Bergische Universität
GH Wuppertal

Algorithms with Guaranteed Error Bounds for the Error Function and the Complementary Error Function

Walter Krämer and Frithjof Blomquist

Preprint 2000/2

Wissenschaftliches Rechnen/
Softwaretechnologie

Impressum

Herausgeber:	Prof. Dr. W. Krämer, Dr. W. Hofschuster Wissenschaftliches Rechnen/Softwaretechnologie Fachbereich 7 (Mathematik) Bergische Universität GH Wuppertal Gaußstr. 20 D-42097 Wuppertal

Internet-Zugriff

Die Berichte sind in elektronischer Form erhältlich über die World Wide Web Seiten

`http://www.math.uni-wuppertal.de/wrswt/literatur.html`

Autoren-Kontaktadresse

Prof. Dr. W. Krämer
Bergische Universität GH Wuppertal
Gaußstr. 20
D-42097 Wuppertal

Dr. Frithjof Blomquist
Adlerweg 6
66346 Püttlingen

Contents

1 Introduction **4**

2 Notation **5**

3 Generalities on Error Splitting **6**

4 Principalities on Error Estimates for Special Functions **7**

5 About the Approximation Error **9**

5.1 Review 9

5.2 Rational Approximation 12

5.3 Estimating the Approximation Error 14

6 The error functions erf(x) and erfc(x) **16**

6.1 The Coarse Algorithm 17

6.2 Approximation of erf(x) in A=[0, 0.65] 19

6.3 Error Bound for erfc(x) in A= [0, 0.65] 23

6.4 Approximation of erfc(x) in $B_1 \bigcup B_2 = [0.65, 6]$ 24

6.5 Error Bound for erf(x) in $B_1 \bigcup B_2 = [0.65, 6]$ 32

6.6 Approximation of erfc(x) in $B_3 = [6, 26.5432]$ 35

6.7 Error Bound for erf(x) in $B_3 \bigcup B_4$, i. e. for $x \geq 6$ 38

6.8 Approximation of erfc(x) in $C = (-\infty, 0]$ 39

6.9 Summary of the results for erf(x) and erfc(x) 39

A The Auxiliary Function e^{-x^2} **40**

B A Simple Test Program, Numerical Results **42**

Abstract

The given algorithms allow the calculation of confinements for values of the error function erf(x) resp. the complementary error function erfc(x) for point and interval arguments in the IEEE–double number format [29]. As well the approximation errors in the various parts as all appearing rounding errors are seized certainly with a priori error estimates by using interval methods. The obtained worst–case error bounds for the maximum relative error are valid for all admissible arguments simultaneously. They are finally used for certain confinement of the ranges over points or intervals. Under the address `http://www.math.uni-wuppertal.de/wrswt/software/erf` a complete XSC–implementation [25] of the algorithms discussed can be found. All approximation coefficients are given, such that a translation into another programming language is very simple.

Key Words: Error Function, Complementary Error Function, Reliable Error Estimates, IEEE–double Format, Special Functions

MSC: 65D15, 65G05, 65G10, 68M15

1 Introduction

For algorithms in the domain of numerical mathematics with result verification so–called interval functions are needed whose result interval with certainty confines the exact range of the considered function with interval arguments. The calculated confinement should be as narrow as possible.

For the implementation of such functions, sure a priori estimates of the approximation errors in the various parts as well as a priori worst–case error estimates of the inaccuracies due to rounding errors are necessary. In the first sections of this paper, this theme is discussed first of all generally. XSC–programs allowing large parts of the error estimates to be performed automatically by the machine, are described in [3, 4, 10, 11, 14, 16] more closely.

In the actual main section the algorithms for the error function erf(x) and its complement erfc(x) are given. For the various parts are developed auxiliary approximation functions and their approximation error is determined analytically. By means of these (programmable and almost perfect) auxiliary approximation functions, the errors of the rational approximations actually used may later be determined by the machine using numerically sure interval methods.

The relative worst–case error bounds found here are finally used for putting together the interval routines from the point functions. By calculating confinements, the separate treatment of arguments leading to values in the underflow area may be omitted (for such arguments the obtained relative error bound generally does not hold). These values are mapped either to 0 or the smallest positive or the largest negative normalized floating point number, in such a way that the desired confinement property remains guaranteed.

The complete XSC program listings of the error function as well as the complementary error function for point and interval arguments and also the source code

of some important auxiliary routines are available on the web under the address `http://www.math.uni-wuppertal.de/wrswt/software/erf`. In the appendix of this paper is listed a short test program with numerical results.

