IMSL®
C Numerical Library
V5.0

Function Catalog

Trusted For Over 30 Years

TABLE OF CONTENTS

IMSL C NUMERICAL LIBRARY

Written in C for C / C++ programmers and based on the world's most widely called numerical subroutines.

The IMSL C Numerical Library ("CNL") is a comprehensive set of over 370 pre-built mathematical and statistical analysis functions that C or C++ programmers can embed directly into their numerical analysis applications. Many of CNL's functions are based upon the same algorithms contained in the company's highly regarded IMSL Fortran 90 MP Library. Visual Numerics, Inc. has been providing algorithms for mathematical and statistical computations under the IMSL name since 1970.

With CNL, we provide "building blocks" which eliminate the need to write code from scratch. These prepackaged functions allow you to apply your industry-specific expertise and reduce your development time.

CNL takes full advantage of the intrinsic characteristics and desirable features of the C language. Variable argument lists simplify calling sequences. The concise set of required arguments contains only information necessary for usage. Optional arguments provide added functionality and power to each function.

CNL is thread safe. Thread safety allows CNL to be used in multithreaded applications ranging from web-based applications to performing advanced data analysis in real time. Which gives you increased throughput, better response time, conservation of system resources and a natural programming structure. Performance benefits can be realized through concurrent and/or parallel execution.

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

You can build applications that are portable across multiple platforms. CNL is available for computer systems running UNIX and Windows operating systems.

Extensive online documentation provides powerful search capabilities with hundreds of code examples.

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

Cost-Effectiveness and Value

CNL significantly shortens program development time and promotes standardization. Variable argument lists have been implemented to simplify calling sequences. You'll find that using CNL saves time in your source code development and saves thousands of dollars in the design, development, documentation, testing and maintenance of your applications.

Accurate, Robust and Reliable

CNL uses descriptive, explanatory function names for intuitive programming. Reserved function names begin with prefixes unique to each product.

Where appropriate, consistent variable names are used to:

- Make function names easy to identify and use and prevent conflicts with other software.
- Provide a common root name for numerical functions that offers the choice of multiple precisions.

Error Handling

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

These error-handling features:

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

Programming Interface Flexibility

CNL takes full advantage of the intrinsic characteristics and desirable features of the C language. The functions support variable-length argument lists. The concise set of required arguments contains only information necessary for usage. Optional arguments provide added functionality and power to each function.

This flexibility:

- Reduces unnecessary code.
- Enables you to adapt each function call by activating optional arguments.

Wide Compatibility and Uniform Operation

The IMSL C Numerical Library is available for UNIX computing environments and Windows 98/2000/NT.

Visual Numerics' commitment to regular feature and enhancement updates:

- Ensures that your software will perform to the highest standards.
- Provides for portable applications.
- Assures that Visual Numerics will keep pace with the latest hardware and software innovations.

Shared Library Technology

The IMSL C Numerical Library is designed to take advantage of shared libraries technology.

This technology:

- Allows more than one user to share information in the library without crowding disk space.
- Provides shorter compile and link time.
- Minimizes the size of executable object modules.

Flexible Licensing Options

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

Comprehensive Documentation

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

IMSL extensive documentation:

- Provides organized, easy-to-find information.
- Documents, explains, and provides references for algorithms.
- Gives at least one example of function usage, with sample input and results.

Unmatched Product Support

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

Product support:

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

MATHEMATICAL FUNCTIONALITY

The IMSL C Numerical Library is a collection of the most commonly needed numerical functions customized for your C programming needs. The mathematical functionality is organized into 10 sections. These capabilities range from solving systems of linear equations to optimization.

- **Linear Systems**, including real and complex full and sparse matrices, linear least squares, matrix decompositions, generalized inverses and vector-matrix operations.

- **Eigensystem Analysis**, including eigenvalues and eigenvectors of complex, real symmetric and complex Hermitian matrices.

- **Interpolation and Approximation**, including constrained curve-fitting splines, cubic splines, least squares approximation and smoothing, and scattered data interpolation.

- **Integration and Differentiation**, including univariate, multivariate and Gauss quadrature.

- **Differential Equations**, using Adams-Gear and Runge-Kutta methods for stiff and nonstiff ordinary differential equations and support for partial differential equations.

- **Transforms**, including real and complex one- and two-dimensional fast Fourier transforms, as well as convolutions and correlations and Laplace transforms.

- **Nonlinear Equations**, including zeros and root finding of polynomials, zeros of a function and root of a system of equations.

- **Optimization**, including unconstrained, and linearly and nonlinearly constrained minimizations.

- **Special Functions**, including error and gamma functions, real order complex valued Bessel functions, statistical functions, and more than fifty functions for financial analysis.

- **Utilities**, including CPU time used, error handling and machine, mathematical, physical constants, retrieval of machine constants, changing error-handling defaults, and performing matrix-matrix multiplication.

STATISTICAL FUNCTIONALITY

The statistical functionality is organized into 12 sections. These capabilities range from analysis of variance to random number generation.

• **Basic Statistics**, including univariate summary statistics, nonparametric tests, such as sign and Wilcoxon rank sum, and goodness-of-fit tests, such as chi-squared and Shapiro-Wilk.

• **Regression**, including stepwise regression, all best regression, multiple linear regression models, polynomial models and nonlinear models.

• **Correlation and Covariance**, 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**, 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, portmanteau lack of fit test and difference of a seasonal or nonseasonal time series.

• **Multivariate Analysis**, including principal components, K-means cluster analysis and factor analysis. Methods of factor analysis include principal components, principal factor, image analysis, unweighted least squares, generalized least squares and maximum likelihood.

• **Survival Analysis**, including analysis of data using a generalized linear model and using various parametric models.

