients.
<b>BSLS3</b>	Computes a three-dimensional tensor-product spline approximant using least squares, returning the tensor-product B-spline coefficients.

## CUBIC SPLINE SMOOTHING

ROUTINE	DESCRIPTION
<b>CSSD</b>	Smooths one-dimensional data by error detection.
<b>CSSMH</b>	Computes a smooth cubic spline approximation to noisy data.
<b>CSSCV</b>	Computes a smooth cubic spline approximation to noisy data using cross-validation to estimate the smoothing parameter.

## RATIONAL $L_\infty$ APPROXIMATION

ROUTINE	DESCRIPTION
<b>RATCH</b>	Computes a rational weighted Chebyshev approximation to a continuous function on an interval.

# CHAPTER 4: INTEGRATION AND DIFFERENTIATION

## UNIVARIATE QUADRATURE

ROUTINE	DESCRIPTION
<b>QDAGS</b>	Integrates a function (which may have endpoint singularities).
<b>QDAG</b>	Integrates a function using a globally adaptive scheme based on Gauss-Kronrod rules.
<b>QDAGP</b>	Integrates a function with singularity points given.
<b>QDAG1D</b>	Integrates a function with a possible internal or endpoint singularity.



## UNIVARIATE QUADRATURE

ROUTINE	DESCRIPTION
<b>QDAGI</b>	Integrates a function over an infinite or semi-infinite interval.
<b>QDAWO</b>	Integrates a function containing a sine or a cosine.
<b>QDAWF</b>	Computes a Fourier integral.
<b>QDAWS</b>	Integrates a function with algebraic logarithmic singularities.
<b>QDAWC</b>	Integrates a function $F(X)/(X - C)$ in the Cauchy principal value sense.
<b>QDNG</b>	Integrates a smooth function using a nonadaptive rule.

## MULTIDIMENSIONAL QUADRATURE

ROUTINE	DESCRIPTION
<b>TWODQ</b>	Computes a two-dimensional iterated integral.
<b>QDAG2D</b>	Integrates a function of two variables with a possible internal or end point singularity.
<b>QDAG3D</b>	Integrates a function of three variables with a possible internal or endpoint singularity.
<b>QAND</b>	Integrates a function on a hyper-rectangle.
<b>QMC</b>	Integrates a function over a hyper-rectangle using a quasi-Monte Carlo method.

## GAUSS RULES AND THREE-TERM RECURRENCES

ROUTINE	DESCRIPTION
<b>GQRUL</b>	Computes a Gauss, Gauss-Radau, or Gauss-Lobatto quadrature rule with various classical weight functions.
<b>GQRFC</b>	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.

## GAUSS RULES AND THREE-TERM RECURRENCES

ROUTINE	DESCRIPTION
<b>RECCF</b>	Computes recurrence coefficients for various monic polynomials.
<b>RECQR</b>	Computes recurrence coefficients for monic polynomials given a quadrature rule.
<b>FQRUL</b>	Computes a Fejér quadrature rule with various classical weight functions.

## DIFFERENTIATION

ROUTINE	DESCRIPTION
<b>DERIV</b>	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

ROUTINE	DESCRIPTION
<b>IVPRK</b>	Solves an initial-value problem for ordinary differential equations using the Runge-Kutta-Verner fifth-order and sixth-order method.
<b>IVMRK</b>	Solves an initial-value problem $y' = f(t, y)$ for ordinary differential equations using Runge-Kutta pairs of various orders.
<b>IVPAG</b>	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

ROUTINE	DESCRIPTION
<b>BVPFD</b>	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.
<b>BVPMS</b>	Solves a (parameterized) system of differential equations with boundary conditions at two points, using a multiple-shooting method.

## SOLUTION OF DIFFERENTIAL-ALGEBRAIC SYSTEMS

ROUTINE	DESCRIPTION
<b>DAESL</b>	Solves a first order differential-algebraic system of equations, $g(t, y, y') = 0$ , with optional additional constraints and user-defined linear system solver.

## FIRST-AND-SECOND-ORDER ORDINARY DIFFERENTIAL EQUATIONS

### SOLUTION OF THE INITIAL-VALUE PROBLEM FOR ODES

ROUTINE	DESCRIPTION
<b>IVOAM</b>	Solves an initial-value problem for a system of ordinary differential equations of order one or two using a variable order Adams method.

## PARTIAL DIFFERENTIAL EQUATIONS

### SOLUTION OF SYSTEMS OF PDES IN ONE DIMENSION

ROUTINE	DESCRIPTION
<b>PDE_1D_MG</b>	Method of lines with Variable Griddings.
<b>MMOLCH</b>	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.
<b>FEYNMAN_KAC</b>	Solves the generalized Feynman-Kac PDE on a rectangular grid using a finite element Galerkin method.
<b>HQSVAL</b>	This rank-1 array function evaluates a Hermite quintic spline or one of its derivatives for an array of input points.

### SOLUTION OF A PDE IN TWO AND THREE DIMENSIONS

ROUTINE	DESCRIPTION
<b>FPS2H</b>	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.
<b>FPS3H</b>	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

ROUTINE	DESCRIPTION
<b>SLEIG</b>	Determines eigenvalues, eigenfunctions and/or spectral density functions for Sturm-Liouville problems.
<b>SLCNT</b>	Calculates the indices of eigenvalues of a Sturm-Liouville problem.

## CHAPTER 6: TRANSFORMS

### REAL TRIGONOMETRIC FFT

ROUTINE	DESCRIPTION
<b>FAST_DFT</b>	Computes the Discrete Fourier Transform of a rank-1 complex array, <b>x</b> .
<b>FAST_2DFT</b>	Computes the Discrete Fourier Transform (2DFT) of a rank-2 complex array, <b>x</b> .
<b>FAST_3DFT</b>	Computes the Discrete Fourier Transform (2DFT) of a rank-3 complex array, <b>x</b> .
<b>FFTRF</b>	Computes the Fourier coefficients of a real periodic sequence.
<b>FFTRB</b>	Computes the real periodic sequence from its Fourier coefficients.
<b>FFTRI</b>	Computes parameters needed by <b>FFTRF</b> and <b>FFTRB</b> .

### COMPLEX EXPONENTIAL FFT

ROUTINE	DESCRIPTION
<b>FFTCF</b>	Computes the Fourier coefficients of a complex periodic sequence.
<b>FFTCB</b>	Computes the complex periodic sequence from its Fourier coefficients.
<b>FFTCI</b>	Computes parameters needed by <b>FFTCF</b> and <b>FFTCB</b> .

### REAL SINE AND COSINE FFTS

ROUTINE	DESCRIPTION
<b>FSINT</b>	Computes the discrete Fourier sine transformation of an odd sequence.
<b>FSINI</b>	Computes parameters needed by <b>FSINT</b> .
<b>FCOST</b>	Computes the discrete Fourier cosine transformation of an even sequence.

## REAL SINE AND COSINE FFTS

ROUTINE	DESCRIPTION
<b>FCOSI</b>	Computes parameters needed by <b>FCOST</b> .

## REAL QUARTER SINE AND QUARTER COSINE FFTS

ROUTINE	DESCRIPTION
<b>QSINF</b>	Computes the coefficients of the sine Fourier transform with only odd wave numbers.
<b>QSINB</b>	Computes a sequence from its sine Fourier coefficients with only odd wave numbers.
<b>QSINI</b>	Computes parameters needed by <b>QSINF</b> and <b>QSINB</b> .
<b>QCOSF</b>	Computes the coefficients of the cosine Fourier transform with only odd wave numbers.
<b>QCOSB</b>	Computes a sequence from its cosine Fourier coefficients with only odd wave numbers.
<b>QCOSI</b>	Computes parameters needed by <b>QCOSF</b> and <b>QCOSB</b> .

## TWO AND THREE DIMENSIONAL COMPLEX FFTS

ROUTINE	DESCRIPTION
<b>FFT2D</b>	Computes Fourier coefficients of a complex periodic two-dimensional array.
<b>FFT2B</b>	Computes the inverse Fourier transform of a complex periodic two dimensional array.
<b>FFT3F</b>	Computes Fourier coefficients of a complex periodic three-dimensional array.
<b>FFT3B</b>	Computes the inverse Fourier transform of a complex periodic three-dimensional array.

## CONVOLUTIONS AND CORRELATIONS

ROUTINE	DESCRIPTION
<b>RCONV</b>	Computes the convolution of two real vectors.
<b>CCONV</b>	Computes the convolution of two complex vectors.
<b>RCORL</b>	Computes the correlation of two real vectors.
<b>CCORL</b>	Computes the correlation of two complex vectors.

## LAPLACE TRANSFORM

ROUTINE	DESCRIPTION
<b>INLAP</b>	Computes the inverse Laplace transform of a complex function.
<b>SINLP</b>	Computes the inverse Laplace transform of a complex function.

## CHAPTER 7: NONLINEAR EQUATIONS

### ZEROS OF A POLYNOMIAL

ROUTINE	DESCRIPTION
<b>ZPLRC</b>	Finds the zeros of a polynomial with real coefficients using Laguerre's method.
<b>ZPORC</b>	Finds the zeros of a polynomial with real coefficients using the Jenkins-Traub three-stage algorithm.
<b>ZPOCC</b>	Finds the zeros of a polynomial with complex coefficients using the Jenkins-Traub three-stage algorithm.

### ZEROS OF A FUNCTION

ROUTINE	DESCRIPTION
<b>ZANLY</b>	Finds the zeros of a univariate complex function using Müller's method.
<b>ZUNI</b>	Finds a zero of a real univariate function.

## ZEROS OF A FUNCTION

ROUTINE	DESCRIPTION
<b>ZBREN</b>	Finds a zero of a real function that changes sign in a given interval.
<b>ZREAL</b>	Finds the real zeros of a real function using Müller's method.

## ROOT OF A SYSTEM OF EQUATIONS

ROUTINE	DESCRIPTION
<b>NEQNF</b>	Solves a system of nonlinear equations using a modified Powell hybrid algorithm and a finite-difference approximation to the Jacobian.
<b>NEQNJ</b>	Solves a system of nonlinear equations using a modified Powell hybrid algorithm with a user-supplied Jacobian.
<b>NEQBF</b>	Solves a system of nonlinear equations using factored secant update with a finite-difference approximation to the Jacobian.
<b>NEQBJ</b>	Solves a system of nonlinear equations using factored secant update with a user-supplied Jacobian.

## CHAPTER 8: OPTIMIZATION

### UNCONSTRAINED MINIMIZATION

#### UNIVARIATE FUNCTION

ROUTINE	DESCRIPTION
<b>UVMIF</b>	Finds the minimum point of a smooth function of a single variable using only function evaluations.
<b>UVMID</b>	Finds the minimum point of a smooth function of a single variable using both function evaluations and first derivative evaluations.
<b>UVMGS</b>	Finds the minimum point of a non-smooth function of a single variable.

#### MULTIVARIATE FUNCTION

ROUTINE	DESCRIPTION
<b>UMINF</b>	Minimizes a function of <b>N</b> variables using a quasi-Newton method and a finite-difference gradient.
<b>UMING</b>	Minimizes a function of <b>N</b> variables using a quasi-Newton method and a user-supplied gradient.

**MULTIVARIATE FUNCTION**

ROUTINE	DESCRIPTION
<b>UMIDH</b>	Minimizes a function of <b>N</b> variables using a modified Newton method and a finite-difference Hessian.
<b>UMIAH</b>	Minimizes a function of <b>N</b> variables using a modified Newton method and a user-supplied Hessian.
<b>UMCGF</b>	Minimizes a function of <b>N</b> variables using a conjugate gradient algorithm and a finite-difference gradient.
<b>UMCGG</b>	Minimizes a function of <b>N</b> variables using a conjugate gradient algorithm and a user-supplied gradient.
<b>UMPOL</b>	Minimizes a function of <b>N</b> variables using a direct search polytope algorithm.

**NONLINEAR LEAST SQUARES**

ROUTINE	DESCRIPTION
<b>UNLSF</b>	Solves a nonlinear least-squares problem using a modified Levenberg-Marquardt algorithm and a finite-difference Jacobian.
<b>UNLSJ</b>	Solves a nonlinear least squares problem using a modified Levenberg-Marquardt algorithm and a user-supplied Jacobian.

**MINIMIZATION WITH SIMPLE BOUNDS**

ROUTINE	DESCRIPTION
<b>BCONF</b>	Minimizes a function of <b>N</b> variables subject to bounds on the variables using a quasi-Newton method and a finite-difference gradient.
<b>BCONG</b>	Minimizes a function of <b>N</b> variables subject to bounds on the variables using a quasi-Newton method and a user-supplied gradient.
<b>BCODH</b>	Minimizes a function of <b>N</b> variables subject to bounds on the variables using a modified Newton method and a finite-difference Hessian.
<b>BCOAH</b>	Minimizes a function of <b>N</b> variables subject to bounds on the variables using a modified Newton method and a user-supplied Hessian.
<b>BCPOL</b>	Minimizes a function of <b>N</b> variables subject to bounds on the variables using a direct search complex algorithm.
<b>BCLSF</b>	Solves a nonlinear least squares problem subject to bounds on the variables using a modified Levenberg-Marquardt algorithm and a finite-difference Jacobian.