2 Notation

$S = S(b,l) = S(b,l,\underline{e},\overline{e})$	floating point screen with base b, mantissa length l and exponent e with $\underline{e} \le e \le \overline{e}$
$S(2,53,-1022,1023)$	IEEE data format double
$\circ \in \{+,-,\bullet,/\}$	exact real operation
$\boxdot\,, \circ \in \{+,-,\bullet,/\}$	floating point operator, machine operation with rounding to the closest floating point number
$\bigtriangledown\,, \circ \in \{+,-,\bullet,/\}$	machine operation with rounding downwards
$\bigtriangleup\,, \circ \in \{+,-,\bullet,/\}$	machine operation with rounding upwards
$\lvert a \circ b - a \boxdot b\rvert := \lvert(a \circ b) - (a \boxdot b)\rvert$	notice implicit bracketing!
`MinReal`	smallest positive normalized floating point number, for IEEE data format: `MinReal`:=$2.22\ldots \cdot 10^{-308}$
`MaxReal`	largest normalized floating point number, for IEEE data format: `MaxReal`:=$1.78\ldots \cdot 10^{308}$
$a,\ x,\ f,\ \ldots$	exact quantities
$\tilde{a},\ \tilde{x},\ \tilde{f},\ \ldots$	machine–calculated, generally erroneous, quantities
ulp	(a) unit (in the) last (mantissa) place
$\varepsilon^* := \frac{1}{2}2^{1-l} = 2^{-l}$	relative machine accuracy with respect to $S(b,l)$
`eps52`	`eps52`:= $2^{1-53} = 2.22044\ldots \cdot 10^{-16}$
`eps53`	`eps53`:= $\frac{1}{2}2^{1-53} = 1.11022\ldots \cdot 10^{-16} = \varepsilon^*$
$\mathrm{succ}(x),\ x \in S$	floating point successor of x
$\mathbb{R}^+$	set of positive reals
$I\mathbb{R}$	set of closed intervals in the reals
$X = [\underline{x},\ \overline{x}] \in I\mathbb{R}$	notation for intervals
$IS := \{[\underline{a},\overline{a}] \lvert \underline{a},\overline{a} \in S, \underline{a} \le \overline{a}\}$	set of machine intervals
$\lvert A\rvert, A \in I\mathbb{R}$	$\lvert A\rvert := \max\limits_{a\in A} \lvert a\rvert$, maximum of absolute values
$\langle\, A\,\rangle, A \in I\mathbb{R}$	$\langle\, A\,\rangle := \min\limits_{a\in A} \lvert a\rvert$, minimum of absolute values
$\mathrm{diam}(A) := \sup(A) - \inf(A)$	diameter of an interval $A \in I\mathbb{R}$
$W_f(X) := \{f(x) \,\vert\, x \in X\}$	range of f on interval X
$H(x) \approx f(x)$	almost perfect auxiliary approximation function for the function $f(x)$. $H(x)$ generally also depends on the momentarily considered approximation domain. $H(x)$ is used to find automatically error bounds for the efficient fast approximation finally sought
$\varepsilon(\mathrm{app},1)$	relative error bound of the almost

	perfect approximation $H(x)$ for f
$\varepsilon(\text{app}, 2)$	relative error bound of the implemented approximation (relative to $H(x)$)
$\varepsilon(f)$	relative total error bound of the final machine realization $\widetilde{f}$ of f

3 Generalities on Error Splitting

In this section those errors are considered which usually occur when evaluating a given continuous function

$$h \,:\, [c,d\,] \longrightarrow I\!R, \quad c,d \in S(B,k)$$

on a machine with floating point system S with k mantissa digits to the base B. The possible errors appear while:

1. calculating a reduced argument x resp. its machine approximation $\widetilde{x}$ using the continuous function

$$r \,:\, [c,d\,] \longrightarrow I\!R, \qquad W_r([c,d\,]) = [a,b] := \{\, x \,|\, x = r(t),\, t \in [c,d\,]\,\}$$

$$\widetilde{x} = \tilde{r}(t) = x \cdot (1+\varepsilon_x), \qquad |\varepsilon_x| \le \varepsilon(x) \quad \text{for all } x \in [a,b], \quad a,b \in I\!R.$$

2. approximating $h(t) = f(r(t)) = f(x)$ by $g(x)$:

$$f(x) \approx g(x), \quad x = r(t) \in [a,b];$$

$$\varepsilon_{\text{app}} := \frac{g(x)-f(x)}{f(x)}, \qquad |\varepsilon_{\text{app}}| \le \varepsilon(\text{app}) \quad \text{for all } x \in [a,b] \text{ with } f(x) \neq 0.$$