• **Probability Distribution Functions and Inverses**, including binomial, hypergeometric, bivariate normal, gamma and many more.

• **Random Number Generation**, including a generator for multivariate normal distributions and pseudorandom numbers from several distributions, including gamma, Poisson and beta. Also, support for low discrepancy series using a generalized Faure sequence.

IMSL® C Numerical Library Functions

IMSL C/Math/Library™

CHAPTER 1: LINEAR SYSTEMS

LINEAR EQUATIONS WITH FULL MATRICES

lin_sol_gen

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

lin_sol_gen (complex)

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

lin_sol_posdef

Solves a real symmetric positive definite system of linear equations $Ax = b$.

lin_sol_ posdef (complex)

Solves a complex Hermitian positive definite system of linear equations $Ax = b$.

LINEAR EQUATIONS WITH BAND MATRICES

lin_sol_gen_band

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

lin_sol_gen_band (complex)

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

lin_sol_posdef _band

Solves a real symmetric positive definite system of linear equations $Ax = b$ in band symmetric storage mode.

lin_sol_posdef _band (complex)

Solves a complex Hermitian positive definite system of linear equations $Ax = b$ in band symmetric storage mode.

LINEAR EQUATIONS WITH GENERAL SPARSE MATRICES

lin_sol_gen_coordinate

Solves a sparse system of linear equations $Ax = b$.

lin_sol_gen_coordinate (complex)

Solves a sparse Hermitian positive definite system of linear equations $Ax = b$, with sparse complex coefficient matrix A.

lin_sol_posdef_coordinate

Solves a sparse real symmetric positive definite system of linear equations $Ax = b$.

lin_sol_posdef_coordinate (complex)

Solves a sparse Hermitian positive definite system of linear equations $Ax = b$.

ITERATIVE METHODS

lin_sol_gen_min_residual

Solves a linear system $Ax = b$ using the restarted generalized minimum residual (GMRES) method.

lin_sol_def_cg

Solves a real symmetric definite linear system using a conjugate gradient method.

LINEAR LEAST-SQUARES WITH FULL MATRICES

lin_least_squares_gen

Solves a linear least-squares problem $Ax = b$.

lin_lsq_lin_constraints

Solves a linear least squares problem with linear constraints.

lin_svd_gen

Computes the SVD, $A = USV^T$, of a real rectangular matrix A.

lin_svd_gen (complex)

Computes the SVD, $A = USV^H$, of a complex rectangular matrix A.

lin_sol_nonnegdef

Solves a real symmetric nonnegative definite system of linear equations $Ax = b$.

CHAPTER 2: EIGENSYSTEM ANALYSIS

LINEAR EIGENSYSTEM PROBLEMS

eig_gen

Computes the eigenexpansion of a real matrix A.

eig_gen (complex)

Computes the eigenexpansion of a complex matrix A.

eig_sym

Computes the eigenexpansion of a real symmetric matrix A.

eig_herm (complex)

Computes the eigenexpansion of a complex Hermitian matrix A.

GENERALIZED EIGENSYSTEM PROBLEMS

eig_symgen

Computes the generalized eigenexpansion of a system $Ax = \lambda Bx$. A and B are real and symmetric. B is positive definite.

geneig

Computes the generalized eigenexpansion of a system $Ax = \lambda Bx$, with A and B real.

geneig (complex)

Computes the generalized eigenexpansion of a system $Ax = \lambda Bx$, with A and B complex.

CHAPTER 3: INTERPOLATION AND APPROXIMATION

CUBIC SPLINE INTERPOLATION

cub_spline_interp_e_cnd

Computes a cubic spline interpolant, specifying various endpoint conditions.

cub_spline_interp_shape

Computes a shape-preserving cubic spline.

CUBIC SPLINE EVALUATION AND INTEGRATION

cub_spline_value

Computes the value of a cubic spline or the value of one of its derivatives.

cub_spline_integral

Computes the integral of a cubic spline.

SPLINE INTERPOLATION

spline_interp

Computes a spline interpolant.

spline_knots

Computes the knots for a spline interpolant.

spline_2d_interp

Computes a two-dimensional, tensor-product spline interpolant from two-dimensional, tensor-product data.

SPLINE EVALUATION AND INTEGRATION

spline_value

Computes the value of a spline or the value of one of its derivatives.

spline_integral

Computes the integral of a spline.

spline_2d_value

Computes the value of a tensor-product spline or the value of one of its partial derivatives.

spline_2d_integral

Evaluates the integral of a tensor-product spline on a rectangular domain.

LEAST-SQUARES APPROXIMATION AND SMOOTHING

user_fcn_least_squares

Computes a least-squares fit using user-supplied functions.

spline_least_squares

Computes a least-squares spline approximation.

spline_2d_least_squares

Computes a two-dimensional, tensor-product spline approximant using least squares.

cub_spline_smooth

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

spline_lsq_constrained

Computes a least-squares constrained spline approximation.

smooth_1d_data

Smooth one-dimensional data by error detection.

SCATTERED DATA INTERPOLATION

scattered_2d_interp

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

SCATTERED DATA LEAST SQUARES

radial_scattered_fit

Computes an approximation to scattered data in $\mathbf{R}^n$ for $n \geq 2$ using radial basis functions.

radial_evaluate

Evaluates a radial basis fit.