## MINIMIZATION WITH SIMPLE BOUNDS

ROUTINE	DESCRIPTION
<b>BCLSJ</b>	Solves a nonlinear least squares problem subject to bounds on the variables using a modified Levenberg-Marquardt algorithm and a user-supplied Jacobian.
<b>BCNLS</b>	Solves a nonlinear least-squares problem subject to bounds on the variables and general linear constraints.

## LINEARLY CONSTRAINED MINIMIZATION

ROUTINE	DESCRIPTION
<b>READ_MPS</b>	Reads an <b>MPS</b> file containing a linear programming problem or a quadratic programming problem.
<b>MPS_FREE</b>	Deallocates the space allocated for the IMSL derived type <b>s_MPS</b> . This routine is usually used in conjunction with <b>READ_MPS</b> .
<b>DENSE_LP</b>	Solves a linear programming problem using an active set strategy.
<b>DLPRS</b>	Solves a linear programming problem via the revised simplex algorithm.
<b>SLPRS</b>	Solves a sparse linear programming problem via the revised simplex algorithm.
<b>TRAN</b>	Solves a transportation problem.
<b>QPROG</b>	Solves a quadratic programming problem subject to linear equality/inequality constraints.
<b>LCONF</b>	Minimizes a general objective function subject to linear equality/inequality constraints.
<b>LCONG</b>	Minimizes a general objective function subject to linear equality/inequality constraints and a user-supplied gradient.

## NONLINEARLY CONSTRAINED MINIMIZATION

ROUTINE	DESCRIPTION
<b>NNLPF</b>	Nonlinearly Constrained Minimization using a sequential equality constrained QP method.
<b>NNLPG</b>	Nonlinearly Constrained Minimization using a sequential equality constrained QP method and a user-supplied gradient.

## SERVICE ROUTINES

ROUTINE	DESCRIPTION
<b>CDGRD</b>	Approximates the gradient using central differences.
<b>FDGRD</b>	Approximates the gradient using forward differences.
<b>FDHES</b>	Approximates the Hessian using forward differences and function values.
<b>GDHES</b>	Approximates the Hessian using forward differences and a user-supplied gradient.
<b>DDJAC</b>	Approximates the Jacobian of <b>M</b> functions in <b>N</b> unknowns using divided differences.
<b>FDJAC</b>	Approximate the Jacobian of <b>M</b> functions in <b>N</b> unknowns using forward differences.
<b>CHGRD</b>	Checks a user-supplied gradient of a function.
<b>CHHES</b>	Checks a user-supplied Hessian of an analytic function.
<b>CHJAC</b>	Checks a user-supplied Jacobian of a system of equations with <b>M</b> functions in <b>N</b> unknowns.
<b>GGUES</b>	Generates points in an <b>N</b> -dimensional space.

## CHAPTER 9: BASIC MATRIX/VECTOR OPERATIONS

### BASIC LINEAR ALGEBRA SUBPROGRAMS (BLAS)

#### LEVEL 1 BLAS

ROUTINE	DESCRIPTION
<b>SSET</b>	Sets the components of a vector to a scalar.
<b>SCOPY</b>	Copies a vector <b>x</b> to a vector <b>y</b> , both single precision.
<b>SSCAL</b>	Multiplies a vector by a scalar, $y \leftarrow \alpha y$ , both single precision.

LEVEL 1 BLAS

ROUTINE	DESCRIPTION
<b>SVCAL</b>	Multiplies a vector by a scalar and stores the result in another vector, $y \leftarrow \alpha x$ , all single precision.
<b>SADD</b>	Adds a scalar to each component of a vector, $x \leftarrow x + a$ , all single precision.
<b>SSUB</b>	Subtract each component of a vector from a scalar, $x \leftarrow a - x$ , all single precision.
<b>SAXPY</b>	Computes the scalar times a vector plus a vector, $y \leftarrow \alpha x + y$ , all single precision.
<b>SSWAP</b>	Interchange vectors $x$ and $y$ , both single precision.
<b>SDOT</b>	Computes the single-precision dot product $x^T y$ .
<b>DSDOT</b>	Computes the single-precision dot product $x^T y$ using a double precision accumulator.
<b>SDSDOT</b>	Computes the sum of a single-precision scalar and a single precision dot product, $a + x^T y$ , using a double-precision accumulator.
<b>SDDOTI</b>	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 $ACC \leftarrow a + x^T y$ .
<b>SHPROD</b>	Computes the Hadamard product of two single-precision vectors.
<b>SXYZ</b>	Computes a single-precision $xyz$ product.
<b>SSUM</b>	Sums the values of a single-precision vector.
<b>SASUM</b>	Sums the absolute values of the components of a single-precision vector.
<b>SNRM2</b>	Computes the Euclidean length or $L_2$ norm of a single-precision vector.
<b>SPRDCT</b>	Multiplies the components of a single-precision vector.

## LEVEL 1 BLAS

ROUTINE	DESCRIPTION
<b>ISMIN</b>	Finds the smallest index of the component of a single-precision vector having minimum value.
<b>ISMAX</b>	Finds the smallest index of the component of a single-precision vector having maximum value.
<b>ISAMIN</b>	Finds the smallest index of the component of a single-precision vector having minimum absolute value.
<b>ISAMAX</b>	Finds the smallest index of the component of a single-precision vector having maximum absolute value.
<b>SROTG</b>	Constructs a Givens plane rotation in single precision.
<b>SROT</b>	Applies a Givens plane rotation in single precision.
<b>SROTM</b>	Applies a modified Givens plane rotation in single precision.
<b>SROTMG</b>	Constructs a modified Givens plane rotation in single precision.

## LEVEL 2 BLAS

ROUTINE	DESCRIPTION
<b>SGEMV</b>	Computes one of the matrix-vector operations: $y \leftarrow \alpha Ax + \beta y$ , or $y \leftarrow \alpha A^T x + \beta y$ .
<b>SGBMV</b>	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.
<b>CHEMV</b>	Compute the matrix-vector operation $y \leftarrow \alpha Ax + \beta y$ where $A$ is a Hermitian matrix.
<b>CHPMV</b>	Compute the matrix-vector operation $y \leftarrow \alpha Ax + \beta y$ where $A$ is a packed Hermitian matrix.
<b>CHBMV</b>	Computes the matrix-vector operation $y \leftarrow \alpha Ax + \beta y$ where $A$ is a Hermitian band matrix in band Hermitian storage.
<b>CTPMV</b>	Performs the matrix-vector operation $x \leftarrow \bar{A}^T x$ in packed form.
<b>CTPSV</b>	Solves the systems of equations $x \leftarrow (\bar{A}^T)^{-1} x \equiv (A^H)^{-1} x$ in packed form.

LEVEL 2 BLAS

ROUTINE	DESCRIPTION
<b>SSYMV</b>	Computes the matrix-vector operation $y \leftarrow \alpha Ax + \beta y$ where $A$ is a symmetric matrix.
<b>SSBMV</b>	Computes the matrix-vector operation $y \leftarrow \alpha Ax + \beta y$ where $A$ is a symmetric matrix in band symmetric storage mode.
<b>SSPMV</b>	Performs the matrix-vector operation $y \leftarrow \alpha Ax + \beta y$ in packed form.
<b>STRMV</b>	Computes one of the matrix-vector operations: $x \leftarrow Ax$ or $x \leftarrow A^T x$ where $A$ is a triangular matrix.
<b>STBMV</b>	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.
<b>STRSV</b>	Solves one of the triangular linear systems: $x \leftarrow A^{-1}x$ or $x \leftarrow (A^{-1})^T x$ where $A$ is a triangular matrix.
<b>STBSV</b>	Solves one of the triangular systems: $x \leftarrow A^{-1}x$ or $x \leftarrow (A^{-1})^T x$ where $A$ is a triangular matrix in band storage mode.
<b>STPMV</b>	Performs one of the matrix-vector operations: $x \leftarrow Ax$ or $x \leftarrow A^T x$ where $A$ is in packed form.
<b>STPSV</b>	Solves one of the systems of equations $x \leftarrow A^{-1}x$ or $x \leftarrow (A^{-1})^T x A^T x$ where $A$ is in packed form.
<b>SGER</b>	Computes the rank-one update of a real general matrix: $A \leftarrow A + \alpha xy^T$ .
<b>CGERU</b>	Computes the rank-one update of a complex general matrix: $A \leftarrow A + \alpha xy^T$ .
<b>CGERC</b>	Computes the rank-one update of a complex general matrix: $A \leftarrow A + \alpha x \bar{y}^T$ .
<b>CHER</b>	Computes the rank-one update of a Hermitian matrix: $A \leftarrow A + \alpha x \bar{x}^T$ with $x$ complex and $\alpha$ real.
<b>CHPR</b>	Computes the rank-one update of a Hermitian matrix: $A \leftarrow A + \alpha x \bar{x}^T$ in packed form with $x$ complex and $\alpha$ real.
<b>CHER2</b>	Computes a rank-two update of a Hermitian matrix: $A \leftarrow A + \alpha x \bar{y}^T + \bar{\alpha} y \bar{x}^T$ .
<b>CHPR2</b>	Performs the hermitian rank 2 operation $A \leftarrow A + \alpha x \bar{y}^T + \bar{\alpha} y \bar{x}^T$ in packed form.

## LEVEL 2 BLAS

ROUTINE	DESCRIPTION
<b>SSYR</b>	Computes the rank-one update of a real symmetric matrix: $A \leftarrow A + \alpha x x^T$ .
<b>SSPR</b>	Performs the symmetric rank 1 operation $A \leftarrow A + \alpha x x^T$ in packed form.
<b>SSYR2</b>	Computes the rank-two update of a real symmetric matrix: $A \leftarrow A + \alpha x y + \alpha y x^T$ .
<b>SSPR2</b>	Performs the symmetric rank 2 operation $A \leftarrow A + \alpha x y + \alpha y x^T$ in packed form.

## LEVEL 3 BLAS

ROUTINE	DESCRIPTION
<b>SGEMM</b>	Computes one of the matrix-matrix operations: $C \leftarrow \alpha A B + \beta C$ , $C \leftarrow \alpha A^T B + \beta C$ , $C \leftarrow \alpha A B^T + \beta C$ , or $C \leftarrow \alpha A^T B^T + \beta C$ .
<b>SSYMM</b>	Computes one of the matrix-matrix operations: $C \leftarrow \alpha A B + \beta C$ or $C \leftarrow \alpha B A + \beta C$ , where $A$ is a symmetric matrix and $B$ and $C$ are $m$ by $n$ matrices.
<b>CHEMM</b>	Computes one of the matrix-matrix operations: $C \leftarrow \alpha A B + \beta C$ or $C \leftarrow \alpha B A + \beta C$ , where $A$ is a Hermitian matrix and $B$ and $C$ are $m$ by $n$ matrices.
<b>SSYRK</b>	Computes one of the symmetric rank $k$ operations: $C \leftarrow \alpha A A^T + \beta C$ 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.
<b>CHERK</b>	Computes one of the Hermitian rank $k$ operations: $C \leftarrow \alpha A \bar{A}^T + \beta C$ or $C \leftarrow \alpha \bar{A}^T A + \beta C$ , 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.
<b>SSYR2K</b>	Computes one of the symmetric rank $2k$ operations: $C \leftarrow \alpha A B^T + \alpha B A^T + \beta C$ 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.
<b>CHER2K</b>	Computes one of the Hermitian rank $2k$ operations: $C \leftarrow \alpha \bar{A} \bar{B}^T + \alpha \bar{B} \bar{A}^T + \beta C$ or $C \leftarrow \alpha \bar{A}^T \bar{B} + \alpha \bar{B}^T \bar{A} + \beta C$ where $C$ is an $n$ by $n$ Hermitian matrix in the first case and $k$ by $n$ matrices in the second case.

**LEVEL 3 BLAS**

ROUTINE	DESCRIPTION
<b>STRMM</b>	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.
<b>STRSM</b>	Solves one of the matrix equations: $B \leftarrow \alpha A^{-1} B$ , $B \leftarrow \alpha BA^{-1}$ or $B \leftarrow \alpha (A^{-1})^T B$ , $B \leftarrow \alpha B(A^{-1})^T$ , where $B$ is an $m$ by $n$ matrix and $A$ is a triangular matrix.
<b>CTRSM</b>	Solves one of the complex matrix equations: $B \leftarrow \alpha (\bar{A}^T)^{-1} B$ or $B \leftarrow \alpha B (\bar{A}^T)^{-1}$ , where $A$ is a triangular matrix.

**BLAS FOR NVIDIA**

ROUTINE	DESCRIPTION
<b>CUBLAS_GET</b>	Returns the switchover value for a positional array argument for a specified BLAS routine.
<b>CUBLAS_SET</b>	Sets the switchover value for an array used by a specified BLAS routine.
<b>CHECK_BUFFER_ALLOCATION</b>	Maintains buffer sizes on the NVIDIA device and performs one-time initialization.
<b>CUDA_ERROR_PRINT</b>	Prints error messages generated through the use of the CUDABLAS Library using the IMSL error handler.

**OTHER MATRIX/VECTOR OPERATIONS****MATRIX COPY**

ROUTINE	DESCRIPTION
<b>CRGRG</b>	Copies a real general matrix.
<b>CCGCG</b>	Copies a complex general matrix.
<b>CRBRB</b>	Copies a real band matrix stored in band storage mode.
<b>CCBCB</b>	Copies a complex band matrix stored in complex band storage mode.