3. evaluating the approximation function g for the generally perturbed values of $\widetilde{x}$ under argument reduction:

$$\varepsilon_g := \frac{\widetilde{g}(\widetilde{x}) - g(x)}{g(x)}, \qquad |\varepsilon_g| \le \varepsilon(g), \quad \widetilde{x} = x \cdot (1+\varepsilon_x), \quad x \in [a,b].$$

Thus, if for $t \in [c,d\,]$ a reduced argument x is calculated using the continuous function $x = r(t)$, and if $[a,b]$ with $a,b \in I\!R$ is the range of r, then instead of $h(t)$ with $t \in [c,d\,]$ the function $f(x) = f(r(t)) = h(t)$ with $x \in [a,b]$ is to be evaluated. Since $x = r(t)$ is calculated on the machine for given t, one does not obtain $x \in [a,b]$, but the generally erroneous value $\widetilde{x} = x \cdot (1+\varepsilon_x) \in S$ with the relative error bound $\varepsilon(x)$. If $g(x) \approx f(x)$ for $x \in [a,b]$ is the approximation function, then the latter is not evaluated with the exact x, but with $\widetilde{x} = x \cdot (1+\varepsilon_k) \in S(B,k)$. Since for this evaluation on the machine more rounding errors are to be expected, one does not obtain $g(\widetilde{x}) \in I\!R$, but the generally erroneous machine value $\widetilde{g}(\widetilde{x}) \in S(B,k)$. Instead of $f(x) = h(t)$ one therefore obtains only $\widetilde{g}(\widetilde{x})$ with the relative error

$$\varepsilon_h(t) := \frac{\widetilde{g}(\widetilde{x}) - h(t)}{h(t)} = \frac{\widetilde{g}(\widetilde{x}) - f(x)}{f(x)},$$

where $f(x) \neq 0$ for $x \in [a,b]$ is presumed. Using the error splitting $|\varepsilon_h(t)|$ may now be estimated [12]:

$$\begin{aligned}\left|\frac{f(x)-\widetilde{g}(\widetilde{x})}{f(x)}\right| &= \left|\frac{f(x)-g(x)+g(x)-\widetilde{g}(\widetilde{x})}{f(x)}\right| \\ &\leq \left|\frac{f(x)-g(x)}{f(x)}\right| + \left|\frac{g(x)-\widetilde{g}(\widetilde{x})}{g(x)}\right| \cdot \left|\frac{g(x)-f(x)+f(x)}{f(x)}\right| \\ &\leq \varepsilon(\text{app}) + [1+\varepsilon(\text{app})]\cdot\varepsilon(g) =: \varepsilon(h)\end{aligned} \tag{1}$$

$\varepsilon(\text{app})$ is the relative approximation error, and $\varepsilon(g)$ means the relative evaluation error of the approximation function g. $\varepsilon(h)$ is the total error bound of $h(t)$ for $t \in [c,d]$, if the reduced argument $x = r(t)$ is calculated only approximately and therefore in general erroneously to $\widetilde{x} = x\cdot(1+\varepsilon_x)$, $\quad |\varepsilon_x| \leq \varepsilon(x)$, and if the approximation function g is evaluated afterwards for this perturbed argument $\widetilde{x}$ using floating point operations on the machine: $\widetilde{g}(\widetilde{x}) = g(x)\cdot(1+\varepsilon_g)$; $\quad |\varepsilon_g| \leq \varepsilon(g)$.

If one calculates $\varepsilon(h)$ according to formula (1), then it is assured that

$$\left|\frac{h(t)-\widetilde{g}(\widetilde{r}(t))}{h(t)}\right| \leq \varepsilon(h), \qquad \text{simultaneously for all } t \in [c,d].$$

This means that $\varepsilon(h)$ is just the dependable total error bound wanted for the machine realization $\widetilde{h}(t) := \widetilde{g}(\widetilde{r}(t))$ of $h(t)$ and therefore of $f(x)$.

4 Principalities on Error Estimates for Special Functions

In connection with realizing special functions (error function, gamma–function, Bessel–functions, ...) in computers it appears that it is often significant for the safe error estimation to use, in addition to the actually wanted and on the calculator to be realized approximation $g(x)$, an auxiliary approximation function $H(x)$ (this is in general an almost perfect approximation e. g. one which is calculable with a long number arithmetic. More exactly can be proceeded as described in the following way.