CHAPTER 4: QUADRATURE

UNIVARIATE QUADRATURE

int_fcn_sing

Integrates a function, which may have endpoint singularities, using a globally adaptive scheme based on Gauss-Kronrod rules.

int_fcn

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

int_fcn_sing_pts

Integrates a function with singularity points given.

int_fcn_alg_log

Integrates a function with algebraic-logarithmic singularities.

int_fcn_inf

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

int_fcn_trig

Integrates a function containing a sine or a cosine factor.

int_fcn_fourier

Computes a Fourier sine or cosine transform.

int_fcn_cauchy

Computes integrals of the form $\int_a^b \frac{f(x)}{x-c} dx$ in the Cauchy principal value sense.

int_fcn_smooth

Integrates a smooth function using a nonadaptive rule.

MULTIVARIATE QUADRATURE

int_fcn_2d

Computes a two-dimensional iterated integral.

int_fcn_hyper_rect

Integrates a function on a hyper-rectangle

$$\int_{a_0}^{b_0} \cdots \int_{a_{n-1}}^{b_{n-1}} f(x_0, \ldots, x_{n-1})\, dx_{n-1} \ldots dx_0.$$

int_fcn_qmc

Integrates a function on a hyper-rectangle using a quasi-Monte-Carlo method.

GAUSS QUADRATURE

gauss_quad_rule

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

DIFFERENTIATION

fcn_derivative

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

CHAPTER 5: DIFFERENTIAL EQUATIONS

RUNGE-KUTTA METHOD

ode_runge_kutta

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

ADAM'S OR GEAR'S METHOD

ode_adams_gear

Solves a stiff initial-value problem for ordinary differential equations using the Adams-Gear methods.

METHOD OF LINES

pde_method_of_lines

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.

FAST POISSON SOLVER

fast_poisson_2d

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.

CHAPTER 6: TRANSFORMS

REAL TRIGONOMETRIC FFTS

fft_real

Computes the real discrete Fourier transform of a real sequence.

fft_real_init

Computes the parameters for imsl_f_fft_real.

COMPLEX EXPONENTIAL FFTS

fft_complex

Computes the complex discrete Fourier transform of a complex sequence.

fft_complex_init

Computes the parameters for imsl_c_fft_complex.

REAL SINE AND COSINE FFTS

fft_cosine

Computes the discrete Fourier cosine transformation of an even sequence.

fft_cosine_init

Computes the parameters needed for imsl_f_fft_cosine.

fft_sine

Computes the discrete Fourier sine transformation of an odd sequence.

fft_sine_init

Computes the parameters needed for imsl_f_fft_sine.

TWO-DIMENSIONAL FFTS

fft_2d_complex

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

CONVOLUTION AND CORRELATION

convolution

Computes the convolution, and optionally, the correlation of two real vectors.

convolution (complex)

Computes the convolution, and optionally, the correlation of two complex vectors.

LAPLACE TRANSFORM

inverse_laplace

Computes the inverse Laplace transform of a complex function.

CHAPTER 7: NONLINEAR EQUATIONS

ZEROS OF A POLYNOMIAL

zeros_poly

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

zeros_poly (complex)

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

ZEROS OF A FUNCTION

zeros_fcn

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

ROOT OF A SYSTEM OF EQUATIONS

zeros_sys_eqn

Solves a system of n nonlinear equations $f(x) = 0$ using a modified Powell hybrid algorithm.

CHAPTER 8: OPTIMIZATION

UNCONSTRAINED MINIMIZATION

min_uncon

Finds the minimum point of a smooth function $f(x)$ of a single variable using only function evaluations.

min_uncon_deriv

Finds the minimum point of a smooth function $f(x)$ of a single variable using both function and first derivative evaluations.

min_uncon_multivar

Minimizes a function $f(x)$ of n variables using a quasi-Newton method.

nonlin_least_squares

Solves a nonlinear least-squares problem using a modified Levenberg-Marquardt algorithm.

LINEARLY CONSTRAINED MINIMIZATION

lin_prog

Solves a linear programming problem using the revised simplex algorithm.

quadratic_prog

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

min_con_gen_lin

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

bounded_least_squares

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

NONLINEARLY UNCONSTRAINED MINIMIZATION

min_con_nonlin

Solves a general nonlinear programming problem using the successive quadratic programming algorithm.

CHAPTER 9: SPECIAL FUNCTIONS

ERROR AND GAMMA FUNCTIONS

erf

Evaluates the real error function $\mathrm{erf}(x)$.

erfc

Evaluates the real complementary error function $\mathrm{erfc}(x)$.

erf_inverse

Evaluates the real inverse error function $\mathrm{erf}^{-1}(x)$.

erfc_inverse

Evaluates the real inverse complementary error function $\mathrm{erfc}^{-1}(x)$.

beta

Evaluates the real beta function $\beta(x, y)$.

log_beta

Evaluates the logarithm of the real beta function $\ln \beta(x, y)$.

beta_incomplete

Evaluates the real incomplete beta function $I_x = \beta_x(a,b)/\beta(a,b)$.

gamma

Evaluates the real gamma function $\Gamma(x)$.

log_gamma

Evaluates the logarithm of the absolute value of the gamma function $\log|\Gamma(x)|$.

gamma_incomplete

Evaluates the incomplete gamma function $\gamma(a, x)$.