## MATRIX CONVERSION

ROUTINE	DESCRIPTION
<b>CRGRB</b>	Converts a real general matrix to a matrix in band storage mode.
<b>CRBRG</b>	Converts a real matrix in band storage mode to a real general matrix.
<b>CCGCB</b>	Converts a complex general matrix to a matrix in complex band storage mode.
<b>CCBCG</b>	Converts a complex matrix in band storage mode to a complex matrix in full storage mode.
<b>CRGCG</b>	Copies a real general matrix to a complex general matrix.
<b>CRRCR</b>	Copies a real rectangular matrix to a complex rectangular matrix.
<b>CRBCB</b>	Converts a real matrix in band storage mode to a complex matrix in band storage mode.
<b>CSFRG</b>	Extends a real symmetric matrix defined in its upper triangle to its lower triangle.
<b>CHFCG</b>	Extends a complex Hermitian matrix defined in its upper triangle to its lower triangle.
<b>CSBRB</b>	Copies a real symmetric band matrix stored in band symmetric storage mode to a real band matrix stored in band storage mode.
<b>CHBCB</b>	Copies a complex Hermitian band matrix stored in band Hermitian storage mode to a complex band matrix stored in band storage mode.
<b>TRNRR</b>	Transposes a rectangular matrix.

## MATRIX MULTIPLICATION

ROUTINE	DESCRIPTION
<b>MXTXF</b>	Computes the transpose product of a matrix, $A^T A$ .
<b>MXTYF</b>	Multiplies the transpose of matrix $A$ by matrix $B$ , $A^T B$ .
<b>MXYTF</b>	Multiplies a matrix $A$ by the transpose of a matrix $B$ , $AB^T$ .



**MATRIX MULTIPLICATION**

ROUTINE	DESCRIPTION
<b>MRRRR</b>	Multiplies two real rectangular matrices, $AB$ .
<b>MCRCR</b>	Multiplies two complex rectangular matrices, $AB$ .
<b>HRRRR</b>	Computes the Hadamard product of two real rectangular matrices.
<b>BLINF</b>	Computes the bilinear form $x^T Ay$ .
<b>POLRG</b>	Evaluates a real general matrix polynomial.

**MATRIX-VECTOR MULTIPLICATION**

ROUTINE	DESCRIPTION
<b>MURRV</b>	Multiplies a real rectangular matrix by a vector.
<b>MURBV</b>	Multiplies a real band matrix in band storage mode by a real vector.
<b>MUCRV</b>	Multiplies a complex rectangular matrix by a complex vector.
<b>MUCBV</b>	Multiplies a complex band matrix in band storage mode by a complex vector.

**MATRIX ADDITION**

ROUTINE	DESCRIPTION
<b>ARBRB</b>	Adds two band matrices, both in band storage mode.
<b>ACBCB</b>	Adds two complex band matrices, both in band storage mode.

**MATRIX NORM**

ROUTINE	DESCRIPTION
<b>NRIRR</b>	Computes the infinity norm of a real matrix.
<b>NR1RR</b>	Computes the 1-norm of a real matrix.
<b>NR2RR</b>	Computes the Frobenius norm of a real rectangular matrix.
<b>NR1RB</b>	Computes the 1-norm of a real band matrix in band storage mode.
<b>NR1CB</b>	Computes the 1-norm of a complex band matrix in band storage mode.

**DISTANCE BETWEEN TWO POINTS**

ROUTINE	DESCRIPTION
<b>DISL2</b>	Computes the Euclidean (2-norm) distance between two points.
<b>DISL1</b>	Computes the 1-norm distance between two points.
<b>DISLI</b>	Computes the infinity norm distance between two points.

**VECTOR CONVOLUTIONS**

ROUTINE	DESCRIPTION
<b>VCONR</b>	Computes the convolution of two real vectors.
<b>VCONC</b>	Computes the convolution of two complex vectors.

**EXTENDED PRECISION ARITHMETIC**

ROUTINE	DESCRIPTION
<b>DQINI</b>	Initializes an extended-precision accumulator with a double-precision scalar.
<b>DQSTO</b>	Stores a double-precision approximation to an extended-precision scalar.

## EXTENDED PRECISION ARITHMETIC

ROUTINE	DESCRIPTION
<b>DQADD</b>	Adds a double-precision scalar to the accumulator in extended precision.
<b>DQMUL</b>	Multiplies double-precision scalars in extended precision.
<b>ZQINI</b>	Initializes an extended-precision complex accumulator to a double complex scalar.
<b>ZQSTO</b>	Stores a double complex approximation to an extended-precision complex scalar.
<b>ZQADD</b>	Adds a double complex scalar to the accumulator in extended precision.
<b>ZQMUL</b>	Multiplies double complex scalars using extended precision.

## CHAPTER 10: LINEAR ALGEBRA OPERATORS AND GENERIC FUNCTIONS

### OPERATORS

ROUTINE	DESCRIPTION
<b>OPERATORS: .x., .tx., .xt., .xh.</b>	Computes matrix-vector and matrix-matrix products.
<b>OPERATORS: .t., .h.</b>	Computes transpose and conjugate transpose of a matrix.
<b>OPERATORS: .i.</b>	Computes the inverse matrix, for square non-singular matrices, or the Moore-Penrose generalized inverse matrix for singular square matrices or rectangular matrices.
<b>OPERATORS: .ix., .xi.</b>	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.

### FUNCTIONS

ROUTINE	DESCRIPTION
<b>CHOL</b>	Computes the Cholesky factorization of a positive-definite, symmetric or self-adjoint matrix, <b>A</b> .
<b>COND</b>	Computes the condition number of a matrix, <b>A</b> .

## FUNCTIONS

ROUTINE	DESCRIPTION
<b>DET</b>	Computes the determinant of a rectangular matrix, <b>A</b> .
<b>DIAG</b>	Constructs a square diagonal matrix from a rank-1 array or several diagonal matrices from a rank-2 array.
<b>DIAGONALS</b>	Extracts a rank-1 array whose values are the diagonal terms of a rank-2 array argument.
<b>EIG</b>	Computes the eigenvalue-eigenvector decomposition of an ordinary or generalized eigenvalue problem.
<b>EYE</b>	Creates a rank-2 square array whose diagonals are all the value one.
<b>FFT</b>	The Discrete Fourier Transform of a complex sequence and its inverse transform.
<b>FFT_BOX</b>	The Discrete Fourier Transform of several complex or real sequences.
<b>IFFT</b>	The inverse of the Discrete Fourier Transform of a complex sequence.
<b>IFFT_BOX</b>	The inverse Discrete Fourier Transform of several complex or real sequences.
<b>ISNAN</b>	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.
<b>NAN</b>	Returns, as a scalar function, a value corresponding to the IEEE 754 Standard format of floating point (ANSI/IEEE 1985) for NaN.
<b>NORM</b>	Computes the norm of a rank-1 or rank-2 array.
<b>ORTH</b>	Orthogonalizes the columns of a rank-2 or rank-3 array.
<b>RAND</b>	Computes a scalar, rank-1, rank-2 or rank-3 array of random numbers.
<b>RANK</b>	Computes the mathematical rank of a rank-2 or rank-3 array.

## FUNCTIONS

ROUTINE	DESCRIPTION
<b>SVD</b>	Computes the singular value decomposition of a rank-2 or rank-3 array, $A = USV^T$ .
<b>UNIT</b>	Normalizes the columns of a rank-2 or rank-3 array so each has Euclidean length of value one.

## CHAPTER 11: UTILITIES

### SCALAPACK UTILITIES

ROUTINE	DESCRIPTION
<b>SCALAPACK_SETUP</b>	This routine sets up a processor grid and calculates default values for various entities to be used in mapping a global array to the processor grid.
<b>SCALAPACK_GETDIM</b>	This routine calculates the row and column dimensions of a local distributed array based on the size of the array to be distributed and the row and column blocking factors to be used.
<b>SCALAPACK_READ</b>	Reads matrix data from a file and transmits it into the two-dimensional block-cyclic form.
<b>SCALAPACK_WRITE</b>	Writes the matrix data to a file.
<b>SCALAPACK_MAP</b>	This routine maps array data from a global array to local arrays in the two-dimensional block-cyclic form required by <b>ScaLAPACK</b> routines.
<b>SCALAPACK_UNMAP</b>	This routine unmaps array data from local distributed arrays to a global array. The data in the local arrays must have been stored in the two-dimensional block-cyclic form required by <b>ScaLAPACK</b> routines.
<b>SCALAPACK_EXIT</b>	This routine exits <b>ScaLAPACK</b> mode for the IMSL Library routines. All processors in the <b>BLACS</b> context call the routine.

### PRINT

ROUTINE	DESCRIPTION
<b>ERROR_POST</b>	Prints error messages.
<b>SHOW</b>	Prints rank-1 or rank-2 arrays of numbers in a readable format.
<b>WRRRN</b>	Prints a real rectangular matrix with integer row and column labels.

## PRINT

ROUTINE	DESCRIPTION
<b>WRRRL</b>	Prints a real rectangular matrix with a given format and labels.
<b>WRIRN</b>	Prints an integer rectangular matrix with integer row and column labels.
<b>WRIRL</b>	Prints an integer rectangular matrix with a given format and labels.
<b>WRCRN</b>	Prints a complex rectangular matrix with integer row and column labels.
<b>WRCRL</b>	Prints a complex rectangular matrix with a given format and labels.
<b>WROPT</b>	Sets or Retrieves an option for printing a matrix.
<b>PGOPT</b>	Sets or Retrieves page width and length for printing.

## PERMUTE

ROUTINE	DESCRIPTION
<b>PERMU</b>	Rearranges the elements of an array as specified by a permutation.
<b>PERMA</b>	Permutes the rows or columns of a matrix.

## SORT

ROUTINE	DESCRIPTION
<b>SORT_REAL</b>	Sorts a rank-1 array of real numbers $x$ so the $y$ results are algebraically nondecreasing, $y_1 \leq y_2 \leq \dots y_n$ .
<b>SVRGN</b>	Sorts a real array by algebraically increasing value.
<b>SVRGP</b>	Sorts a real array by algebraically increasing value and returns the permutation that rearranges the array.
<b>SVIGN</b>	Sorts an integer array by algebraically increasing value.

## **SORT**

<b>ROUTINE</b>	<b>DESCRIPTION</b>
<b>SVIGP</b>	Sorts an integer array by algebraically increasing value and returns the permutation that rearranges the array.
<b>SVRBN</b>	Sorts a real array by nondecreasing absolute value.
<b>SVRBP</b>	Sorts a real array by nondecreasing absolute value and returns the permutation that rearranges the array.
<b>SVIBN</b>	Sorts an integer array by nondecreasing absolute value.
<b>SVIBP</b>	Sorts an integer array by nondecreasing absolute value and returns the permutation that rearranges the array.

## **SEARCH**

<b>ROUTINE</b>	<b>DESCRIPTION</b>
<b>SRCH</b>	Searches a sorted vector for a given scalar and returns its index.
<b>ISRCH</b>	Searches a sorted integer vector for a given integer and returns its index.
<b>SSRCH</b>	Searches a character vector, sorted in ascending ASCII order, for a given string and returns its index.

## **CHARACTER STRING MANIPULATION**

<b>ROUTINE</b>	<b>DESCRIPTION</b>
<b>ACHAR</b>	Returns a character given its ASCII value.
<b>IACHAR</b>	Returns the integer ASCII value of a character argument.
<b>ICASE</b>	Returns the ASCII value of a character converted to uppercase.
<b>IICSR</b>	Compares two character strings using the ASCII collating sequence but without regard to case.
<b>IIDEX</b>	Determines the position in a string at which a given character sequence begins without regard to case.

## CHARACTER STRING MANIPULATION

ROUTINE	DESCRIPTION
<b>CVTSI</b>	Converts a character string containing an integer number into the corresponding integer form.

## TIME, DATE AND VERSION

ROUTINE	DESCRIPTION
<b>CPSEC</b>	Returns CPU time used in seconds.
<b>TIMDY</b>	Gets time of day.
<b>TDATE</b>	Gets today's date.
<b>NDAYS</b>	Computes the number of days from January 1, 1900, to the given date.
<b>NDYIN</b>	Gives the date corresponding to the number of days since January 1, 1900.
<b>IDYWK</b>	Computes the day of the week for a given date.
<b>VERML</b>	Obtains IMSL MATH LIBRARY-related version, system and serial numbers.

## RANDOM NUMBER GENERATION

ROUTINE	DESCRIPTION
<b>RAND_GEN</b>	Generates a rank-1 array of random numbers.
<b>RNGET</b>	Retrieves the current value of the seed used in the IMSL random number generators.
<b>RNSET</b>	Initializes a random seed for use in the IMSL random number generators.
<b>RNOPT</b>	Selects the uniform (0, 1) multiplicative congruential pseudorandom number generator.
<b>RNIN32</b>	Initializes the 32-bit Mersenne Twister generator using an array.



## RANDOM NUMBER GENERATION

ROUTINE	DESCRIPTION
<b>RNGE32</b>	Retrieves the current table used in the 32-bit Mersenne Twister generator.
<b>RNSE32</b>	Sets the current table used in the 32-bit Mersenne Twister generator.
<b>RNIN64</b>	Initializes the 64-bit Mersenne Twister generator using an array.
<b>RNGE64</b>	Retrieves the current table used in the 64-bit Mersenne Twister generator.
<b>RNSE64</b>	Sets the current table used in the 64-bit Mersenne Twister generator.
<b>RNUNF</b>	Generates a pseudorandom number from a uniform (0, 1) distribution.
<b>RNUN</b>	Generates pseudorandom numbers from a uniform (0, 1) distribution.