One may proceed again from a continuous real–valued function

$$f \,:\, [a,b] \longrightarrow I\!R$$

to be evaluated on a calculator with a floating point screen $S(B,k)$. In order to obtain short run–time, the approximating function $g \approx f$ should be as constructed as simply as possible (e. g. rational function). If one denotes again with $\widetilde{f}(x)$ the in general erroneous machine result, then

$$\widetilde{f}(x) = f(x)(1+\varepsilon_f), \qquad |\varepsilon_f| \leq \varepsilon(f) \quad \text{for all } x \in [a,b] \cap S(B,k).$$

In order to calculate the wanted error bound $\varepsilon(f)$, generally **two** approximation steps are necessary. First an auxiliary function $H(x)$ approximating $f(x)$ has to be found which may be built up actually quite complicated. It is only demanded that $H(x)$ be programmable with interval functions resp. interval operations already implemented. The corresponding approximation error bound $\varepsilon(\mathrm{app},1)$ generally has to be developed analytically (by hand). The extraction of the actual approximation function $g(x)$ for implementing the initial function $f(x)$ is then made using $H(x)$. A bound $\varepsilon(\mathrm{app},2)$ for the hereby appearing (second) approximation error may now be automatically determined by interval arithmetical means. More exactly, it is proceeded as following:

Step 1: $\qquad f(x) \approx H(x), \qquad x \in [a,b],$

where $H(x)$ is built only from the standard functions supplied by the interval long number module `mpitaylor`. The relative approximation error is given for $f(x) \neq 0$ by:

$$\varepsilon_{\mathrm{app},1} = \frac{f(x) - H(x)}{f(x)}; \qquad |\varepsilon_{\mathrm{app},1}| \leq \varepsilon(\mathrm{app},1), \quad x \in [a,b].$$

The calculation of $\varepsilon(\mathrm{app},1)$ is therefore a purely mathematical problem to be solved for every function individually. For this it is not demanded that $H(x)$ can be evaluated on the calculator as quickly as possible. The main objective is rather to calculate a **guaranteed** upper limit $\varepsilon(\mathrm{app},1)$ for the corresponding approximation error!

Step 2: $\qquad H(x) \approx g(x), \qquad x \in [a,b],$

where g now denotes the function which actually approximates f on the calculator. In order to obtain short run–time, g is chosen in many cases as a broken rational function whose coefficients may be calculated using a computer algebra system. For the relative approximation error (with respect to the auxiliary function $H(x)$) it is possible, assuming $H(x) \neq 0$, to determine automatically on the calculator the upper bound $\varepsilon(\mathrm{app},2)$:

$$\varepsilon_{\mathrm{app},2} = \frac{H(x) - g(x)}{H(x)}, \qquad |\varepsilon_{\mathrm{app},2}| \leq \varepsilon(\mathrm{app},2), \quad x \in [a,b].$$

For this may be used the XSC–program `AppErr` [4, 14].

Corresponding to the approximation $f \approx g$ the relative approximation error is defined as

$$\varepsilon_{\mathrm{app}} := \frac{f(x) - g(x)}{f(x)}, \qquad |\varepsilon_{\mathrm{app}}| \leq \varepsilon(\mathrm{app}), \quad x \in [a,b],$$

and $|\varepsilon_{\mathrm{app}}|$ may be approximated using the bounds $\varepsilon(\mathrm{app},1)$ and $\varepsilon(\mathrm{app},2)$ and the triangle inequality:

$$\boxed{|\varepsilon_{\mathrm{app}}| \leq \varepsilon(\mathrm{app},1) + [1 + \varepsilon(\mathrm{app},1)] \cdot \varepsilon(\mathrm{app},2) =: \varepsilon(\mathrm{app})\,.} \tag{2}$$

If $\widetilde{g}(x)$ denotes the generally erroneous machine result of g, then the representation

$$\widetilde{g}(x) = g \cdot (1 + \varepsilon_g)$$

follows, and the upper bound $\varepsilon(g)$ for the maximum absolute value of the relative evaluation error ε_g may be calculated automatically with an XSC–program [4, 10]. Thereafter

$$\widetilde{g}(x) = g(x) \cdot (1 + \varepsilon_g); \qquad |\varepsilon_g| \leq \varepsilon(g) \quad \text{for all } x \in [a,b] \cap S(B,k).$$

With the total error estimate (1) one finally obtains with $\widetilde{f}(x) = f(x)(1+\varepsilon_f)$

$$\boxed{|\varepsilon_f| \leq \varepsilon(\text{app}) + [1 + \varepsilon(\text{app})] \cdot \varepsilon(g) =: \varepsilon(f)\,.} \tag{3}$$

Hereby is assumed

$$x \in [a,b] \cap S(B,k) \;\wedge\; f(x) = 0 \quad \Longrightarrow \quad \widetilde{f}(x) \equiv \widetilde{g}(x) = 0\,.$$

This requirement may be fulfilled in general simply by a suitable special treatment of the zeros of the considered function in the developing of the algorithm for g.