BESSEL FUNCTIONS

bessel_J0

Evaluates the real Bessel function of the first kind of order zero $J_0(x)$.

bessel_J1

Evaluates the real Bessel function of the first kind of order one $J_1(x)$.

bessel_Jx

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

bessel_Y0

Evaluates the real Bessel function of the second kind of order zero $Y_0(x)$.

bessel_Y1

Evaluates the real Bessel function of the second kind of order one $Y_1(x)$.

bessel_Yx

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

bessel_I0

Evaluates the real modified Bessel function of the first kind of order zero $I_0(x)$.

bessel_exp_I0

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

bessel_I1

Evaluates the real modified Bessel function of the first kind of order one $I_1(x)$.

bessel_exp_I1

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

bessel_Ix

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

bessel_K0

Evaluates the real modified Bessel function of the third kind of order zero $K_0(x)$.

bessel_exp_K0

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

bessel_K1

Evaluates the real modified Bessel function of the third kind of order one $K_1(x)$.

bessel_exp_K1

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

bessel_Kx

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

ELLIPTIC INTEGRALS

elliptic_integral_K

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

elliptic_integral_E

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

elliptic_integral_RF

Evaluates Carlson's elliptic integral of the first kind $R_F(x, y, z)$.

elliptic_integral_RD

Evaluates Carlson's elliptic integral of the second kind $R_D(x, y, z)$.

elliptic_integral_RJ

Evaluates Carlson's elliptic integral of the third kind $R_J(x, y, z, \rho)$.

elliptic_integral_RC

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

FRESNEL INTEGRALS

fresnel_integral_C

Evaluates the cosine Fresnel integral.

fresnel_integral_S

Evaluates the sine Fresnel integral.

AIRY FUNCTIONS

airy_Ai

Evaluates the Airy function.

airy_Bi

Evaluates the Airy function of the second kind.

airy_Ai_derivative

Evaluates the derivative of the Airy function.

airy_Bi_derivative

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

KELVIN FUNCTIONS

kelvin_ber0

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

kelvin_bei0

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

kelvin_ker0

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

kelvin_kei0

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

kelvin_ber0_derivative

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

kelvin_bei0_derivative

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

kelvin_ker0_derivative

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

kelvin_kei0_derivative

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

STATISTICAL FUNCTIONS

normal_cdf

Evaluates the standard normal (Gaussian) distribution function.

normal_inverse_cdf

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

chi_squared_cdf

Evaluates the chi-squared distribution function.

chi_squared_inverse_cdf

Evaluates the inverse of the chi-squared distribution function.

F_cdf

Evaluates the *F* distribution function.

F_inverse_cdf

Evaluates the inverse of the *F* distribution function.

t_cdf

Evaluates the Student's *t* distribution function.

t_inverse_cdf

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

gamma_cdf

Evaluates the gamma distribution function.

binomial_cdf

Evaluates the binomial distribution function.

hypergeometric_cdf

Evaluates the hypergeometric distribution function.

poisson_cdf

Evaluates the Poisson distribution function.

beta_cdf

Evaluates the beta probability distribution function.

beta_inverse_cdf

Evaluates the inverse of the beta distribution function.

bivariate_normal_cdf

Evaluates the bivariate normal distribution function.

FINANCIAL FUNCTIONS

cumulative_interest

Evaluates the cumulative interest paid between two periods.

cumulative_principal

Evaluates the cumulative principal paid between two periods.

depreciation_db

Evaluates the depreciation of an asset. (Fixed-declining-balance method).

depreciation_ddb

Evaluates the depreciation of an asset. (Double-declining-balance method).

depreciation_sln

Evaluates the depreciation of an asset. (Straight-line method).

depreciation_syd

Evaluates the depreciation of an asset. (Sum-of-years digits method).

depreciation_vdb

Evaluates the depreciation of an asset for any given period, including partial periods. (Double-declining-balance method).

dollar_decimal

Converts a fractional price to a decimal price.

dollar_fraction

Converts a decimal price to a fractional price.

effective_rate

Evaluates the effective annual interest rate.

future_value

Evaluates an investment's future value.

future_value_schedule

Evaluates the future value of an initial principal taking into consideration a schedule of compound interest rates.

interest_payment

Evaluates the interest payment for an investment for a given period.

interest_rate_annuity

Evaluates an annuity's interest rate per period.

internal_rate_of_return

Evaluates the internal rate of return for a schedule of cash flows.

internal_rate_schedule

Evaluates the internal rate of return for a schedule of cash flows. It is not necessary that the cash flows be periodic.

modified_internal_rate

Evaluates the modified internal rate of return for a schedule of periodic cash flows.

net_present_value

Evaluates an investment's net present value. The calculation is based on a schedule of periodic cash flows and a discount rate.

nominal_rate

Evaluates the nominal annual interest rate.

number_of_periods

Evaluates the number of periods for an investment for which periodic and constant payments are made and the interest rate is constant.

payment

Evaluates the periodic payment for an investment.

present_value

Evaluates an investment's present value.

present_value_schedule

Evaluates the present value for a schedule of cash flows. It is not necessary that the cash flows be periodic.

principal_payment

Evaluates the payment on the principal for a specified period.

BOND FUNCTIONS

accr_interest_maturity

Evaluates the interest that has accrued on a security, which pays interest at maturity.

accr_interest_periodic

Evaluates the interest that has accrued on a security, which pays interest periodically.

bond_equivalent_yield

Evaluates a Treasury bill's bond-equivalent yield.

convexity

Evaluates the convexity for a security.

coupon_days

Evaluates the number of days in the coupon period containing the settlement date.

coupon_number

Evaluates the number of coupons payable between the settlement date and the maturity date.

days_before_settlement

Evaluates the number of days starting with the beginning of the coupon period and ending with the settlement date.

days_to_next_coupon

Evaluates the number of days starting with the settlement date and ending with the next coupon date.

depreciation_amordegrc

Evaluates the depreciation for each accounting period. During the evaluation of the function a depreciation coefficient based on the asset life is applied.

depreciation_amorlinc

Evaluates the depreciation for each accounting period. This function is similar to depreciation_amordegrc, except that depreciation_amordegrc has a depreciation coefficient that is applied during the evaluation that is based on the asset life.

discount_price

Evaluates a discounted security's price per $100 face value.

discount_rate

Evaluates a security's discount rate.

