Subject: Intrinsic math functions
From: Van Snyder
Reference: 05-248r3

1 Introduction

As specified in 05-248r3, the HYPOT function a is nearly useless, and ERFC is suboptimal.

1.1 HYPOT

The HYPOT function specified in 05-248r3 is nearly useless, since (1) it's identical to CABS, and (2) the interesting function in any case is the one that computes the L_2 norm of a vector of any length, not just length 2. Instead of HYPOT we ought to add the NRM2 functions from the BLAS, with the generic name TBD. Once one has CABS (or indeed NRM2) one needs only a statement function to have a HYPOT spelling. Presumably, if the length is given by an initialization expression, and happens to be two, a decent processor will optimize NRM2 just as well as it would HYPOT — or maybe it won't if HYPOT isn't in the SPEC benchmark. Either HYPOT or NRM2 can be done with SQRT(DOT_PRODUCT(A,A)), but a carefully-done L_2 norm function (e.g., the one in the BLAS) will not experience an overflow in the calculation of an intermediate result unless the final result overflows.

Proposal: Add a function that computes the L_2 norm of an array.

1.2 ERFC

The ERFC function specified in 05-248r3 is not the most useful specification for that functionality. ERFC is asymptotic to $\exp(-x^2)/(x\sqrt{\pi})$, and as such underflows for $x >\approx 9$ in IEEE single precision. The expression $\exp(x^2)\text{erfc}(x)$, which doesn't underflow until $x > \text{HUGE}(x)/\sqrt{\pi}$, appears more frequently in statistical calculations. Real math function libraries (as opposed to libm) frequently include a "scaled erfc" function, frequently called ERFCE, that computes $\exp(x^2)\text{erfc}(x)$. Where carefully done, implementations of this function do not experience overflow of intermediate or final results for any positive X, and intermediate or final results underflow only for $x > \text{HUGE}(x)/\sqrt{\pi}$. It *is not* computed by computing $\text{erfc}(x)$ and multiplying by $\exp(x^2)$, as a naive user might be tempted to do — especially if all we provide is the functionality presently specified in 05-248r3. $\exp(x^2)$ would overflow for $x >\approx 9$ in IEEE single precision, and $\text{erfc}(x)$ would underflow around the same value, so multiplying the results of those functions would produce nonsense for $x >\approx 9$.

Proposal: Add a function that computes an exponentially scaled complementary error function.

2 Syntax

No new syntax and no changes to existing syntax.

3 Edits

Edits refer to 04-007. Page and line numbers are displayed in the margin. Absent other instructions, a page and line number or line number range implies all of the indicated text is to be replaced by associated text, while a page and line number followed by + (-) indicates that associated text is to be inserted after (before) the indicated line. Remarks are noted in the margin, or appear between [and] in the text.

3.1 Exponentially scaled complementary error function

[Insert into list of Mathematical functions in 13.5.2:] 294:30+

ERFC_SCALED (X)	Exponentially-scaled complementary error function

---

[Insert after 13.7.5 EPSILON (X):] 315:24+

13.7.5$\frac{1}{2}$ ERFC_SCALED (X)

Description. Exponentially-scaled complementary error function.

Class. Elemental function.

Argument. X shall be of type real.

Result Characteristics. Same as X.

Result Value. The value of the result is a processor-dependent approximation to the exponentially-scaled complementary error function, $\exp(x^2)\frac{2}{\sqrt{\pi}}\int_x^\infty \exp(-t^2)\mathrm{d}t$.

Example. The value of ERFC_SCALED(20.0) is 0.02817434874 (approximately)

NOTE 13.8$\frac{1}{2}$

The complementary error function is asymptotic to $\exp(-x^2)/(x\sqrt{\pi})$. As such it underflows for $x >\approx 9$ when using single-precision IEEE arithmetic. The exponentially-scaled complementary error function is asymptotic to $1/(x\sqrt{\pi})$. As such it does not underflow until $x >$ HUGE$(x)/\sqrt{\pi}$.

3.2 L_2 Norm

[Insert into list of Array reduction functions in 13.5.12:] 297:7+

NORM2 (X) L_2 norm of an array

[Insert after 13.7.87 NOT (I):] 340:26+

13.7.87$\frac{1}{2}$ NORM2 (X)

Description. L_2 norm of an array.

Class. Transformational function.

Argument. X shall be of type real. It shall not be a scalar.

Result Characteristics. Scalar of the same type and kind type parameter value as X.

Result Value. The result has a value equal to a processor-dependent approximation to the L_2 norm of X if X is a rank-one array, the Frobenius norm of X if X is a rank-two array, and the generalized L_2 norm of X for higher-rank arrays. In all cases, this is the square root of the sum of the squares of all elements.

Case (i): X is a rank-one array.

$$\mathrm{NORM2}(X) = \sqrt{\sum_{i=1}^{\mathrm{SIZE}(X)} X(i)^2}$$

Case (ii): X is a rank-two array.

$$\mathrm{NORM2}(X) = \sqrt{\sum_{i=1}^{\mathrm{SIZE}(X,1)} \sum_{j=1}^{\mathrm{SIZE}(X,2)} X(i,j)^2}$$

Case (n): X is a rank-n array.

$$\mathrm{NORM2}(X) = \sqrt{\sum_{i_1=1}^{\mathrm{SIZE}(X,1)} \cdots \sum_{i_n=1}^{\mathrm{SIZE}(X,n)} X(i_1,\ldots,i_n)^2}$$

Example. The value of NORM2((/ 3.0, 4.0 /)) is 5.0 (approximately).

NOTE 13.16$\frac{1}{2}$

It is recommended that the processor compute NORM2 in such a way that intermediate results do not overflow or underflow unless the final result would overflow or underflow, respectively.