## LOW DISCREPANCY SEQUENCES

ROUTINE	DESCRIPTION
<b>FAURE_INIT</b>	Generates pseudorandom numbers from a uniform (0, 1) distribution.
<b>FAURE_FREE</b>	Frees the structure containing information about the Faure sequence.
<b>FAURE_NEXT</b>	Computes a shuffled Faure sequence.

## OPTIONS MANAGER

ROUTINE	DESCRIPTION
<b>IUMAG</b>	This routine handles MATH LIBRARY and STAT LIBRARY type <b>INTEGER</b> options.
<b>UMAG</b>	Gets and puts type <b>REAL</b> options.
<b>SUMAG</b>	This routine handles MATH LIBRARY and STAT LIBRARY type <b>SINGLE PRECISION</b> options.

## OPTIONS MANAGER

ROUTINE

DESCRIPTION

**DUMAG**

This routine handles MATH LIBRARY and STAT LIBRARY type **DOUBLE PRECISION** options.

## LINE PRINTER GRAPHICS

ROUTINE

DESCRIPTION

**PLOTP**

Prints a plot of up to 10 sets of points.

## MISCELLANEOUS

ROUTINE

DESCRIPTION

**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.

**MP\_SETUP**

Initializes or finalizes **MPI**.

# IMSL MATH SPECIAL FUNCTIONS LIBRARY

## CHAPTER 1: ELEMENTARY FUNCTIONS

### ELEMENTARY FUNCTIONS

ROUTINE	DESCRIPTION
<b>CARG</b>	Evaluates the argument of a complex number.
<b>CBRT</b>	Evaluates the cube root.
<b>EXPRL</b>	Evaluates the exponential function factored from first order, $(\text{EXP}(X) - 1.0)/X$ .
<b>LOG10</b>	Extends FORTRAN's generic log10 function to evaluate the principal value of the complex common logarithm.
<b>ALNREL</b>	Evaluates the natural logarithm of one plus the argument.

## CHAPTER 2: HYPERBOLIC FUNCTIONS

### TRIGONOMETRIC FUNCTIONS

ROUTINE	DESCRIPTION
<b>TAN</b>	Extends FORTRAN's generic tan to evaluate the complex tangent.
<b>COT</b>	Evaluates the cotangent.
<b>SINDG</b>	Evaluates the sine for the argument in degrees.
<b>COSDG</b>	Evaluates the cosine for the argument in degrees.
<b>ASIN</b>	Extends FORTRAN's generic <b>ASIN</b> function to evaluate the complex arc sine.

## TRIGONOMETRIC FUNCTIONS

ROUTINE	DESCRIPTION
<b>ACOS</b>	Extends FORTRAN's generic <b>ACOS</b> function to evaluate the complex arc cosine.
<b>ATAN</b>	Extends FORTRAN's generic function <b>ATAN</b> to evaluate the complex arc tangent.
<b>ATAN2</b>	This function extends FORTRAN's generic function <b>ATAN2</b> to evaluate the complex arc tangent of a ratio.

## HYPERBOLIC FUNCTIONS

ROUTINE	DESCRIPTION
<b>SINH</b>	Extends FORTRAN's generic function <b>SINH</b> to evaluate the complex hyperbolic sine.
<b>COSH</b>	Extends FORTRAN's generic function <b>COSH</b> to evaluate the complex hyperbolic cosine.
<b>TANH</b>	Extends FORTRAN's generic function <b>TANH</b> to evaluate the complex hyperbolic tangent.

## TRIGONOMETRIC AND HYPERBOLIC FUNCTIONS

ROUTINE	DESCRIPTION
<b>ASINH</b>	Evaluates the arc hyperbolic sine.
<b>ACOSH</b>	Evaluates the arc hyperbolic cosine.
<b>ATANH</b>	Evaluates the arc hyperbolic tangent.

# CHAPTER 3: EXPONENTIAL INTEGRALS AND RELATED FUNCTIONS

## EXPONENTIAL INTEGRALS AND RELATED FUNCTIONS

ROUTINE	DESCRIPTION
<b>EI</b>	Evaluates the exponential integral for arguments greater than zero and the Cauchy principal value for arguments less than zero.
<b>E1</b>	Evaluates the exponential integral for arguments greater than zero and the Cauchy principal value of the integral for arguments less than zero.

## EXPONENTIAL INTEGRALS AND RELATED FUNCTIONS

ROUTINE	DESCRIPTION
<b>ENE</b>	Evaluates the exponential integral of integer order for arguments greater than zero scaled by EXP(X).
<b>ALI</b>	Evaluates the logarithmic integral.
<b>SI</b>	Evaluates the sine integral.
<b>CI</b>	Evaluates the cosine integral.
<b>CIN</b>	Evaluates a function closely related to the cosine integral.
<b>SHI</b>	Evaluates the hyperbolic sine integral.
<b>CHI</b>	Evaluates the hyperbolic cosine integral.
<b>CINH</b>	Evaluates a function closely related to the hyperbolic cosine integral.

## CHAPTER 4: GAMMA FUNCTION AND RELATED FUNCTIONS

### FACTORIAL FUNCTION

ROUTINE	DESCRIPTION
<b>FAC</b>	Evaluates the factorial of the argument.
<b>BINOM</b>	Evaluates the binomial coefficient.

### GAMMA FUNCTIONS

ROUTINE	DESCRIPTION
<b>GAMMA</b>	Evaluates the complete gamma function.
<b>GAMR</b>	Evaluates the reciprocal gamma function.

## GAMMA FUNCTIONS

ROUTINE	DESCRIPTION
<b>ALNGAM</b>	Evaluates the logarithm of the absolute value of the gamma function.
<b>ALGAMS</b>	Returns the logarithm of the absolute value of the gamma function and the sign of gamma.

## INCOMPLETE GAMMA FUNCTIONS

ROUTINE	DESCRIPTION
<b>GAMI</b>	Evaluates the incomplete gamma function.
<b>GAMIC</b>	Evaluates the complementary incomplete gamma function.
<b>GAMIT</b>	Evaluates the Tricomi form of the incomplete gamma function.

## PSI FUNCTION

ROUTINE	DESCRIPTION
<b>PSI</b>	Evaluates the logarithmic derivative of the gamma function.
<b>PSI1</b>	Evaluates the second derivative of the log gamma function.

## POCHHAMMER'S FUNCTIONS

ROUTINE	DESCRIPTION
<b>POCH</b>	Evaluates a generalization of Pochhammer's symbol.
<b>POCH1</b>	Evaluates a generalization of Pochhammer's symbol starting from the first order.

## BETA FUNCTION

ROUTINE	DESCRIPTION
<b>BETA</b>	Evaluates the complete beta function.
<b>ALBETA</b>	Evaluates the natural logarithm of the complete beta function for positive arguments.

## BETA FUNCTION

ROUTINE	DESCRIPTION
<b>BETAI</b>	Evaluates the incomplete beta function ratio.

## CHAPTER 5: ERROR FUNCTIONS AND RELATED FUNCTIONS

### ERROR FUNCTIONS

ROUTINE	DESCRIPTION
<b>ERF</b>	Evaluates the error function.
<b>ERFC</b>	Evaluates the complementary error function.
<b>ERFCE</b>	Evaluates the exponentially scaled complementary error function.
<b>CERFE</b>	Evaluates the complex scaled complemented error function.
<b>ERFI</b>	Evaluates the inverse error function.
<b>ERFCI</b>	Evaluates the inverse complementary error function.
<b>DAWS</b>	Evaluates Dawson's function.

### FRESNEL INTEGRALS

ROUTINE	DESCRIPTION
<b>FRESC</b>	Evaluates the cosine Fresnel integral.
<b>FRESS</b>	Evaluates the sine Fresnel integral.

## CHAPTER 6: BESSEL FUNCTIONS

### BESSEL FUNCTIONS OF ORDERS 0 AND 1

ROUTINE	DESCRIPTION
<b>BSJ0</b>	Evaluates the Bessel function of the first kind of order zero.
<b>BSJ1</b>	Evaluates the Bessel function of the first kind of order one.
<b>BSY0</b>	Evaluates the Bessel function of the second kind of order zero.
<b>BSY1</b>	Evaluates the Bessel function of the second kind of order one.
<b>BSI0</b>	Evaluates the modified Bessel function of the first kind of order zero.
<b>BSI1</b>	Evaluates the modified Bessel function of the first kind of order one.
<b>BSK0</b>	Evaluates the modified Bessel function of the second kind of order zero.
<b>BSK1</b>	Evaluates the modified Bessel function of the second kind of order one.
<b>BSI0E</b>	Evaluates the exponentially scaled modified Bessel function of the first kind of order zero.
<b>BSI1E</b>	Evaluates the exponentially scaled modified Bessel function of the first kind of order one.
<b>BSK0E</b>	Evaluates the exponentially scaled modified Bessel function of the second kind of order zero.
<b>BSK1E</b>	Evaluates the exponentially scaled modified Bessel function of the second kind of order one.

### SERIES OF BESSEL FUNCTIONS, INTEGER ORDER

ROUTINE	DESCRIPTION
<b>BSJNS</b>	Evaluates a sequence of Bessel functions of the first kind with integer order and real or complex arguments.
<b>BSINS</b>	Evaluates a sequence of modified Bessel functions of the first kind with integer order and real or complex arguments.



## SERIES OF BESSEL FUNCTIONS, REAL ORDER AND ARGUMENT

ROUTINE	DESCRIPTION
<b>BSJS</b>	Evaluates a sequence of Bessel functions of the first kind with real order and real positive arguments.
<b>BSYS</b>	Evaluates a sequence of Bessel functions of the second kind with real nonnegative order and real positive arguments.
<b>BSIS</b>	Evaluates a sequence of modified Bessel functions of the first kind with real order and real positive arguments.
<b>BSIES</b>	Evaluates a sequence of exponentially scaled modified Bessel functions of the first kind with nonnegative real order and real positive arguments.
<b>BSKS</b>	Evaluates a sequence of modified Bessel functions of the second kind of fractional order.
<b>BSKES</b>	Evaluates a sequence of exponentially scaled modified Bessel functions of the second kind of fractional order.

## SERIES OF BESSEL FUNCTIONS, REAL ARGUMENT AND COMPLEX ARGUMENT

ROUTINE	DESCRIPTION
<b>CBJS</b>	Evaluates a sequence of Bessel functions of the first kind with real order and complex arguments.
<b>CBYS</b>	Evaluates a sequence of Bessel functions of the second kind with real order and complex arguments.
<b>CBIS</b>	Evaluates a sequence of modified Bessel functions of the first kind with real order and complex arguments.
<b>CBKS</b>	Evaluates a sequence of modified Bessel functions of the second kind with real order and complex arguments.

## CHAPTER 7: KELVIN FUNCTIONS

### KELVIN FUNCTIONS

ROUTINE	DESCRIPTION
<b>BER0</b>	Evaluates the Kelvin function of the first kind, ber, of order zero.
<b>BEI0</b>	Evaluates the Kelvin function of the first kind, bei, of order zero.

## KELVIN FUNCTIONS

ROUTINE	DESCRIPTION
<b>AKER0</b>	Evaluates the Kelvin function of the second kind, $ker$ , of order zero.
<b>AKEI0</b>	Evaluates the Kelvin function of the second kind, $kei$ , of order zero.
<b>BERP0</b>	Evaluates the derivative of the Kelvin function of the first kind, $ber$ , of order zero.
<b>BEIP0</b>	Evaluates the derivative of the Kelvin function of the first kind, $bei$ , of order zero.
<b>AKERP0</b>	Evaluates the derivative of the Kelvin function of the second kind, $ker$ , of order zero.
<b>AKEIP0</b>	Evaluates the derivative of the Kelvin function of the second kind, $kei$ , of order zero.
<b>BER1</b>	Evaluates the Kelvin function of the first kind, $ber$ , of order one.
<b>BEI1</b>	Evaluates the Kelvin function of the first kind, $bei$ , of order one.
<b>AKER1</b>	Evaluates the Kelvin function of the second kind, $ker$ , of order one.
<b>AKEI1</b>	Evaluates the Kelvin function of the second kind, $kei$ , of order one.

## CHAPTER 8: AIRY FUNCTIONS

### REAL AIRY FUNCTIONS

ROUTINE	DESCRIPTION
<b>AI</b>	Evaluates the Airy function.
<b>BI</b>	Evaluates the Airy function of the second kind.
<b>AID</b>	Evaluates the derivative of the Airy function.

## REAL AIRY FUNCTIONS

ROUTINE	DESCRIPTION
<b>BID</b>	Evaluates the derivative of the Airy function of the second kind.
<b>AIE</b>	Evaluates the exponentially scaled Airy function.
<b>BIE</b>	Evaluates the exponentially scaled Airy function of the second kind.
<b>AIDE</b>	Evaluates the exponentially scaled derivative of the Airy function.
<b>BIDE</b>	Evaluates the exponentially scaled derivative of the Airy function of the second kind.

## COMPLEX AIRY FUNCTIONS

ROUTINE	DESCRIPTION
<b>CAI</b>	Evaluates the Airy function of the first kind for complex arguments.
<b>CBI</b>	Evaluates the Airy function of the second kind for complex arguments.
<b>CAID</b>	Evaluates the derivative of the Airy function of the first kind for complex arguments.
<b>CBID</b>	Evaluates the derivative of the Airy function of the second kind for complex arguments.