5 About the Approximation Error

5.1 Review

When implementing a real–valued function defined on a real interval

$$f : [a,b] \;\longrightarrow\; I\!R$$

on a calculator, the following two points are to be observed:

- In order to obtain short run–time, f should be approximated on the interval $[a,b]$ by a rational function, where numerator and denominator each are defined by a polynomial:

$$f(x) \approx g(x) := \frac{\mathrm{P}_N(x)}{\mathrm{Q}_M(x)}; \quad \mathrm{Q}_M(x) \neq 0, \quad x \in [a,b], \quad N, M \in \{0, 1, 2, \ \ldots\}$$

- Because of the approximation $f(x) \approx g(x)$ and because of the generally inevitable rounding error in the evaluation of g the machine result will be equal to the exact value $f(x)$ only in exceptional cases. In order to calculate a **guaranteed** error bound it is therefore necessary, among other things, also to determine a guaranteed upper bound for the approximation error for all $x \in [a,b]$.

The absolute resp. relative approximation error is defined by

$$\begin{aligned} \Delta(x) &:= f(x) - g(x), \quad |\Delta(x)| \leq \Delta(\text{app}), \quad x \in [a,b], \\ \varepsilon(x) &:= \frac{f(x) - g(x)}{f(x)}, \quad |\varepsilon(x)| \leq \varepsilon(\text{app}), \quad f(x) \neq 0, \quad x \in [a,b]. \end{aligned}$$

In order to determine the upper bounds $\Delta(\text{app}), \varepsilon(\text{app})$ there are various methods depending on the given function f and its approximation interval. For this $\Delta(\text{app}), \varepsilon(\text{app})$ should not be too large, and the upper limits must be calculated **correctly** from the mathematical point of view.

For standard functions (f = exp, sin, arctan, ...) $[a, b]$ may be reduced to a relatively small Interval whose centre is the origin of the coordinate system. As an approximating function may therefore be chosen a Taylor–polynomial of low order (short run–time), and the remainder of the Taylor–series may be estimaed either by a geometrical series or by the following member of the series if the power series is a Leibniz–series [6, 12]. Doing this, one obtains for the upper limit of the approximation error a simple expression in closed form which may be safely estimated from above using interval calculus.

When calculating the approximation error for special functions in mathematical physics, things are different as now an argument–reduction to a very small interval in general is not possible as opposed to standard functions.

The following example should illustrate this. With the Riemann zeta-function $\zeta(x)$ und the Euler–constant $\gamma = 0.57721\ldots$ for the function $f(x) = -\ln(\Gamma(x))$ the power–series expansion

$$f(x) \equiv (x-2)(\gamma-1) - \sum_{k=2}^{\infty}(-1)^k[\zeta(k)-1]\cdot\frac{(x-2)^k}{k}, \quad |x-2| \le \frac{1}{2}$$

holds. If one now approximates $f(x)$ in the interval [1.5, 2.5] by the Taylor–polynomial

$$\mathrm{T}_N(x) := (x-2)(\gamma-1) - \sum_{k=2}^{N}(-1)^k[\zeta(k)-1]\cdot\frac{(x-2)^k}{k} \approx f(x),$$

one obtains for the absolute error only with N=26 the upper limit $\Delta(\text{app}) = 4.1121\cdot 10^{-18}$ by the method described above. For run–time reasons the machine evaluation of $\mathrm{T}_{26}(x)$ is thus absolutely inacceptable. The calculation time, however, is reduced by about a factor of 2.4 if one replaces $\mathrm{T}_{26}(x)$ by a rational best approximation

$$f(x) \approx \mathrm{T}_{26}(x) \approx \frac{\mathrm{P}_6(x-2)}{\mathrm{Q}_5(x-2)}, \quad |x-2| \le \frac{1}{2} \tag{4}$$

where

$$\mathrm{P}_6(x-2) := \sum_{k=0}^{6} a_k\cdot(x-2)^k, \qquad \mathrm{Q}_5(x-2) := \sum_{k=0}^{5} b_k\cdot(x-2)^k\,.$$

The polynomial coefficients a_k, b_k may be determined e. g. by a computer algebra system (long number calculation). For the rational approximation here, the error must be estimated safely with respect to T_{26}.