discount_yield

Evaluates a discounted security's annual yield.

duration

Evaluates a security's annual duration where the security has periodic interest payments.

interest_rate_security

Evaluates a fully invested security's interest rate.

modifiedmacauley_duration

Evaluates a security's modified Macauley duration. The security has an assumed par value of $100.

next_coupon_date

Evaluates the first coupon date that follows the settlement date.

previous_coupon_date

Evaluates the coupon date that immediately precedes the settlement date.

price

Evaluates a security's price per $100 face value. The security pays periodic interest.

price_maturity

Evaluates a security's price per $100 face value. The security pays interest at maturity.

received_maturity

Evaluates the amount one receives when a fully invested security reaches the maturity date.

treasury_bill_price

Evaluates a Treasury bill's price per $100 face value.

treasury_bill_yield

Evaluates a Treasury bill's yield.

year_fraction

Evaluates the fraction of a year represented by the number of whole days between two dates.

yield_maturity

Evaluates a security's annual yield. The security pays interest at maturity.

yield_periodic

Evaluates a security's yield. The security pays periodic interest.

CHAPTER 10: STATISTICS AND RANDOM NUMBER GENERATION

STATISTICS

simple_statistics

Computes basic univariate statistics.

table_oneway

Tallies observations into a one-way frequency table.

chi_squared_test

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

covariances

Computes the sample variance-covariance or correlation matrix.

regression

Fits a multiple linear regression model using least squares.

poly_regression

Performs a polynomial least-squares regression.

ranks

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

RANDOM NUMBERS

random_seed_get

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

random_seed_set

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

random_option

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

random_uniform

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

random_normal

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

random_poisson

Generates pseudorandom numbers from a Poisson distribution.

random_gamma

Generates pseudorandom numbers from a standard gamma distribution.

random_beta

Generates pseudorandom numbers from a beta distribution.

random_exponential

Generates pseudorandom numbers from a standard exponential distribution.

faure_next_point

Computes a shuffled Faure sequence.

CHAPTER 11: PRINTING FUNCTIONS

PRINT

write_matrix

Prints a rectangular matrix (or vector) stored in contiguous memory locations.

SET

page

Sets or retrieve the page width or length.

write_options

Sets or retrieve an option for printing a matrix.

CHAPTER 12: UTILITIES

SET OUTPUT FILES

output_file

Sets the output file or the error message output file.

version

Returns integer information describing the version of the library, license number, operating system, and compiler.

TIME AND DATE:

ctime

Returns the number of CPU seconds used.

date_to_days

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

days_to_date

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

ERROR HANDLING

error_options

Sets various error handling options.

error_code

Gets the code corresponding to the error message from the last function called.

CONSTANTS

constant

Returns the value of various mathematical and physical constants.

machine (integer)

Returns integer information describing the computer's arithmetic.

machine (float)

Returns information describing the computer's floating-point arithmetic.

SORTING

sort

Sorts a vector by algebraic value. Optionally, a vector can be sorted by absolute value, and a sort permutation can be returned.

sort (integer)

Sorts an integer vector by algebraic value. Optionally, a vector can be sorted by absolute value, and a sort permutation can be returned.

COMPUTING VECTOR NORMS

vector_norm

Computes various norms of a vector or the difference of two vectors.

LINEAR ALGEBRA SUPPORT

mat_mul_rect

Computes the transpose of a matrix, a matrix-vector product, a matrix-matrix product, the bilinear form, or any triple product.

mat_mul_rect (complex)

Computes the transpose of a matrix, the conjugate-transpose of a matrix, a matrix-vector product, a matrix-matrix product, the bilinear form, or any triple product.

mat_mul_rect_band

Computes the transpose of a matrix, a matrix-vector product, or a matrix-matrix product, all matrices stored in band form.

mat_mul_rect_band (complex)

Computes the transpose of a matrix, a matrix-vector product, or a matrix-matrix product, all matrices of complex type and stored in band form.

mat_mul_rect_coordinate

Computes the transpose of a matrix, a matrix-vector product, or a matrix-matrix product, all matrices stored in sparse coordinate form.

mat_mul_rect_coordinate (complex)

Computes the transpose of a matrix, a matrix-vector product or a matrix-matrix product, all matrices stored in sparse coordinate form.

mat_add_band

Adds two band matrices, both in band storage mode, $C \leftarrow \alpha A + \beta B$.

mat_add_band (complex)

Adds two band complex matrices, both in band storage mode, $C \leftarrow \alpha A + \beta B$.

mat_add_coordinate

Performs element-wise addition of two real matrices stored in coordinate format, $C \leftarrow \alpha A + \beta B$.

mat_add_coordinate (complex)

Performs element-wise addition on two complex matrices stored in coordinate format, $C \leftarrow \alpha A + \beta B$.

matrix_norm

Computes various norms of a rectangular matrix.

matrix_norm_band

Computes various norms of a matrix stored in band storage mode.

matrix_norm_coordinate

Computes various norms of a matrix stored in coordinate format.

generate_test_band

Generates test matrices of class $E(n, c)$.

generate_test_band (complex)

Generates test matrices of class $E_c(n, c)$.

generate_test_coordinate

Generates test matrices of class $D(n, c)$ and $E(n, c)$.

generate_test_coordinate (complex)

Generates test matrices of class $D(n, c)$ and $E(n, c)$.