## CHAPTER 9: ELLIPTIC FUNCTIONS

### ELLIPTIC FUNCTIONS

ROUTINE	DESCRIPTION
<b>ELK</b>	Evaluates the complete elliptic integral of the kind $K(x)$ .
<b>ELE</b>	Evaluates the complete elliptic integral of the second kind $E(x)$ .
<b>ELRF</b>	Evaluates Carlson's incomplete elliptic integral of the first kind $R_f(x, y, z)$ .

## ELLIPTIC FUNCTIONS

ROUTINE	DESCRIPTION
<b>ELRD</b>	Evaluates Carlson's incomplete elliptic integral of the second kind $R_D(x, y, z)$ .
<b>ELRJ</b>	Evaluates Carlson's incomplete elliptic integral of the third kind $R_J(x, y, z, rho)$ .
<b>ELRC</b>	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

ROUTINE	DESCRIPTION
<b>CWPL</b>	Evaluates the Weierstrass $\wp$ function in the lemniscatic case for complex argument with unit period parallelogram.
<b>CWPLD</b>	Evaluates the first derivative of the Weierstrass $\wp$ function in the lemniscatic case for complex argument with unit period parallelogram.
<b>CWPQ</b>	Evaluates the Weierstrass $\wp$ function in the equianharmonic case for complex argument with unit period parallelogram.
<b>CWPQD</b>	Evaluates the first derivative of the Weierstrass $\wp$ function in the equianharmonic case for complex argument with unit period parallelogram.

### JACOBI ELLIPTIC FUNCTIONS

ROUTINE	DESCRIPTION
<b>EJSN</b>	Evaluates the Jacobi elliptic function $\text{sn}(x, m)$ .
<b>EJCN</b>	Evaluates the Jacobi elliptic function $\text{cn}(x, m)$ .
<b>EJDN</b>	Evaluates the Jacobi elliptic function $\text{dn}(x, m)$ .

## CHAPTER 11: PROBABILITY DISTRIBUTIONS FUNCTIONS AND INVERSES

### DISCRETE RANDOM VARIABLES: CUMULATIVE DISTRIBUTION FUNCTIONS AND PROBABILITY DENSITY FUNCTIONS

ROUTINE	DESCRIPTION
<b>BINDF</b>	Evaluates the binomial cumulative distribution function.
<b>BINPR</b>	Evaluates the binomial probability density function.
<b>GEODF</b>	Evaluates the discrete geometric cumulative distribution function.
<b>GEOIN</b>	Evaluates the inverse of the geometric cumulative distribution function.
<b>GEOPR</b>	Evaluates the discrete geometric probability density function.
<b>HYPDF</b>	Evaluates the hypergeometric cumulative distribution function.
<b>HYPPR</b>	Evaluates the hypergeometric probability density function.
<b>POIDF</b>	Evaluates the Poisson cumulative distribution function.
<b>POIPR</b>	Evaluates the Poisson probability density function.
<b>UNDDF</b>	Evaluates the discrete uniform cumulative distribution function.
<b>UNDIN</b>	Evaluates the inverse of the discrete uniform cumulative distribution function.
<b>UNDPR</b>	Evaluates the discrete uniform probability density function.

## CONTINUOUS RANDOM VARIABLES: DISTRIBUTION FUNCTIONS AND THEIR INVERSES

ROUTINE	DESCRIPTION
<b>AKS1DF</b>	Evaluates the cumulative distribution function of the one-sided Kolmogorov-Smirnov goodness of fit $D^+$ or $D^-$ test statistic based on continuous data for one sample.
<b>AKS2DF</b>	Evaluates the cumulative distribution function of the one-sided Kolmogorov-Smirnov goodness of fit $D$ test statistic based on continuous data for two samples.
<b>ALNDF</b>	Evaluates the lognormal cumulative distribution function.
<b>ALNIN</b>	Evaluates the inverse of the lognormal cumulative distribution function.
<b>ALNPR</b>	Evaluates the lognormal probability density function.
<b>ANORDF</b>	Evaluates the standard normal (Gaussian) cumulative distribution function.
<b>ANORIN</b>	Evaluates the inverse of the standard normal (Gaussian) cumulative distribution function.
<b>ANORPR</b>	Evaluates the normal probability density function.
<b>BETDF</b>	Evaluates the beta cumulative distribution function.
<b>BETIN</b>	Evaluates the inverse of the beta cumulative distribution function.
<b>BETPR</b>	Evaluates the beta probability density function.
<b>BETNDF</b>	Evaluates the noncentral beta cumulative distribution function (CDF).
<b>BETNIN</b>	Evaluates the inverse of the noncentral beta cumulative distribution function (CDF).
<b>BETNPR</b>	Evaluates the noncentral beta probability density function.
<b>BNRDF</b>	Evaluates the bivariate normal cumulative distribution function.

## CONTINUOUS RANDOM VARIABLES: DISTRIBUTION FUNCTIONS AND THEIR INVERSES

ROUTINE	DESCRIPTION
<b>CHIDF</b>	Evaluates the chi-squared cumulative distribution function.
<b>CHIIN</b>	Evaluates the inverse of the chi-squared cumulative distribution function.
<b>CHIPR</b>	Evaluates the chi-squared probability density function.
<b>CSNDF</b>	Evaluates the noncentral chi-squared cumulative distribution function.
<b>CSNIN</b>	Evaluates the inverse of the noncentral chi-squared cumulative distribution function.
<b>CSNPR</b>	Evaluates the noncentral chi-squared probability density function.
<b>EXPDF</b>	Evaluates the exponential cumulative distribution function.
<b>EXPIN</b>	Evaluates the inverse of the exponential cumulative distribution function.
<b>EXPPR</b>	Evaluates the exponential probability density function.
<b>EXVDF</b>	Evaluates the extreme value cumulative distribution function.
<b>EXVIN</b>	Evaluates the inverse of the extreme value cumulative distribution function.
<b>EXVPR</b>	Evaluates the extreme value probability density function.
<b>FDF</b>	Evaluates the $F$ cumulative distribution function.
<b>FIN</b>	Evaluates the inverse of the $F$ cumulative distribution function.
<b>FPR</b>	Evaluates the $F$ probability density function.
<b>FNDF</b>	Evaluates the noncentral $F$ cumulative distribution function (CDF).

## CONTINUOUS RANDOM VARIABLES: DISTRIBUTION FUNCTIONS AND THEIR INVERSES

ROUTINE	DESCRIPTION
<b>FNIN</b>	Evaluates the inverse of the noncentral $F$ cumulative distribution function (CDF).
<b>FNPR</b>	Evaluates the noncentral $F$ probability density function.
<b>GAMDF</b>	Evaluates the gamma cumulative distribution function.
<b>GAMIN</b>	Evaluates the inverse of the gamma cumulative distribution function.
<b>GAMPR</b>	Evaluates the gamma probability density function.
<b>RALDF</b>	Evaluates the Rayleigh cumulative distribution function.
<b>RALIN</b>	Evaluates the inverse of the Rayleigh cumulative distribution function.
<b>RALPR</b>	Evaluates the Rayleigh probability density function.
<b>TDF</b>	Evaluates the Student's $t$ cumulative distribution function.
<b>TIN</b>	Evaluates the inverse of the Student's $t$ cumulative distribution function.
<b>TPR</b>	Evaluates the Student's $t$ probability density function.
<b>TNDF</b>	Evaluates the noncentral Student's $t$ cumulative distribution function.
<b>TNIN</b>	Evaluates the inverse of the noncentral Student's $t$ cumulative distribution function.
<b>TNPR</b>	Evaluates the noncentral Student's $t$ probability density function.
<b>UNDF</b>	Evaluates the uniform cumulative distribution function.
<b>UNIN</b>	Evaluates the inverse of the uniform cumulative distribution function.



## CONTINUOUS RANDOM VARIABLES: DISTRIBUTION FUNCTIONS AND THEIR INVERSES

ROUTINE	DESCRIPTION
<b>UNPR</b>	Evaluates the uniform probability density function.
<b>WBLDF</b>	Evaluates the Weibull cumulative distribution function.
<b>WBLIN</b>	Evaluates the inverse of the Weibull cumulative distribution function.
<b>WBLPR</b>	Evaluates the Weibull probability density function.

## GENERAL CONTINUOUS RANDOM VARIABLE

ROUTINE	DESCRIPTION
<b>GCDF</b>	Evaluates a general continuous cumulative distribution function given ordinates of the density.
<b>GCIN</b>	Evaluates the inverse of a general continuous cumulative distribution function given ordinates of the density.
<b>GFNIN</b>	Evaluates the inverse of a general continuous cumulative distribution function given in a subprogram.

## CHAPTER 12: MATHIEU FUNCTIONS

### MATHIEU FUNCTIONS

ROUTINE	DESCRIPTION
<b>MATEE</b>	Evaluates the eigenvalues for the periodic Mathieu functions.
<b>MATCE</b>	Evaluates a sequence of even, periodic, integer order, real Mathieu functions.
<b>MATSE</b>	Evaluates a sequence of odd, periodic, integer order, real Mathieu functions.

## CHAPTER 13: MISCELLANEOUS FUNCTIONS

### MISCELLANEOUS FUNCTIONS

ROUTINE	DESCRIPTION
<b>SPENC</b>	Evaluates a form of Spence's integral.
<b>INITS</b>	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.
<b>CSEVL</b>	Evaluates the $N$ -term Chebyshev series.

## REFERENCE MATERIAL: LIBRARY ENVIRONMENTS UTILITIES

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

ROUTINE	DESCRIPTION
<b>ERSET</b>	Sets error handler default print and stop actions.
<b>IERCD</b>	Retrieves the code for an informational error.
<b>N1RTY</b>	Retrieves an error type for the most recently called IMSL routine.
<b>IMACH</b>	Retrieves integer machine constants.
<b>AMACH</b>	Retrieves single precision machine constants.
<b>DMACH</b>	Retrieves double precision machine constants.
<b>IFNAN</b>	Checks if a floating-point number is NaN (not a number).
<b>UMACH</b>	Sets or Retrieves input or output device unit numbers.

# IMSL STAT LIBRARY

## CHAPTER 1: BASIC STATISTICS

### FREQUENCY TABULATIONS

ROUTINE	DESCRIPTION
<b>OWFRQ</b>	Tallies observations into a one-way frequency table.
<b>TWFRQ</b>	Tallies observations into a two-way frequency table.
<b>FREQ</b>	Tallies multivariate observations into a multiway frequency table.

### UNIVARIATE SUMMARY STATISTICS

ROUTINE	DESCRIPTION
<b>UVSTA</b>	Computes basic univariate statistics.

### RANKS AND ORDER STATISTICS

ROUTINE	DESCRIPTION
<b>RANKS</b>	Computes the ranks, normal scores, or exponential scores for a vector of observations
<b>LETTR</b>	Produces a letter value summary.
<b>ORDST</b>	Determines order statistics.
<b>EQTIL</b>	Computes empirical quantiles.

## PARAMETRIC ESTIMATES AND TESTS

ROUTINE	DESCRIPTION
<b>TWOMV</b>	Computes statistics for mean and variance inferences using samples from two normal populations.
<b>BINES</b>	Estimates the parameter $p$ of the binomial distribution.
<b>POIES</b>	Estimates the parameter of the Poisson distribution.
<b>NRCES</b>	Computes maximum likelihood estimates of the mean and variance from grouped and/or censored normal data.

## GROUPED DATA

ROUTINE	DESCRIPTION
<b>GRPES</b>	Computes basic statistics from grouped data.

## CONTINUOUS DATA IN A TABLE

ROUTINE	DESCRIPTION
<b>CSTAT</b>	Computes cell frequencies, cell means, and cell sums of squares for multivariate data.
<b>MEDPL</b>	Computes a median polish of a two-way table.

## CHAPTER 2: REGRESSION

### SIMPLE LINEAR REGRESSION

ROUTINE	DESCRIPTION
<b>RLINE</b>	Fits a line to a set of data points using least squares.
<b>RONE</b>	Analyzes a simple linear regression model.
<b>RINCF</b>	Performs response control given a fitted simple linear regression model.
<b>RINPF</b>	Performs inverse prediction given a fitted simple linear regression model.

## MULTIVARIATE GENERAL LINEAR MODEL ANALYSIS

### MODEL FITTING

ROUTINE	DESCRIPTION
<b>RLSE</b>	Fits a multiple linear regression model using least squares.
<b>RCOV</b>	Fits a multivariate linear regression model given the variance-covariance matrix.
<b>RGIVN</b>	Fits a multivariate linear regression model via fast Givens transformations.
<b>RGLM</b>	Fits a multivariate general linear model.
<b>RLEQU</b>	Fits a multivariate linear regression model with linear equality restrictions $HB = G$ imposed on the regression parameters given results from routine <b>RGIVN</b> after <b>IDO</b> = 1 and <b>IDO</b> = 2 and prior to <b>IDO</b> = 3.

