NMC Numerical Mathematics Consortium

Technical Scope

When providing guidance to the development of mathematical applications, and more specifically, numeric algorithm development based on well-defined mathematical principles, the goals defining the scope of those recommendations are similar in most respects. Several attempts to specify math functionality over the last fifteen years lack universal backing. In industry today, the need for advanced mathematics in everything from automobiles to wireless networking is growing rapidly. As a result, core math functionality is less of a differentiator from a vendor marketing perspective and more of an educational burden to the end-user.

As a first step, the consortium is enumerating the fundamental set of math functions and categorizing them for presentation as well as consumption. These functions are widely used in a broad range of application domains that require numerically savvy mathematical algorithms. Despite their common usage, subtle differences exist in both function arguments and definition. Each nuance evolves into a nuisance when peering across the landscape of mathematical software.

To correct this imbalance, the consortium is deriving mathematically-sound semantic definitions for these core functions. This initiative impacts the entire algorithm supply chain from the hardware vendor and software developer to the consumer. Why *semantic* definitions? Let's take a look at the difference.

Semantics Over Syntax

In layman's terms, *semantics* deals with behavior while *syntax* dictates structure or order. Given present-day programming languages, software developers are familiar with syntax. In fact, most attempts to standard math functionality in the past decade concentrated efforts in the arena of syntax. Unfortunately, this work has not produced a universally accepted specification for math algorithm development.

Why this labor failed to generate collective backing is not known. However, some consequences of establishing functionality based on syntax are obvious:

- The syntax of programming and scripting languages are different by design. They are used to differentiate products and cater to the unique needs of consumers. When the syntax of functions is defined, only a handful of the existing languages are compliant. The rest are forced to make a choice: keep compatibility with existing customer needs or break in favor of the new approach.
- The syntax of a function includes the number of arguments, their types and their order in the function call. As with the languages, vendors choose specific types and ordering

based on their target audience. Certain arguments are provided based on the underlying implementation, have no bearing on the mathematics of the function, and are useful in limited domains.

- Changes to function syntax require consumers to re-learn existing mathematics functionality. If this encompasses a hundred or more functions, then even the most math knowledgeable customer is affected.

Regardless of why, one thing is certain - the approaches failed to unify an ever-widening set of mathematical functions.

Shifting focus from syntax to semantics, the consortium is concentrating on function semantics. This approach puts the mathematics of a function at center stage. A semantic function definition requires:

1. The mathematical definition(s) performed by the function.
2. The definition of the input and output arguments required by the mathematical definition(s).
3. The data types required for function arguments in purely mathematical terms.

Optionally, these definitions contain additional information such as published references useful in researching particular mathematical concepts. Out of necessity, a function's arguments are listed in a particular order. This order is *not* a part of the specification – it is used solely to improve readability and comprehension.

By relying on semantic definitions, both vendor and consumer are free to choose whatever development environments or algorithm implementations suit their needs. If requirements change during a development cycle or multiple environments are demanded, then reuse of function-based algorithms is possible and savings are realized. Because these definitions leave implementation details up to the designer, the vendor is free to offer specialized versions in addition to a version conforming to the consortium's specification. This fosters creativity, allows differentiation, and yet facilitates compatibility with existing software packages.

Core Mathematics Functionality

The core mathematical functions are at the heart of the consortium's efforts. This list numbers more than 250, so the functions are partitioned into categories based on their mathematical genre.

Approximation, Integration and Interpolation

Boolean Functions

Relational Operators

Elementary Functions

Linear Algebra

Math Constants

Matrix Generation

Matrix Operators

Membership Functions

Polynomial Functions

Set Operators

Special Functions

Statistic Functions

Vector Analysis

On the following pages, a synopsis of each category is given and the functions in it are enumerated. For more detailed information, refer to the technical documentation available at www.nmconsortium.org.

NMC - Core Mathematical Function List

APPROXIMATION, INTEGRATION AND INTERPOLATION

Use members of the APPROXIMATION class to perform approximation, integration, and interpolation calculations.