Numeric Utilities

c_neg

Changes the sign of a complex number.

c_add

Adds two complex numbers.

c_sub

Subtracts a complex number from a complex number.

c_mul

Multiplies two complex numbers.

c_div

Divides a complex number by a complex number.

c_eq

Tests for equality of two complex numbers.

cz_convert

Truncates a double precision complex number to a single precision complex number.

zc_convert

Increases precision of a single precision complex number to a double precision complex number.

cf_convert

Makes a complex number from an ordered pair.

c_conjg

Conjungates a complex number.

c_abs

Computes a magnitude of a complex number.

c_arg

Computes an angle of a complex number.

c_sqrt

Computes a square root of a complex number.

c_cos

Computes a trigonometric cosine of a complex number.

c_sin

Computes a trigonometric sine of a complex number.

c_exp

Computes an exponential function of a complex number.

c_log

Computes a natural logarithm of a complex number.

cf_power

Computes a complex number raised to a real power.

cc_power

Computes a complex number raised to a complex power.

fi_power

Computes a real number raised to an integral power.

ii_power

Computes an integer raised to an integral power.

IMSL C/Stat/Library ™

CHAPTER 1: BASIC STATISTICS

SIMPLE SUMMARY STATISTICS

simple_statistics

Computes basic univariate statistics.

normal_one_sample

Computes statistics for mean and variance inferences using a sample from a normal population.

normal_two_sample

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

TABULATE, SORT, RANK

table_oneway

Tallies observations into a one-way frequency table.

table_twoway

Tallies observations into a two-way frequency table.

sort_data

Sorts observations by specified keys, with option to tally cases into a multi-way frequency table.

ranks

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

CHAPTER 2: REGRESSION

MULTIVARIATE LINEAR REGRESSION—MODEL FITTING

regressors_for_glm

Generates regressors for a general linear model.

regression

Fits a multiple linear regression model using least squares.

MULTIVARIATE LINEAR REGRESSION—STATISTICAL INFERENCE AND DIAGNOSTICS

regression_summary

Produces summary statistics for a regression model given the information from the fit.

regression_prediction

Computes predicted values, confidence intervals, and diagnostics after fitting a regression model.

hypothesis_partial

Constructs a completely testable hypothesis.

hypothesis_scph

Sums of cross products for a multivariate hypothesis.

hypothesis_test

Tests for the multivariate linear hypothesis.

VARIABLE SELECTION

regression_selection

Selects the best multiple linear regression models.

regression_stepwise

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

POLYNOMIAL AND NONLINEAR REGRESSION

poly_regression

Performs a polynomial least-squares regression.

poly_prediction

Computes predicted values, confidence intervals, and diagnostics after fitting a polynomial regression model.

nonlinear_regression

Fits a nonlinear regression model.

nonlinear_optimization

Fits a nonlinear regression model using Powell's algorithm.

ALTERNATIVES TO LEAST SQUARES

Lnorm_regression

Fits a multiple linear regression model using L_p criteria other than least squares.

CHAPTER 3: CORRELATION AND COVARIANCE

VARIANCES, COVARIANCES, AND CORRELATIONS

covariances

Computes the sample variance-covariance or correlation matrix.

partial_covariances

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

pooled_covariances

Computes a pooled variance-covariance from the observations.

robust_covariances

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

CHAPTER 4: ANALYSIS OF VARIANCE

ONE-WAY CLASSIFICATION

anova_oneway

Analyzes a one-way classification model.

BALANCED DESIGNS

anova_factorial

Analyzes a balanced factorial design with fixed effects.

multiple_comparisons

Performs Student-Newman-Keuls multiple comparisons test.

NESTED DESIGNS

anova_nested

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

MULTIPLE COMPARISONS TEST

anova_balanced

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

CHAPTER 5: CATEGORICAL AND DISCRETE DATA ANALYSIS

STATISTICS IN THE TWO-WAY CONTINGENCY TABLE

contingency_table

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

exact_enumeration

Computes exact probabilities in a two-way contingency table, using the total enumeration method.

exact_network

Computes exact probabilities in a two-way contingency table using the network algorithm.

GENERALIZED CATEGORICAL MODELS

categorical_glm

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

CHAPTER 6: NONPARAMETRIC STATISTICS

ONE SAMPLE TESTS—NONPARAMETRIC STATISTICS

sign_test

Performs a sign test.

wilcoxon_sign_rank

Performs a Wilcoxon rank sum test.

noether_cyclical_trend

Performs the Noether's test for cyclical trend.

cox_stuart_trends_test

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

tie_statistics

Computes tie statistics for a sample of observations.

TWO OR MORE SAMPLES

wilcoxon_rank_sum

Performs a Wilcoxon rank sum test.

kruskal_wallis_test

Performs a Kruskal-Wallis's test for identical population medians.

friedmans_test

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

cochran_q_test

Performs Cochran's *Q* test for related observations.

k_trends_test

Performs k-sample trends test against ordered alternatives.

CHAPTER 7: TESTS OF GOODNESS OF FIT

GENERAL GOODNESS-OF-FIT-TESTS

chi_squared_test

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

normality_test

Performs a test for normality.

kolmogorov_one

Performs a Kolmogorov-Smirnov's one-sample test for continuos distributions.

kolmogorov_two

Performs a Kolmogorov-Smirnov's two-sample test

multivar_normality_test

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

TESTS FOR RANDOMNESS

randomness_test

Performs a test for randomness.

CHAPTER 8: TIME SERIES AND FORECASTING

ARIMA MODELS

arma

Computes least-square estimates of parameters for an ARMA model.

arma_forecast

Computes forecasts and their associated probability limits for an ARMA model.

difference

Differences a seasonal or nonseasonal time series.

box_cox_transform

Performs a Box-Cox transformation.

autocorrelation

Computes the sample autocorrelation function of a stationary time series.

partial_autocorrelation

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

lack_of_fit

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

garch

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

kalman

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

CHAPTER 9: MULTIVARIATE ANALYSIS

CLUSTER ANALYSIS

cluster_k_means

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

PRINCIPAL COMPONENTS

principal_components

Computes principal components.