### STATISTICAL INFERENCE AND DIAGNOSTICS

ROUTINE	DESCRIPTION
<b>RSTAT</b>	Computes statistics related to a regression fit given the coefficient estimates.
<b>RCOVB</b>	Computes the estimated variance-covariance matrix of the estimated regression coefficients given the $R$ matrix.
<b>CESTI</b>	Constructs an equivalent completely testable multivariate general linear hypothesis $HBU = G$ from a partially testable hypothesis $H_pBU = G_p$ .
<b>RHPSS</b>	Computes the matrix of sums of squares and crossproducts for the multivariate general linear hypothesis $HBU = G$ given the coefficient estimates and the $R$ matrix.
<b>RHPTE</b>	Performs tests for a multivariate general linear hypothesis $HBU = G$ given the hypothesis sums of squares and crossproducts matrix $S_H$ and the error sums of squares and crossproducts matrix $S_E$ .
<b>RLOFE</b>	Computes a lack of fit test based on exact replicates for a fitted regression model.
<b>RLOFN</b>	Computes a lack of fit test based on near replicates for a fitted regression model.
<b>RCASE</b>	Computes case statistics and diagnostics given data points, coefficient estimates and the $R$ matrix for a fitted general linear model.

## STATISTICAL INFERENCE AND DIAGNOSTICS

ROUTINE	DESCRIPTION
<b>ROTIN</b>	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

ROUTINE	DESCRIPTION
<b>GCLAS</b>	Gets the unique values of each classification variable.
<b>GRGLM</b>	Generates regressors for a general linear model.

## VARIABLES SELECTION

ROUTINE	DESCRIPTION
<b>RBEST</b>	Selects the best multiple linear regression models.
<b>RSTEP</b>	Builds multiple linear regression models using forward selection, backward selection or stepwise selection.
<b>GSWEP</b>	Performs a generalized sweep of a row of a nonnegative definite matrix.
<b>RSUBM</b>	Retrieves a symmetric submatrix from a symmetric matrix.

## POLYNOMIAL REGRESSION AND SECOND-ORDER MODELS

### POLYNOMIAL REGRESSION ANALYSIS

ROUTINE	DESCRIPTION
<b>RCURV</b>	Fits a polynomial curve using least squares.
<b>RPOLY</b>	Analyzes a polynomial regression model.

### SECOND-ORDER MODEL DESIGN

ROUTINE	DESCRIPTION
<b>RCOMP</b>	Generates an orthogonal central composite design.

## UTILITY ROUTINES FOR POLYNOMIAL MODELS AND SECOND-ORDER MODELS

ROUTINE	DESCRIPTION
<b>RFORP</b>	Fits an orthogonal polynomial regression model.
<b>RSTAP</b>	Computes summary statistics for a polynomial regression model given the fit based on orthogonal polynomials.
<b>RCASP</b>	Computes case statistics for a polynomial regression model given the fit based on orthogonal polynomials.
<b>OPOLY</b>	Generates orthogonal polynomials with respect to x-values and specified weights.
<b>GCSCP</b>	Generates centered variables, squares, and crossproducts.
<b>TCSCP</b>	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

ROUTINE	DESCRIPTION
<b>RNLIN</b>	Fits a nonlinear regression model.

## FITTING LINEAR MODELS BASED ON CRITERIA OTHER THAN LEAST SQUARES

ROUTINE	DESCRIPTION
<b>RLAV</b>	Fits a multiple linear regression model using the least absolute values criterion.
<b>RLLP</b>	Fits a multiple linear regression model using the $L_p$ norm criterion.
<b>RLMV</b>	Fits a multiple linear regression model using the minimax criterion.
<b>PLSR</b>	Performs partial least squares regression for one or more response variables and one or more predictor variables.

## CHAPTER 3: CORRELATION

### THE CORRELATION MATRIX

ROUTINE	DESCRIPTION
<b>CORVC</b>	Computes the variance-covariance or correlation matrix.
<b>COVPL</b>	Computes a pooled variance-covariance matrix from the observations.
<b>PCORR</b>	Computes partial correlations or covariances from the covariance or correlation matrix.
<b>RBCOV</b>	Computes a robust estimate of a covariance matrix and mean vector.

### CORRELATION MEASURES FOR A CONTINGENCY TABLE

ROUTINE	DESCRIPTION
<b>CTRHO</b>	Estimates the bivariate normal correlation coefficient using a contingency table.
<b>TETCC</b>	Categorizes bivariate data and computes the tetrachoric correlation coefficient.

### A DICHOTOMOUS VARIABLE WITH A CLASSIFICATION VARIABLE

ROUTINE	DESCRIPTION
<b>BSPBS</b>	Computes the biserial and point-biserial correlation coefficients for a dichotomous variable and a numerically measurable classification variable.
<b>BSCAT</b>	Computes the biserial correlation coefficient for a dichotomous variable and a classification variable.

### MEASURES BASED UPON RANKS

ROUTINE	DESCRIPTION
<b>CNCRD</b>	Calculates and tests the significance of the Kendall coefficient of concordance.
<b>KENDL</b>	Computes and tests Kendall's rank correlation coefficient.
<b>KENDP</b>	Computes the frequency distribution of the total score in Kendall's rank correlation coefficient.



## CHAPTER 4: ANALYSIS OF VARIANCE

### GENERAL ANALYSIS

ROUTINE	DESCRIPTION
<b>AONEW</b>	Analyzes a one-way classification model.
<b>AONEC</b>	Analyzes a one-way classification model with covariates.
<b>ATWOB</b>	Analyzes a randomized block design or a two-way balanced design.
<b>ABIBD</b>	Analyzes a balanced incomplete block design or a balanced lattice design.
<b>ALATN</b>	Analyzes a Latin square design.
<b>ANWAY</b>	Analyzes a balanced $n$ -way classification model with fixed effects.
<b>ABALD</b>	Analyzes a balanced complete experimental design for a fixed, random, or mixed model.
<b>ANEST</b>	Analyzes a completely nested random model with possibly unequal numbers in the subgroups.

### INFERENCE ON MEANS AND VARIANCE COMPONENTS

ROUTINE	DESCRIPTION
<b>CTRST</b>	Computes contrast estimates and sums of squares.
<b>SCIPM</b>	Computes simultaneous confidence intervals on all pairwise differences of means.
<b>SNKMC</b>	Performs Student-Newman-Keuls multiple comparison test.
<b>CIDMS</b>	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

ROUTINE	DESCRIPTION
<b>ROREX</b>	Reorders the responses from a balanced complete experimental design.

## CHAPTER 5: CATEGORICAL AND DISCRETE DATA ANALYSIS

### STATISTICS IN THE TWO-WAY CONTINGENCY TABLE

ROUTINE	DESCRIPTION
<b>CTTWO</b>	Performs a chi-squared analysis of a 2 by 2 contingency table.
<b>CTCHI</b>	Performs a chi-squared analysis of a two-way contingency table.
<b>CTPRB</b>	Computes exact probabilities in a two-way contingency table.
<b>CTEPR</b>	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

ROUTINE	DESCRIPTION
<b>PRPFT</b>	Performs iterative proportional fitting of a contingency table using a log-linear model.
<b>CTLLN</b>	Computes model estimates and associated statistics for a hierarchical log-linear model.
<b>CTPAR</b>	Computes model estimates and covariances in a fitted log-linear model.
<b>CTASC</b>	Computes partial association statistics for log-linear models in a multidimensional contingency table.
<b>CTSTP</b>	Builds hierarchical log-linear models using forward selection, backward selection, or stepwise selection.

### RANDOMIZATION TESTS

ROUTINE	DESCRIPTION
<b>CTRAN</b>	Performs generalized Mantel-Haenszel tests in a stratified contingency table.

## GENERALIZED CATEGORICAL MODELS

ROUTINE	DESCRIPTION
<b>CTGLM</b>	Analyzes categorical data using logistic, Probit, Poisson, and other generalized linear models.

## WEIGHTED LEAST SQUARES ANALYSIS

ROUTINE	DESCRIPTION
<b>CTWLS</b>	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

ROUTINE	DESCRIPTION
<b>SIGNT</b>	Performs a sign test of the hypothesis that a given value is in a specified quantile of a distribution.
<b>SNRNK</b>	Performs a Wilcoxon signed rank test.

### TESTS OF TREND

ROUTINE	DESCRIPTION
<b>NCTRD</b>	Performs the Noether test for cyclical trend.
<b>SDPLC</b>	Performs the Cox and Stuart sign test for trends in dispersion and location.

### TIES

ROUTINE	DESCRIPTION
<b>NTIES</b>	Computes tie statistics for a sample of observations.

## MORE THAN TWO SAMPLES

### ONE-WAY TESTS OF LOCATION

#### ROUTINE

#### DESCRIPTION

**KRSKL**

Performs a Kruskal-Wallis test for identical population medians.

**BHAKV**

Performs a Bhapkar  $V$  test.

### TWO-WAY TESTS OF LOCATION

#### ROUTINE

#### DESCRIPTION

**FRDMN**

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

**QTEST**

Performs a Cochran  $Q$  test for related observations.

### TESTS FOR TREND

#### ROUTINE

#### DESCRIPTION

**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

#### ROUTINE

#### DESCRIPTION

**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 tests for multivariate normality.

**ADNRM**

Performs an Anderson-Darling test for normality.

## GENERAL GOODNESS-OF-FIT TESTS FOR A SPECIFIED DISTRIBUTION

ROUTINE	DESCRIPTION
<b>CVMNRM</b>	Performs a Cramer-von Mises test for normality.

## TWO SAMPLE TESTS

ROUTINE	DESCRIPTION
<b>KSTWO</b>	Performs a Kolmogorov-Smirnov two-sample test.

## TESTS FOR RANDOMNESS

ROUTINE	DESCRIPTION
<b>RUNS</b>	Performs a runs up test.
<b>PAIRS</b>	Performs a pairs test.
<b>DSQAR</b>	Performs a $d^2$ test.
<b>DCUBE</b>	Performs a triplets test.

# CHAPTER 8: TIME SERIES ANALYSIS AND FORECASTING

## GENERAL METHODOLOGY

### TIME SERIES TRANSFORMATION

ROUTINE	DESCRIPTION
<b>BCTR</b>	Performs a forward or an inverse Box-Cox (power) transformation.
<b>DIFF</b>	Differences a time series.
<b>ESTIMATE_MISSING</b>	Estimates missing values in a time series.
<b>SEASONAL_FIT</b>	Determines an optimal differencing for seasonal adjustments of a time series.

## SAMPLE CORRELATION FUNCTION

ROUTINE	DESCRIPTION
<b>ACF</b>	Computes the sample autocorrelation function of a stationary time series.
<b>PACF</b>	Computes the sample partial autocorrelation function of a stationary time series.
<b>CCF</b>	Computes the sample cross-correlation function of two stationary time series.
<b>MCCF</b>	Computes the multichannel cross-correlation function of two mutually stationary multichannel time series.

## TIME DOMAIN METHODOLOGY

### NONSEASONAL TIME SERIES MODEL ESTIMATION

ROUTINE	DESCRIPTION
<b>ARMME</b>	Computes method of moments estimates of the autoregressive parameters of an ARMA model.
<b>MAMME</b>	Computes method of moments estimates of the moving average parameters of an ARMA model.
<b>NSPE</b>	Computes preliminary estimates of the autoregressive and moving average parameters of an ARMA model.
<b>NSLSE</b>	Computes least-squares estimates of parameters for a nonseasonal ARMA model.
<b>MAX_ARMA</b>	Exact maximum likelihood estimation of the parameters in a univariate ARMA (autoregressive, moving average) time series model.
<b>REG_ARIMA</b>	Fits a univariate, non-seasonal ARIMA time series model with the inclusion of one or more regression variables.
<b>GARCH</b>	Computes estimates of the parameters of a GARCH( $p,q$ ) model.
<b>SPWF</b>	Computes the Wiener forecast operator for a stationary stochastic process.
<b>NSBJF</b>	Computes Box-Jenkins forecasts and their associated probability limits for a nonseasonal ARMA model.

**TRANSFER FUNCTION MODEL****ROUTINE****DESCRIPTION****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****ROUTINE****DESCRIPTION****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****ROUTINE****DESCRIPTION****AUTO\_UNI\_AR**

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

**TS\_OUTLIER\_IDENTIFICATION**

Detects and determines outliers and simultaneously estimates the model parameters in a time series whose underlying outlier free series follows a general seasonal or nonseasonal ARMA model.

**TS\_OUTLIER\_FORECAST**

Computes forecasts, associated probability limits and  $\psi$  weights for an outlier contaminated time series.

**AUTO\_ARIMA**

Automatically identifies time series outliers, determines parameters of a multiplicative seasonal ARIMA  $(p, 0, q) \times (0, d, 0)_s$  model and produces forecasts that incorporate the effects of outliers whose effects persist beyond the end of the series.

**AUTO\_FPE\_UNI\_AR**

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

**AUTO\_PARM**

Estimates structural breaks in non-stationary univariate time series.

**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

### ROUTINE

### DESCRIPTION

**BAY\_SEA**

Bayesian seasonal adjustment modeling. The model allows for a decomposition of a time series into trend, seasonal, and an error component.

## CONTROLLER DESIGN

### ROUTINE

### DESCRIPTION

**OPT\_DES**

Optimal controller design which allows for multiple channels for both the controlled and manipulated variables.

## DIAGNOSTICS

### ROUTINE

### DESCRIPTION

**LOFCF**

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

## FREQUENCY DOMAIN METHODOLOGY

### SMOOTHING FUNCTIONS

#### ROUTINE

#### DESCRIPTION

**DIRIC**

Computes the Dirichlet kernel.

**FEJER**

Computes the Fejér kernel.