In order to formulate this problem in somewhat more general terms, one often replaces the special polynomial $\mathrm{T}_{26}(x)$ by an auxiliary function $H(x)$ sufficiently many times differentiable which is built up in $|x-x_0| \le \eta$ from only finitely many standard functions provided by the XSC–module `mpitaylor`. One thus wants to find for the following approximation

$$H(x) \approx \frac{\mathrm{P}_N(x-x_0)}{\mathrm{Q}_M(x-x_0)}, \quad \mathrm{Q}_M(x-x_0) \neq 0, \quad |x-x_0| \le \eta \tag{5}$$

the upper bounds $\Delta(\text{app})$ resp. $\varepsilon(\text{app})$ for the absolute resp. relative approximation error. For $\varepsilon(x)$ e. g. this leads to

$$\varepsilon(x) := \frac{\mathrm{P}_N(x-x_0) - \mathrm{Q}_M(x-x_0)\cdot H(x)}{\mathrm{Q}_M(x-x_0)\cdot H(x)}, \quad |x-x_0| \le \eta, \ \text{ denominator} \neq 0, \qquad (6)$$

and the calculation of a guaranteed upper bound $\varepsilon(\text{app})$ for $|\varepsilon(x)|$ seems to be simple at first glance, being used to solve similar problems with methods of (verified) global optimization without difficulties. It turns out to be principally impossible to determine $\varepsilon(\text{app})$ by global optimization! In order to see this, it may be reminded that global optimization is possible only if the range of $\varepsilon(x)$ can be calculated **without** substantial overestimation for every subinterval of a finite partition of $|x-x_0| \le \eta$ [9, S.106]. If $[x]$ denotes such a subinterval, then in the numerator of (6) the expression

$$\mathrm{P}_N([x]-x_0) - \mathrm{Q}_M([x]-x_0)\cdot H([x]) \qquad (7)$$

is to be evaluated intervalwise, where the resulting intervals of minuend and subtrahend agree more the higher the polynomial degrees N, M are chosen in the rational approximation. In the expression (7) are therefore two almost identical intervals — and not points — to be subtracted which leads to very big overestimations. This is the reason why global optimization algorithms fail when error curves in approximation problems are studied. By the way, the problem is also not solved by evaluating the two interval addends in (7) using the interval long–term module `mpi_ari` [13] with high accuracy; for the fact that two almost identical intervals are to be subtracted is not eliminated by long number arithmetics. A purely theoretical solution could consist in choosing the subintervals $[x]$ quasi point–shaped, which would however lead to unpractical calculation times.

An estimation of the approximation error will therefore only be possible if one succeeds in transforming the expression (7) in such a way that prevents subtraction of two identical intervals. For this, one develops $H(x)$ in x_0 e. g. with the methods of automatic differentiation into a Taylor–polymomial with remainder $H(x) = \sum_{k=0}^{K} s_k \cdot (x-x_0)^k + \mathrm{R}(x,K)$. One obtains for the expression 7 the form

$$\underbrace{\left[\mathrm{P}_N(x-x_0) - \mathrm{Q}_M(x-x_0)\cdot\sum_{k=0}^{K} s_k\cdot(x-x_0)^k\right]}_{(*)} - \mathrm{Q}_M(x-x_0)\cdot\mathrm{R}(x,K)\,. \qquad (8)$$

The actual trick consists in calculating the polynomial difference $(*)$ **directly** by determining in (8) the coefficient of the subtrahend first and then takes the difference of the corresponding polynomial coefficients. Namely, if one confines the coefficient of minuend and subtrahend with long number interval arithmetics, then one obtains quasi point–like intervals separated far enough to calculate their differences without overestimation! Thus, the evaluation of $(*)$ is reduced to that of a single polynomial, which may be performed e. g. using the interval–Horner–scheme if $|x-x_0| \le \eta$ is partitioned into several subintervals for prevention of overestimation, whenever necessary.

The estimation of the remainder in Lagrangian form is done by automatic differentiation and evaluation of the $(k+1)$-th derivative of $H(x)$ on the interval $|x - x_0| \leq \eta$, where an interval–partitioning may also become necessary.

Let it be emphasized that the procedure described above shows very clearly that a naive application of interval calculus does not lead to the desired result, whereas under purposeful application on the right spot (difference of polynomial coefficients) interval arithmetics is a highly useful tool!

5.2 Rational Approximation

In this section we consider the approximation of a given auxiliary function $H(x)$ by a rational function, and show how a guaranteed upper bound for the absolute resp. relative approximation error can be calculated. Using an XSC–program these bounds may be calculated automatically.