Cubic Spline Interpolation
Numerical Integration (2D)
Numerical Integration (3D)
Numerical Integration (Adaptive Labatto)
Numerical Integration (Adaptive Simpson)
Numerical Integration (Trapezoidal Method)
Rational Approximation

BOOLEAN FUNCTIONS

Use members of the BOOLEAN class to manipulate Boolean content of variables.

Logical AND
Logical Exclusive OR
Logical NOT
Logical OR
To Logical

RELATIONAL OPERATORS

Use members of the COMPARISON class to compare numbers and matrices.

Equal?
Greater Than or Equal?
Greater Than?
Less Than or Equal?
Less Than?
Matrix Equal?
Not Equal?

ELEMENTARY FUNCTIONS

Use members of the ELEMENTARY class to perform basic exponential, trigonometric or other special numeric computations.

Basic Numerics

Absolute Value
Factorial
Greatest Common Divisor
Least Common Multiple
List Factors
Modulo
Quotient & Remainder
Round To Nearest Integer
Round Towards 0
Round Towards Negative Infinity
Round Towards Positive Infinity
Sign

Basic Geometry

Cartesian To Cylindrical
Cartesian To Polar
Cartesian To Spherical
Cylindrical To Cartesian
Givens Rotation
Polar To Cartesian
Spherical To Cartesian

Complex Support

Complex Argument
Complex Conjugate
Complex Modulus
Extract Imaginary Part
Extract Real Part
To Complex

Exponential

Exponential
Hyperbolic Cosecant
Hyperbolic Cosine
Hyperbolic Cotangent
Hyperbolic Secant
Hyperbolic Sine
Hyperbolic Tangent
Inverse Hyperbolic Cosecant
Inverse Hyperbolic Cosine
Inverse Hyperbolic Cotangent
Inverse Hyperbolic Secant
Inverse Hyperbolic Sine
Inverse Hyperbolic Tangent
Logarithm Base 10
Logarithm Base 2
Natural Logarithm
Next Power Of 2
Power Of 2
Real Natural Logarithm
Real Power
Real Square Root
Square Root

Trigonometric

Cosecant
Cosine
Cotangent
Inverse Cosecant
Inverse Cosine
Inverse Cotangent
Inverse Secant
Inverse Sine
Inverse Tangent
Secant
Sinc
Sine
Tangent

LINEAR ALGEBRA

Use members of the LINALGEBRA class to manipulate real and complex matrices.

Balance Matrix
Biconjugate Gradient Solver
Biconjugate Gradient Stabilized Solver
Cholesky Decomposition
Cholesky Rank 1 Update
Condition Number
Condition Number Estimate
Condition Number via SVD
Count Nonzero Elements
Determinant
Dot Product
Eigenvectors and Eigenvalues
Exchange Matrix Columns
Exchange Matrix Elements
Exchange Matrix Rows
Extract Lower Triangular
Extract Upper Triangular
Find Nonzero Elements
General Matrix Function
Get Diagonal
Get Nonzero Elements
Hessenberg Form
LU Decomposition
Matrix Conjugate Transpose
Matrix Exponential
Matrix Exponential (Eigenvectors)
Matrix Exponential (Pade Approximation)
Matrix Exponential (Taylor series)
Matrix Inverse
Matrix Natural Logarithm
Matrix Power
Matrix Pseudo-Inverse
Matrix Rank
Matrix Size
Matrix Square Root
Matrix Trace
Matrix Transpose
Norm
Norm Estimate
Null Space
Number Of Dimensions
Number Of Elements
QR Decomposition
QZ Decomposition
Range Space
Row Echelon Form
Schur Decomposition
Set Diagonal
Singular Value Decomposition
Subspace Angle

MATH CONSTANTS

Use members of the MATHCONSTANTS class to reference known mathematical data type states useful in manipulating and evaluating expressions.