### SPECTRAL DENSITY ESTIMATION

#### ROUTINE

#### DESCRIPTION

**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

ROUTINE	DESCRIPTION
<b>CPFFT</b>	Computes the cross periodogram of two stationary time series using a fast Fourier transform.
<b>CSSWD</b>	Estimates the nonnormalized cross-spectral density of two stationary time series using a spectral window given the time series data.
<b>CSSWP</b>	Estimates the nonnormalized cross-spectral density of two stationary time series using a spectral window given the spectral densities and cross periodogram.
<b>CSWED</b>	Estimates the nonnormalized cross-spectral density of two stationary time series using a weighted cross periodogram given the time series data.
<b>CSWEP</b>	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

ROUTINE	DESCRIPTION
<b>PRINC</b>	Computes principal components from a variance-covariance matrix or a correlation matrix.
<b>KPRIN</b>	Maximum likelihood or least-squares estimates for principal components from one or more matrices.

### FACTOR ANALYSIS

#### FACTOR EXTRACTION

ROUTINE	DESCRIPTION
<b>FACTR</b>	Extracts initial factor loading estimates in factor analysis.

#### FACTOR ROTATION AND SUMMARIZATION

ROUTINE	DESCRIPTION
<b>FROTA</b>	Computes an orthogonal rotation of a factor loading matrix using a generalized orthomax criterion, including quartimax, varimax, and equamax rotations.
<b>FOPCS</b>	Computes an orthogonal Procrustes rotation of a factor-loading matrix using a target matrix.

## FACTOR ROTATION AND SUMMARIZATION

ROUTINE	DESCRIPTION
<b>FDOBL</b>	Computes a direct oblimin rotation of a factor loading matrix.
<b>FPRMX</b>	Computes an oblique Promax or Procrustes rotation of a factor loading matrix using a target matrix, including pivot and power vector options.
<b>FHARR</b>	Computes an oblique rotation of an unrotated factor loading matrix using the Harris-Kaiser method.
<b>FGCRF</b>	Computes direct oblique rotation according to a generalized fourth-degree polynomial criterion.
<b>FIMAG</b>	Computes the image transformation matrix.
<b>FRVAR</b>	Computes the factor structure and the variance explained by each factor.

## FACTOR SCORES

ROUTINE	DESCRIPTION
<b>FCOEF</b>	Computes a matrix of factor score coefficients for input to the routine FSCOR.
<b>FSCOR</b>	Computes a set of factor scores given the factor score coefficient matrix.

## RESIDUAL CORRELATION

ROUTINE	DESCRIPTION
<b>FRESI</b>	Computes communalities and the standardized factor residual correlation matrix.

## INDEPENDENCE OF SETS OF VARIABLES AND CANONICAL CORRELATION ANALYSIS

ROUTINE	DESCRIPTION
<b>MVIND</b>	Computes a test for the independence of k sets of multivariate normal variables.
<b>CANCR</b>	Performs canonical correlation analysis from a data matrix.
<b>CANVC</b>	Performs canonical correlation analysis from a variance-covariance matrix or a correlation matrix.

## CHAPTER 10: DISCRIMINANT ANALYSIS

### PARAMETRIC DISCRIMINATION

ROUTINE	DESCRIPTION
<b>DSCR</b>	Performs a linear or a quadratic discriminant function analysis among several known groups.
<b>DMSCR</b>	Uses Fisher's linear discriminant analysis method to reduce the number of variables.

### NONPARAMETRIC DISCRIMINATION

ROUTINE	DESCRIPTION
<b>NNBRD</b>	Performs $k$ nearest neighbor discrimination.

## CHAPTER 11: CLUSTER ANALYSIS

### HIERARCHICAL CLUSTER ANALYSIS

ROUTINE	DESCRIPTION
<b>CDIST</b>	Computes a matrix of dissimilarities (or similarities) between the columns (or rows) of a matrix.
<b>CLINK</b>	Performs a hierarchical cluster analysis given a distance matrix.
<b>CNUMB</b>	Computes cluster membership for a hierarchical cluster tree.

### K-MEANS CLUSTER ANALYSIS

ROUTINE	DESCRIPTION
<b>KMEAN</b>	Performs a $K$ -means (centroid) cluster analysis.

## CHAPTER 12: SAMPLING

SAMPLING	
ROUTINE	DESCRIPTION
<b>SMPPR</b>	Computes statistics for inferences regarding the population proportion and total given proportion data from a simple random sample.
<b>SMPPS</b>	Computes statistics for inferences regarding the population proportion and total given proportion data from a stratified random sample.
<b>SMPRR</b>	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.
<b>SMPRS</b>	Computes statistics for inferences regarding the population mean and total using ratio or regression estimation given continuous data from a stratified random sample.
<b>SMPSC</b>	Computes statistics for inferences regarding the population mean and total using single stage cluster sampling with continuous data.
<b>SMPSR</b>	Computes statistics for inferences regarding the population mean and total, given data from a simple random sample.
<b>SMPSS</b>	Computes statistics for inferences regarding the population mean and total, given data from a stratified random sample.
<b>SMPST</b>	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	
ROUTINE	DESCRIPTION
<b>KAPMR</b>	Computes Kaplan-Meier estimates of survival probabilities in stratified samples.
<b>KTBLE</b>	Prints Kaplan-Meier estimates of survival probabilities in stratified samples.
<b>TRNBL</b>	Computes Turnbull's generalized Kaplan-Meier estimates of survival probabilities in samples with interval censoring.
<b>PHGLM</b>	Analyzes time event data via the proportional hazards model.

## SURVIVAL ANALYSIS

ROUTINE	DESCRIPTION
<b>SVGLM</b>	Analyzes censored survival data using a generalized linear model.
<b>STBLE</b>	Estimates survival probabilities and hazard rates for various parametric models.

## ACTUARIAL TABLES

ROUTINE	DESCRIPTION
<b>ACTBL</b>	Produces population and cohort life tables.

# CHAPTER 14: MULTIDIMENSIONAL SCALING

## MULTIDIMENSIONAL SCALING ROUTINES

ROUTINE	DESCRIPTION
<b>MSIDV</b>	Performs individual-differences multidimensional scaling for metric data using alternating least squares.

## UTILITY ROUTINES

ROUTINE	DESCRIPTION
<b>MSDST</b>	Computes distances in a multidimensional scaling model.
<b>MSSTN</b>	Transforms dissimilarity/similarity matrices and replaces missing values by estimates to obtain standardized dissimilarity matrices.
<b>MSDBL</b>	Obtains normalized product-moment (double centered) matrices from dissimilarity matrices.
<b>MSINI</b>	Computes initial estimates in multidimensional scaling models.
<b>MSTRS</b>	Computes various stress criteria in multidimensional scaling.

## CHAPTER 15: DENSITY AND HAZARD ESTIMATION

### ESTIMATES FOR A DENSITY

ROUTINE	DESCRIPTION
<b>DESPL</b>	Performs nonparametric probability density function estimation by the penalized likelihood method.
<b>DESKN</b>	Performs nonparametric probability density function estimation by the kernel method.
<b>DNFFT</b>	Computes Gaussian kernel estimates of a univariate density via the fast Fourier transform over a fixed interval.
<b>DESPT</b>	Estimates a probability density function at specified points using linear or cubic interpolation.

### MODIFIED LIKELIHOOD ESTIMATES FOR HAZARDS

ROUTINE	DESCRIPTION
<b>HAZRD</b>	Performs nonparametric hazard rate estimation using kernel functions and quasi-likelihoods.
<b>HAZEZ</b>	Performs nonparametric hazard rate estimation using kernel functions. Easy-to-use version of HAZRD.
<b>HAZST</b>	Performs hazard rate estimation over a grid of points using a kernel function.

## CHAPTER 16: LINE PRINTER GRAPHICS

### HISTOGRAMS

ROUTINE	DESCRIPTION
<b>VHSTP</b>	Prints a vertical histogram.
<b>VHS2P</b>	Prints a vertical histogram with every bar subdivided into two parts.
<b>HHSTP</b>	Prints a horizontal histogram.

## SCATTER PLOTS

ROUTINE	DESCRIPTION
<b>SCTP</b>	Prints a scatter plot of several groups of data.

## EXPLORATORY DATA ANALYSIS

ROUTINE	DESCRIPTION
<b>BOXP</b>	Prints boxplots for one or more samples.
<b>STMLP</b>	Prints a stem-and-leaf plot.

## EMPIRICAL PROBABILITY DISTRIBUTION

ROUTINE	DESCRIPTION
<b>CDFP</b>	Prints a sample cumulative distribution function (CDF), a theoretical CDF, and confidence band information.
<b>CDF2P</b>	Prints a plot of two sample cumulative distribution functions.
<b>PROBP</b>	Prints a probability plot.

## OTHER GRAPHICS ROUTINES

ROUTINE	DESCRIPTION
<b>PLOTP</b>	Prints a plot of up to 10 sets of points.
<b>TREEP</b>	Prints a binary tree.

# CHAPTER 17: PROBABILITY DISTRIBUTIONS FUNCTIONS AND INVERSES

## PROBABILITY DISTRIBUTION FUNCTIONS AND INVERSES

ROUTINE	DESCRIPTION
<b>BINDF</b>	Evaluates the binomial cumulative distribution function.
<b>BINPR</b>	Evaluates the binomial probability density function.

## PROBABILITY DISTRIBUTION FUNCTIONS AND INVERSES

ROUTINE	DESCRIPTION
<b>GEODF</b>	Evaluates the discrete geometric cumulative distribution function.
<b>GEOIN</b>	Evaluates the inverse of the geometric cumulative distribution function.
<b>GEOPR</b>	Evaluates the discrete geometric probability density function.
<b>HYPDF</b>	Evaluates the hypergeometric cumulative distribution function.
<b>HYPPR</b>	Evaluates the hypergeometric probability density function.
<b>POIDF</b>	Evaluates the Poisson cumulative distribution function.
<b>POIPR</b>	Evaluates the Poisson probability density function.
<b>UNDDF</b>	Evaluates the discrete uniform cumulative distribution function.
<b>UNDIN</b>	Evaluates the inverse of the discrete uniform cumulative distribution function.
<b>UNDPR</b>	Evaluates the discrete uniform probability density function.

## CONTINUOUS RANDOM VARIABLES: DISTRIBUTION FUNCTIONS AND THEIR INVERSES

ROUTINE	DESCRIPTION
<b>AKS1DF</b>	Evaluates the cumulative distribution function of the one-sided Kolmogorov-Smirnov goodness of fit $D^+$ or $D^-$ test statistic based on continuous data for one sample.
<b>AKS2DF</b>	Evaluates the cumulative distribution function of the one-sided Kolmogorov-Smirnov goodness of fit $D$ test statistic based on continuous data for two samples.
<b>ALNDF</b>	Evaluates the lognormal cumulative distribution function.
<b>ALNIN</b>	Evaluates the inverse of the lognormal cumulative distribution function.



## CONTINUOUS RANDOM VARIABLES: DISTRIBUTION FUNCTIONS AND THEIR INVERSES

ROUTINE	DESCRIPTION
<b>ALNPR</b>	Evaluates the lognormal probability density function.
<b>ANORDF</b>	Evaluates the standard normal (Gaussian) cumulative distribution function.
<b>ANORIN</b>	Evaluates the inverse of the standard normal (Gaussian) cumulative distribution function.
<b>ANORPR</b>	Evaluates the normal probability density function.
<b>BETDF</b>	Evaluates the beta cumulative distribution function.
<b>BETIN</b>	Evaluates the inverse of the beta cumulative distribution function.
<b>BETPR</b>	Evaluates the beta probability density function.
<b>BETNDF</b>	Evaluates the noncentral beta cumulative distribution function (CDF).
<b>BETNIN</b>	Evaluates the inverse of the noncentral beta cumulative distribution function (CDF).
<b>BETNPR</b>	Evaluates the noncentral beta probability density function.
<b>BNRDF</b>	Evaluates the bivariate normal cumulative distribution function.
<b>CHIDF</b>	Evaluates the chi-squared cumulative distribution function.
<b>CHIIN</b>	Evaluates the inverse of the chi-squared cumulative distribution function.
<b>CHIPR</b>	Evaluates the chi-squared probability density function.
<b>CSNDF</b>	Evaluates the noncentral chi-squared cumulative distribution function.
<b>CSNIN</b>	Evaluates the inverse of the noncentral chi-squared cumulative distribution function.

## CONTINUOUS RANDOM VARIABLES: DISTRIBUTION FUNCTIONS AND THEIR INVERSES

ROUTINE	DESCRIPTION
<b>CSNPR</b>	Evaluates the noncentral chi-squared probability density function.
<b>EXPDF</b>	Evaluates the exponential cumulative distribution function.
<b>EXPIN</b>	Evaluates the inverse of the exponential cumulative distribution function.
<b>EXPPR</b>	Evaluates the exponential probability density function.
<b>EXVDF</b>	Evaluates the extreme value cumulative distribution function.
<b>EXVIN</b>	Evaluates the inverse of the extreme value cumulative distribution function.
<b>EXVPR</b>	Evaluates the extreme value probability density function.
<b>FDf</b>	Evaluates the $F$ cumulative distribution function.
<b>FIN</b>	Evaluates the inverse of the $F$ cumulative distribution function.
<b>FPR</b>	Evaluates the $F$ probability density function.
<b>FNDF</b>	Evaluates the noncentral $F$ cumulative distribution function (CDF).
<b>FNIN</b>	Evaluates the inverse of the noncentral $F$ cumulative distribution function (CDF).
<b>FNPR</b>	Evaluates the noncentral $F$ probability density function.
<b>GAMDF</b>	Evaluates the gamma cumulative distribution function.
<b>GAMIN</b>	Evaluates the inverse of the gamma cumulative distribution function.
<b>GAMPR</b>	Evaluates the gamma probability density function.