As the procedure uses automatic differentiation, it is assumed that the auxiliary function $H(x)$ is given as a finite expression in

$$\text{exp, ln, sqr, sqrt, sin, cos, arctan, pow}$$

as well as the basic operations $-(\text{unary}), +, -, *, /$. (Such expressions can at present be treated with the module `mpitaylor`.)

The rational function approximating $H(x)$ may have

$$A_0 + A_1 \cdot (x - x_0)^1 + \ \ldots \ + A_N \cdot (x - x_0)^N$$

as numerator polynomial and

$$B_0 + B_1 \cdot (x - x_0)^1 + \ \ldots \ + B_M \cdot (x - x_0)^M$$

as denominator polynomial.

For fixed polynomial degrees N, M one often obtains a very effective approximation, if the coefficients A_j, B_j are determined according to Chebyshev. The absolute or relative approximation error then has in the interior of the approximation interval $|x - x_0| \leq \eta$ at least $N + M$ relative extrema with oscillating or equally absolute–valued extremal values. Equality of absolute value of the extremal values can in practice not be realized for the following reasons:

- The A_j, B_j can be calculated only up to finitely many decimals.
- It is in general sensible to round A_j, B_j to the nearest number in the screen in which the polynomials are to be evaluated.

Due to the necessary rounding of the polynomial coefficients the absolute extremal values of the approximation error will be different, and in general it cannot be guaranteed that the number of extrema does not change.

If a_j, b_j denote the coefficients arising from the A_j, B_j by an appropriate rounding, then the polynomials actually used for approximating on the machine are

$$\begin{aligned} \mathrm{P}_N(x - x_0) &:= a_0 + a_1 \cdot (x - x_0)^1 + \ \ldots \ + a_N \cdot (x - x_0)^N; \quad \text{numerator polynomial} \\ \mathrm{Q}_M(x - x_0) &:= b_0 + b_1 \cdot (x - x_0)^1 + \ \ldots \ + b_M \cdot (x - x_0)^M; \quad \text{denominator polynomial} \end{aligned}$$

i. e. $H(x)$ is approximated by

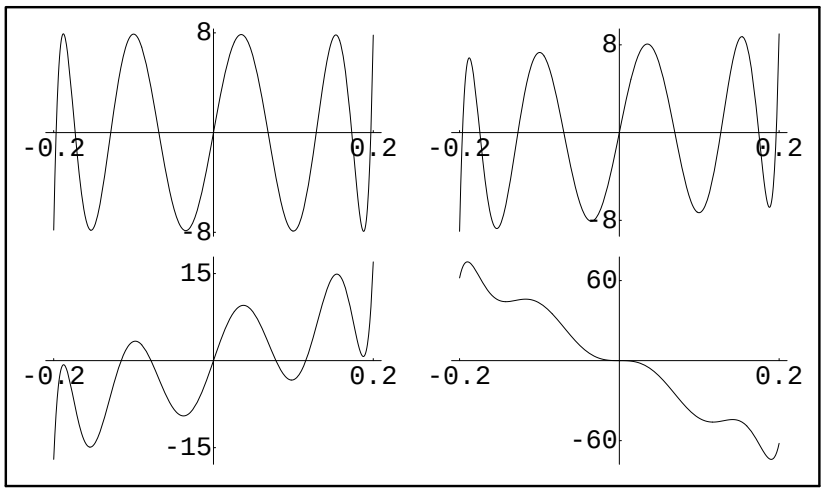
$$H(x) \approx \frac{\mathrm{P}_N(x-x_0)}{\mathrm{Q}_M(x-x_0)}, \quad \mathrm{Q}_M(x-x_0) \neq 0, \quad |x-x_0| \leq \eta.$$

The following example shows with four graphs the influence of rounding the polynomial coefficients on the progress of the relative approximation error. It is considered $H(x) := e^x, \quad x_0 = 0, \quad \eta = 0.2, \quad M = N = 4$. The first 18 decimals of the coefficients A_j, B_j calculated with Mathematica are

j	A_j	B_j
0	$9.99999999999999\ldots \cdot 10^{-1}$	$+1.00000000000000000\ldots \cdot 10^{+0}$
1	$4.99999999999998228\ldots \cdot 10^{-1}$	$-4.99999999999998228\ldots \cdot 10^{-1}$
2	$1.07140305127468128\ldots \cdot 10^{-1}$	$+1.07140305127468128\ldots \cdot 10^{-1}$
3	$1.19034858976579290\ldots \cdot 10^{-2}$	$-1.19034858976579290\ldots \cdot 10^{-2}$
4	$5.95025533669726278\ldots \cdot 10^{-4}$	$+5.95025533669726278\ldots \cdot 10^{-4}$

Table 1: the first 18 decimals of $A[j], B[j]$

Rel. Appr.-Fehler: a[j],b[j] mit 17,16,15,14 dezimalen Stellen



The four graphs in the figure above show the relative approximation error multiplied by 10^{+17} if the coefficients A_j, B_j calculated by Mathematica are rounded to 17, 16, 15 resp. 14 decimals of a_j, b_j. Thereby one sees e. g. that the maximum absolute value of the relative approximation error triples if the coefficients are rounded to only 14 decimals.