FACTOR ANALYSIS

factor_analysis

Extracts initial factor-loading estimates in factor analysis.

DISCRIMINANT ANALYSIS

discriminant_analysis

Performs discriminant function analysis.

CHAPTER 10: SURVIVAL ANALYSIS

GENERALIZED LINEAR MODEL

survival_glm

Analyzes survival data using a generalized linear model.

PARAMETRIC ESTIMATES

survival_estimates

Estimates using various parametric models.

CHAPTER 11: PROBABILITY AND DISTRIBUTION FUNCTIONS

DISCRETE RANDOM VARIABLES

binomial_cdf

Evaluates the binomial distribution function.

binomial_pdf

Evaluates the binomial probability function.

hypergeometric_cdf

Evaluates the hypergeometric distribution function.

poisson_cdf

Evaluates the Poisson distribution function.

CONTINUOUS RANDOM VARIABLES

beta_cdf

Evaluates the beta probability distribution function.

beta_inverse_cdf

Evaluates the inverse of the beta distribution function.

bivariate_normal_cdf

Evaluates the bivariate normal distribution function.

bivariate_normal_cdf

Evaluates the bivariate normal distribution function.

bivariate_normal_cdf

Evaluates the bivariate normal distribution function.

chi_squared_cdf

Evaluates the chi-squared distribution function.

chi_squared_inverse_cdf

Evaluates the inverse of the chi-squared distribution function.

non_central_chi_sq

Evaluates the noncentral chi-squared distribution function.

non_central_chi_sq_inv

Evaluates the inverse of the noncentral chi-squared function.

F_cdf

Evaluates the *F* distribution function.

F_inverse_cdf

Evaluates the inverse of the *F* distribution function.

gamma_cdf

Evaluates the gamma distribution function.

normal_cdf

Evaluates the standard normal (Gaussian) distribution function.

normal_inverse_cdf

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

t_cdf

Evaluates the Student's *t* distribution function.

t_inverse_cdf

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

non_central_t_cdf

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

non_central_t_inv_cdf

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

CHAPTER 12: RANDOM NUMBER GENERATION

UNIVARIATE DISCRETE DISTRIBUTIONS

random_binomial

Generates pseudorandom binomial numbers.

random_geometric

Generates pseudorandom numbers from a geometric distribution.

random_hypergeometric

Generates pseudorandom numbers from a hypergeometric distribution.

random_logarithmic

Generates pseudorandom numbers from a logarithmic distribution.

random_neg_binomial

Generates pseudorandom numbers from a negative binomial distribution.

random_poisson

Generates pseudorandom numbers from a Poisson distribution.

random_uniform_discrete

Generates pseudorandom numbers from a discrete uniform distribution.

random_general_discrete

Generates pseudorandom numbers from a general discrete distribution using an alias method or optionally a table lookup method.

discrete_table_setup

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

UNIVARIATE CONTINUOUS DISTRIBUTIONS

random_beta

Generates pseudorandom numbers from a beta distribution.

random_cauchy

Generates pseudorandom numbers from a Cauchy distribution.

random_chi_squared

Generates pseudorandom numbers from a chi-squared distribution.

random_exponential

Generates pseudorandom numbers from a standard exponential distribution.

random_exponential_mix

Generates pseudorandom mixed numbers from a standard exponential distribution.

random_gamma

Generates pseudorandom numbers from a standard gamma distribution.

random_lognormal

Generates pseudorandom numbers from a lognormal distribution.

random_normal

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

random_stable

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

random_student_t

Generates pseudorandom Student's *t*.

random_triangular

Generates pseudorandom numbers from a triangular distribution.

random_uniform

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

random_von_mises

Generates pseudorandom numbers from a von Mises distribution.

random_weibull

Generates pseudorandom numbers from a Weibull distribution.

random_general_continuous

Generates pseudorandom numbers from a general continuous distribution.

continuous_table_setup

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

MULTIVARIATE CONTINUOUS DISTRIBUTIONS

random_normal_multivariate

Generates pseudorandom numbers from a multivariate normal distribution.

random_orthogonal_matrix

Generates a pseudorandom orthogonal matrix or a correlation matrix.

random_mvar_from_data

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

random_multinomial

Generates pseudorandom numbers from a multinomial distribution.

random_sphere

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

random_table_twoway

Generates a pseudorandom two-way table.

ORDER STATISTICS

random_order_normal

Generates pseudorandom order statistics from a standard normal distribution.

random_order_uniform

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

STOCHASTIC PROCESSES

random_arma

Generates pseudorandom ARMA process numbers.

random_npp

Generates pseudorandom numbers from a nonhomogeneous Poisson process.

SAMPLES AND PERMUTATIONS

random_permutation

Generates a pseudorandom permutation.

random_sample_indices

Generates a simple pseudorandom sample of indices.

random_sample

Generates a simple pseudorandom sample from a finite population.

UTILITY FUNCTIONS

random_option

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

random_option_get

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

random_seed_get

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

random_substream_seed_get

Retrieves a seed for the congruential generators that do not do shuffling that will generate random numbers beginning 100,000 numbers farther along.

random_seed_set

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

random_table_set

Sets the current table used in the shuffled generator.

random_table_get

Retrieves the current table used in the shuffled generator.

random_GFSR_table_set

Sets the current table used in the GFSR generator.

random_GFSR_table_get

Retrieves the current table used in the GFSR generator.

LOW-DISCREPANCY SEQUENCE

faure_next_point

Computes a shuffled Faure sequence

CHAPTER 13: PRINTING FUNCTIONS

PRINT

write_matrix

Prints a rectangular matrix (or vector) stored in contiguous memory locations.