FALSE
Infinity
PI
TRUE

NMC - Core Mathematical Function List

MATRIX GENERATION
Use members of the MATRIXGEN class to create matrices and vectors of special structure or content.

Special Elements
Companion Matrix
Hadamard Matrix
Hankel Matrix
Hilbert Matrix
Identity Matrix
Inverse Hilbert Matrix
Magic Square
Matrix Of TRUE
Matrix Of FALSE
Matrix Of Ones
Matrix Of Zeros
Pascal Matrix
Peaks Matrix Generator
Prime Numbers
Rosser Matrix
Toeplitz Matrix
Vandermonde Matrix
Wilkinson Matrix

Special Forms
All Permutations
Block Diagonal Matrix
Diagonal Matrix
Horizontal Concatenation
Matrix Concatenation
Replicate Matrix
Reshape Matrix
Vertical Concatenation

MATRIX OPERATORS
Use members of the MATRIXOPS class to generate new matrices using unitary and binary matrix functions.

Algebraic
Elementwise Addition
Elementwise Left Division
Elementwise Multiplication
Elementwise Power
Elementwise Right Division
Elementwise Subtraction
Matrix Left Division
Matrix Multiplication
Matrix Right Division
Negate
Unary Addition

Other
Accumulate Products
Accumulate Sums
Circular Shift
Differences
Kronecker Product
Product Of Elements
Rotate 90°
Summation Of Elements

MEMBERSHIP FUNCTIONS
Use members of the MEMBERSHIP class to determine if elements belong to a given class of objects.

Data Type
Is Empty?
Is Finite?
Is Infinite?
Is Logical?
Is Member?
Is Numeric?
Is Object Type?
Is Ordered?
Is Prime?
Is Real?

POLYNOMIAL FUNCTIONS
Use members of the POLYNOMIALS class to generate or manipulate polynomials and polynomial roots.

Convert To Pole-Residue Form
Convert To Rational Polynomial
Convolution
Deconvolution
Evaluate Polynomial
Evaluate Polynomial (Matrix Value)
Fit Polynomial
Piecewise Polynomial
Polynomial Eigenvalue Solver
Polynomial From Roots
Polynomial Integral
Polynomial Roots

SET OPERATORS
Use members of the SETOPS class to construct sets.

Remove Duplicates
Set Difference
Set Exclusive OR
Set Intersection
Set Union

SPECIAL FUNCTIONS
Use members of SPECIAL class to compute Airy, Bessel, beta, elliptic, and other special functions.

Airy (1^{st} & 2^{nd} Kind)
Associated Legendre
Bessel (1^{st} Kind)
Bessel (2^{nd} Kind)
Bessel (3^{rd} Kind)
Beta
Center Zero Frequency Component
Complete Elliptic Integral (1^{st} & 2^{nd} Kind)
Error
Error Complement
Exponential Integral
Fourier Transform
Gamma
Incomplete Beta
Incomplete Gamma
Inverse Error
Inverse Error Complement
Inverse Fourier Transform
Jacobi Elliptic
Linearly Sampled Interval
Logarithmically Sampled Interval
Modified Bessel (1^{st} Kind)
Modified Bessel (2^{nd} Kind)
Natural Logarithm Of Beta
Natural Logarithm Of Gamma
Digamma
Restore Zero Frequency Component
Scaled Error Complement
Unwrap Phase Angle

STATISTIC FUNCTIONS
Use members of the STATISTICS class to perform statistical analysis.

Binomial Coefficient
Correlation Coefficients
Covariance Matrix
Delete Linear Trends
Histogram
Histogram Count
Maximum Element
Mean Value
Median Value
Minimum Element
Pseudo-random Number (Distribution-dependent)
Pseudo-random Number (Normal Distribution)
Pseudo-random Number (Uniform Distribution)
Random Permutation
Standard Deviation
Variance

VECTOR ANALYSIS FUNCTIONS
Use members of the VECTOR ANALYSIS class to generate vectors from vectors and matrices.

Cross Product
Curl
Gradient