## CONTINUOUS RANDOM VARIABLES: DISTRIBUTION FUNCTIONS AND THEIR INVERSES

ROUTINE	DESCRIPTION
<b>RALDF</b>	Evaluates the Rayleigh cumulative distribution function.
<b>RALIN</b>	Evaluates the inverse of the Rayleigh cumulative distribution function.
<b>RALPR</b>	Evaluates the Rayleigh probability density function.
<b>TDF</b>	Evaluates the Student's $t$ cumulative distribution function.
<b>TIN</b>	Evaluates the inverse of the Student's $t$ cumulative distribution function.
<b>TPR</b>	Evaluates the Student's $t$ probability density function.
<b>TNDF</b>	Evaluates the noncentral Student's $t$ cumulative distribution function.
<b>TNIN</b>	Evaluates the inverse of the noncentral Student's $t$ cumulative distribution function.
<b>TNPR</b>	Evaluates the noncentral Student's $t$ probability density function.
<b>UNDF</b>	Evaluates the uniform cumulative distribution function.
<b>UNIN</b>	Evaluates the inverse of the uniform cumulative distribution function.
<b>UNPR</b>	Evaluates the uniform probability density function.
<b>WBLDF</b>	Evaluates the Weibull cumulative distribution function.
<b>WBLIN</b>	Evaluates the inverse of the Weibull cumulative distribution function.
<b>WBLPR</b>	Evaluates the Weibull probability density function.

## GENERAL CONTINUOUS RANDOM VARIABLES

ROUTINE	DESCRIPTION
<b>GCDF</b>	Evaluates a general continuous cumulative distribution function given ordinates of the density.
<b>GCIN</b>	Evaluates the inverse of a general continuous cumulative distribution function given ordinates of the density.
<b>GFNIN</b>	Evaluates the inverse of a general continuous cumulative distribution function given in a subprogram.

## PARAMETER ESTIMATION

ROUTINE	DESCRIPTION
<b>MLE</b>	Calculates maximum likelihood estimates for the parameters of one of several univariate probability distributions.

# CHAPTER 18: RANDOM NUMBER GENERATION

## UTILITY ROUTINES FOR RANDOM NUMBER GENERATORS

ROUTINE	DESCRIPTION
<b>RNOPT</b>	Selects the uniform (0,1) multiplicative congruential pseudorandom number generator.
<b>RNOPG</b>	Retrieves the indicator of the type of uniform random number generator.
<b>RNSET</b>	Initializes a random seed for use in the IMSL random number generators.
<b>RNGET</b>	Retrieves the current value of the seed used in the IMSL random number generators.
<b>RNSES</b>	Initializes the table in the IMSL random number generators that use shuffling.
<b>RNGES</b>	Retrieves the current value of the table in the IMSL random number generators that use shuffling.
<b>RNSEF</b>	Retrieves the array used in the IMSL GFSR random number generator.
<b>RNGEF</b>	Retrieves the current value of the array used in the IMSL GFSR random number generator.

## UTILITY ROUTINES FOR RANDOM NUMBER GENERATORS

ROUTINE	DESCRIPTION
<b>RNISD</b>	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).
<b>RNIN32</b>	Initializes the 32-bit Mersenne Twister generator using an array.
<b>RNGE32</b>	Retrieves the current table used in the 32-bit Mersenne Twister generator.
<b>RNSE32</b>	Sets the current table used in the 32-bit Mersenne Twister generator.
<b>RNIN64</b>	Initializes the 64-bit Mersenne Twister generator using an array.
<b>RNGE64</b>	Retrieves the current table used in the 64-bit Mersenne Twister generator.
<b>RNSE64</b>	Sets the current table used in the 64-bit Mersenne Twister generator.

## BASIC UNIFORM DISTRIBUTION

ROUTINE	DESCRIPTION
<b>RNUN</b>	Generates pseudorandom numbers from a uniform (0, 1) distribution.
<b>RNUNF</b>	Generates a pseudorandom number from a uniform (0, 1) distribution.

## UNIVARIATE DISCRETE DISTRIBUTIONS

ROUTINE	DESCRIPTION
<b>RNBIN</b>	Generates pseudorandom numbers from a binomial distribution.
<b>RNGDA</b>	Generates pseudorandom numbers from a general discrete distribution using an alias method.
<b>RNGDS</b>	Sets up table to generate pseudorandom numbers from a general discrete distribution.
<b>RNGDT</b>	Generates pseudorandom numbers from a general discrete distribution using a table lookup method.

## UNIVARIATE DISCRETE DISTRIBUTIONS

ROUTINE	DESCRIPTION
<b>RNGEO</b>	Generates pseudorandom numbers from a geometric distribution.
<b>RNHYP</b>	Generates pseudorandom numbers from a hypergeometric distribution.
<b>RNLGR</b>	Generates pseudorandom numbers from a logarithmic distribution.
<b>RNNBN</b>	Generates pseudorandom numbers from a negative binomial distribution.
<b>RNPOI</b>	Generates pseudorandom numbers from a Poisson distribution.
<b>RNUND</b>	Generates pseudorandom numbers from a discrete uniform distribution.

## UNIVARIATE CONTINUOUS DISTRIBUTIONS

ROUTINE	DESCRIPTION
<b>RNBET</b>	Generates pseudorandom numbers from a beta distribution.
<b>RNCHI</b>	Generates pseudorandom numbers from a chi-squared distribution.
<b>RNCHY</b>	Generates pseudorandom numbers from a Cauchy distribution.
<b>RNEXP</b>	Generates pseudorandom numbers from a standard exponential distribution.
<b>RNEXV</b>	Generates pseudorandom numbers from an extreme value distribution.
<b>RNFDF</b>	Generates pseudorandom numbers from the $F$ distribution.
<b>RNEXT</b>	Generates pseudorandom numbers from a mixture of two exponential distributions.
<b>RNGAM</b>	Generates pseudorandom numbers from a standard gamma distribution.

## UNIVARIATE CONTINUOUS DISTRIBUTIONS

ROUTINE	DESCRIPTION
<b>RNGCS</b>	Sets up table to generate pseudorandom numbers from a general continuous distribution.
<b>RNGCT</b>	Generates pseudorandom numbers from a general continuous distribution.
<b>RNLNL</b>	Generates pseudorandom numbers from a lognormal distribution.
<b>RNNOA</b>	Generates pseudorandom numbers from a standard normal distribution using an acceptance/rejection method.
<b>RNNOF</b>	Generates a pseudorandom number from a standard normal distribution.
<b>RNNOR</b>	Generates pseudorandom numbers from a standard normal distribution using an inverse CDF method.
<b>RNRAL</b>	Generates pseudorandom numbers from a Rayleigh distribution.
<b>RNSTA</b>	Generates pseudorandom numbers from a stable distribution.
<b>RNSTT</b>	Generates pseudorandom numbers from a Student's $t$ distribution.
<b>RNTRI</b>	Generates pseudorandom numbers from a triangular distribution on the interval $(0, 1)$ .
<b>RNVMS</b>	Generates pseudorandom numbers from a von Mises distribution.
<b>RNWIB</b>	Generates pseudorandom numbers from a Weibull distribution.

## MULTIVARIATE DISTRIBUTIONS

ROUTINE	DESCRIPTION
<b>RNCOR</b>	Generates a pseudorandom orthogonal matrix or a correlation matrix.
<b>RNDAT</b>	Generates pseudorandom numbers from a multivariate distribution determined from a given sample.

## MULTIVARIATE DISTRIBUTIONS

ROUTINE	DESCRIPTION
<b>RNMTN</b>	Generates pseudorandom numbers from a multinomial distribution.
<b>RNMVN</b>	Generates pseudorandom numbers from a multivariate normal distribution.
<b>RNSPH</b>	Generates pseudorandom points on a unit circle or $K$ -dimensional sphere.
<b>RNTAB</b>	Generates a pseudorandom two-way table.
<b>RNMVGC</b>	Given a Cholesky factorization of a correlation matrix, generates pseudorandom numbers from a Gaussian Copula distribution.
<b>RNMVTC</b>	Given a Cholesky factorization of a correlation matrix, generates pseudorandom numbers from a Student's $t$ Copula distribution.
<b>CANCOR</b>	Given an input array of deviate values, generates a canonical correlation array.

## ORDER STATISTICS

ROUTINE	DESCRIPTION
<b>RNNOS</b>	Generates pseudorandom order statistics from a standard normal distribution.
<b>RNUNO</b>	Generates pseudorandom order statistics from a uniform (0, 1) distribution.

## STOCHASTIC PROCESSES

ROUTINE	DESCRIPTION
<b>RNARM</b>	Generates a time series from a specified ARMA model.
<b>RNNPP</b>	Generates pseudorandom numbers from a nonhomogenous Poisson process.



## SAMPLES AND PERMUTATIONS

ROUTINE	DESCRIPTION
<b>RNPER</b>	Generates a pseudorandom permutation.
<b>RNSRI</b>	Generates a simple pseudorandom sample of indices.
<b>RNSRS</b>	Generates a simple pseudorandom sample from a finite population.

## LOW DISCREPANCY SEQUENCES

ROUTINE	DESCRIPTION
<b>FAURE_FREE</b>	Frees the structure containing information about the Faure sequence.
<b>FAURE_INIT</b>	Shuffled Faure sequence initialization.
<b>FAURE_NEXT</b>	Computes a shuffled Faure sequence.

## CHAPTER 19: UTILITIES

### PRINT

ROUTINE	DESCRIPTION
<b>PGOPT</b>	Sets or retrieves page width and length for printing.
<b>WRIRL</b>	Prints an integer rectangular matrix with a given format and labels.
<b>WRIRN</b>	Prints an integer rectangular matrix with integer row and column labels.
<b>WROPT</b>	Sets or retrieves an option for printing a matrix.
<b>WRRRL</b>	Prints a real rectangular matrix with a given format and labels.
<b>WRRRN</b>	Prints a real rectangular matrix with integer row and column labels.

## PERMUTE

ROUTINE	DESCRIPTION
<b>MVNAN</b>	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.
<b>PERMA</b>	Permutates the rows or columns of a matrix.
<b>PERMU</b>	Rearranges the elements of an array as specified by a permutation.
<b>RORDM</b>	Reorders rows and columns of a symmetric matrix.

## SORT

ROUTINE	DESCRIPTION
<b>SCOLR</b>	Sorts columns of a real rectangular matrix using keys in rows.
<b>SROWR</b>	Sorts rows of a real rectangular matrix using keys in columns.
<b>SVIGN</b>	Sorts an integer array by algebraically increasing value.
<b>SVIGP</b>	Sorts an integer array by algebraically increasing value and returns the permutation that rearranges the array.
<b>SVRGN</b>	Sorts a real array by algebraically increasing value.
<b>SVRGP</b>	Sorts a real array by algebraically increasing value and returns the permutation that rearranges the array.

## SEARCH

ROUTINE	DESCRIPTION
<b>ISRCH</b>	Searches a sorted integer vector for a given integer and returns its index.
<b>SRCH</b>	Searches a sorted vector for a given scalar and returns its index.
<b>SSRCH</b>	Searches a character vector, sorted in ascending ASCII order, for a given string and returns its index.

## CHARACTER STRING MANIPULATION

ROUTINE	DESCRIPTION
<b>ACHAR</b>	Returns a character given its ASCII value.
<b>CVTSI</b>	Converts a character string containing an integer number into the corresponding integer form.
<b>IACHAR</b>	Returns the integer ASCII value of a character argument.
<b>ICASE</b>	Returns the ASCII value of a character converted to uppercase.
<b>IICSR</b>	Compares two character strings using the ASCII collating sequence but without regard to case.
<b>IIDEX</b>	Determines the position in a string at which a given character sequence begins without regard to case.

## TIME, DATE AND VERSION

ROUTINE	DESCRIPTION
<b>CPSEC</b>	Returns CPU time used in seconds.
<b>IDYWK</b>	Computes the day of the week for a given date.
<b>NDAYS</b>	Computes the number of days from January 1, 1900, to the given date.
<b>NDYIN</b>	Gives the date corresponding to the number of days since January 1, 1900.
<b>TDATE</b>	Gets today's date.
<b>TIMDY</b>	Gets time of day.
<b>VERSL</b>	Obtains STAT/LIBRARY-related version, system and serial numbers.

## RETRIEVAL OF DATA SETS

ROUTINE

DESCRIPTION

**GDATA**

Retrieves a commonly analyzed data set.

## CHAPTER 20: MATHEMATICAL SUPPORT

### LINEAR SYSTEMS

ROUTINE

DESCRIPTION

**CHFAC**

Cholesky factorization  $R^T R$  of a nonnegative definite matrix.

**GIRTS**

Solves a triangular linear system given  $R$ .

**MCHOL**

Modified Cholesky factorization.

### SPECIAL FUNCTIONS

ROUTINE

DESCRIPTION

**AMILLR**

Mill's ratio.

**ENOS**

Expected value of a normal order statistic.

### NEAREST NEIGHBORS

ROUTINE

DESCRIPTION

**NGHBR**

Searches a  $k$ - $d$  tree for the  $m$  nearest neighbors.

**QUADT**

Forms a  $k$ - $d$  tree.