SET

page

Sets or retrieves the page width or length.

write_options

Sets or retrieves an option for printing a matrix.

CHAPTER 14: UTILITIES

SET OUTPUT FILES

output_file

Sets the output file or the error message output file.

version

Returns integer information describing the version of the library, license number, operating system, and compiler.

ERROR HANDLING

error_options

Sets various error handling options.

error_code

Returns the code corresponding to the error message from the last function called.

CONSTANTS

machine (integer)

Returns integer information describing the computer's arithmetic.

machine (float)

Returns information describing the computer's floating-point arithmetic.

data_sets

Retrieves a commonly analyzed data set.

MATHEMATICAL SUPPORT

mat_mul_rect

Computes the transpose of a matrix, a matrix-vector product, a matrix-matrix product, a bilinear form, or any triple product.

permute_vector

Rearranges the elements of a vector as specified by a permutation.

permute_matrix

Permutes the rows or columns of a matrix.

binomial_coefficient

Evaluates the binomial coefficient.

beta

Evaluates the complete beta function.

beta_incomplete

Evaluates the real incomplete beta function.

og_beta

Evaluates the log of the real beta function.

gamma

Evaluates the real gamma functions.

gamma_incomplete

Evaluates the incomplete gamma function.

log_gamma

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

ctime

Returns the number of CPU seconds used.

IMSL FORTRAN 90 MP LIBRARY

The IMSL Fortran 90 MP Library ("F90MP") is used by technical professionals for business, engineering, finance, research and education applications worldwide. Users can leverage the high performance technology of vector processors, shared memory parallelism and/or distributed memory parallelism and at the same time continue to obtain reliable results from the mathematical and statistical functionality within this library.

F90MP provides Fortran 90 coded, not translated, subroutines. Our implementation complies with the ANSI and ISO standards. The incorporation of array syntax provides performance gains with optimized compilers. The object-oriented interface provides easy access to the underlying algorithms.

F90MP also provides backward compatibility for developers who have used our FORTRAN 77 functions; all of the FORTRAN 77 function definitions are included in the IMSL Fortran 90 MP Library.

JNL - A NUMERICAL LIBRARY FOR JAVA™

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

JNL is an object-oriented implementation of several important classes of mathematical functions drawn from the IMSL algorithm repository. Visual Numerics has taken individual algorithms and reimplemented them as object-oriented, Java methods. JNL is designed with extensibility in mind; new classes may be derived from existing ones to add functionality to satisfy particular requirements.

Because JNL is a 100% Pure Java class library, it can be deployed on any platform that supports Java. A JNL-based application will work seamlessly on a PC, a Macintosh, a UNIX workstation or any other Java-enabled platform.

JNL can be used to write client-side applets, server-side applications or even non-networked desktop applications. JNL applets perform all processing on the Java client, whether it is a thin client, such as a network computer, a PC or workstation equipped with a Java Virtual Machine. Client-side processing reduces the number of "round trips" to a networked server, which in turn minimizes network traffic and system latency. JNL is Visual Numerics' contribution to the worldwide Java development community, and is available free of charge via our website.

C Numerical Library Version 5.0

With over *75* new functions!

• 50 new functions in the area of finance and bonds, intended to help brokerage firms or insurance companies perform financial modeling to analyze various risk factors associated with their business. We've also included routines for calculating depreciation of assets, internal rates of return, bond amortization, and net present values.

• A time series routine useful in navigation, surveying, vehicle tracking (aircraft, space-craft, missiles), geology, oceanography, fluid dynamics, steel/paper/power industries, and demographic estimation.

• Routines to compute low discrepancy series of random points using a generalized Faure sequence.

• An algorithm for efficient multi-dimensional quadrature. Very high-dimension quadrature arises in some financial optimization applications. Such integrals are used in evaluating collateralized mortgage obligations (CMOs). Valuing a 30-year (360 month) CMO requires that an integral over a 360 dimensions box be evaluated.

• More than twenty new random number routines, including Generalized Feedback Shift Register (GFSR) generator support.

P/N 7333

Platform & System Requirements

Platform	Operating System	Compiler
Silicon Graphics 64 bit	IRIX 6.5	MIPSpro C, Version 7.3.1.1m
Compaq Alpha	TRU64 UNIX, Version 5.1-732	Compaq C V6.3-025
Fujitsu 300 / 500 / 700 / SX	VPP5000UU UPX/V V L	UXP/V C V20L20 Driver L99121
HP PA/RISC 1.1 32 bit	HP-UX Release 11.0	HP92453-01 A.11.01.20 HP C Compiler
HP PA/RISC 2.0 64 bit	HP-UX Release 11.0	HP92453-01 A.11.01.20 HP C Compiler
Intel Based systems	Red Hat Linux 6.1	egcs-2.91.66
Intel Based systems	Windows 98/NT/2000	Visual C++ 6.0
IBM RS/6000 32-bit	AIX, Version 4.3.3	C and C++ Compilers Version 3.6.6.0 & C for AIX Compiler Version 5.0.0.0
IBM RS/6000 64-bit	AIX, Version 4.3.3	C and C++ Compilers Version 3.6.6.0 & C for AIX Compiler Version 5.0.0.0
Sun SPARC 32-bit	Sun Solaris, Version 7.0	Workshop Compiler C 5.1
Sun SPARC 64-bit	Sun Solaris, Version 7.0	Workshop Compiler C 5.1

DISCOVER why **IMSL** has been trusted for over 30 years!

Contact your sales representative today for information on how we can help you.

Visual Numerics 30TH ANNIVERSARY

[www.vni.com]