Due to the in practice necessary rounding of the polynomial coefficients A_j, B_j the relative extremal values do not have the same absolute value, so that an estimate of the absolute or relative approximation error relative to a_j, b_j is always required.

5.3 Estimating the Approximation Error

After introducing the principal method in the first section and after demonstrating the influence of rounding the polynomial coefficients on the approximation error, now are gathered formulae enabling the calculation (with a suitable XSC–program) of guaranteed upper limits for the absolute resp. relative error of the approximation

$$H(x) \approx \frac{\mathrm{P}_N(x-x_0)}{\mathrm{Q}_M(x-x_0)}, \quad \mathrm{Q}_M(x-x_0) \neq 0, \quad |x-x_0| \leq \eta \tag{9}$$

With the approximating polynomials (the coefficients are machine numbers)

$$\mathrm{P}_N(x-x_0) := \sum_{j=0}^{N} a_j \cdot (x-x_0)^j, \quad \mathrm{Q}_M(x-x_0) := \sum_{j=0}^{M} b_j \cdot (x-x_0)^j \tag{10}$$

and the Taylor expansion of the auxiliary function $H(x)$

$$H(x) = \mathrm{T}_K(x-x_0) + \mathrm{R}_K(x) \tag{11}$$

$$\mathrm{T}_K(x-x_0) := \sum_{j=0}^{K} s_j \cdot (x-x_0)^j$$

$$\mathrm{R}_K(x) := \frac{H^{(K+1)}(\zeta)}{(K+1)!} \cdot (x-x_0)^{K+1}, \quad \zeta = \zeta(x) \text{ between } x \text{ and } x_0 \tag{12}$$

the estimates for the absolute and relative approximation error are valid:

$$|\Delta(x)| \leq \left| \frac{\mathrm{Q}_M(x-x_0) \cdot \mathrm{T}_K(x-x_0) - \mathrm{P}_N(x-x_0)}{\mathrm{Q}_M(x-x_0)} \right| + |\mathrm{R}_K(x)|\,,$$

$$|\varepsilon(x)| \leq \left| \frac{\mathrm{Q}_M(x-x_0) \cdot \mathrm{T}_K(x-x_0) - \mathrm{P}_N(x-x_0)}{H(x) \cdot \mathrm{Q}_M(x-x_0)} \right| + \left| \frac{\mathrm{R}_K(x)}{H(x)} \right|, \quad H(x) \neq 0\,.$$

Since in practice the polynomial degrees N, M usually differ at most by 1, it is not a real restriction if in the following it is assumed that $M + K \geq N$. The basic idea is now to calculate the numerator polynomial above by evaluating the differences z_j of the polynomial coefficients using interval long number arithmetics

$$\mathrm{Q}_M \cdot \mathrm{T}_K - \mathrm{P}_N \equiv \mathrm{Z}_{M+K}(x-x_0) := \sum_{j=0}^{M+K} z_j \cdot (x-x_0)^j\,.$$

By inserting one then obtains the following estimations:

$$|\Delta(x)| \leq \left|\frac{\mathrm{Z}_{M+K}(x-x_0)}{\mathrm{Q}_M(x-x_0)}\right| + |\mathrm{R}_K(x)|, \quad |x-x_0| \leq \eta, \tag{13}$$

$$|\varepsilon(x)| \leq \left|\frac{\mathrm{Z}_{M+K}(x-x_0)}{H(x)\cdot\mathrm{Q}_M(x-x_0)}\right| + \left|\frac{\mathrm{R}_K(x)}{H(x)}\right|, \quad H(x) \neq 0 \tag{14}$$

In order to calculate the upper limits of $|\Delta(x)|, |\varepsilon(x)|$ with respect to $|x-x_0| \leq \eta$ the right hand side of (13),(14) are to be evaluated **with intervals**. Because of the appearing overestimations, the set $|x-x_0| \leq \eta$ must be divided into a sufficiently large number of intervals.

Estimating the Remainder in (13) resp. in (14)

According to (12), if

$$u := \frac{H^{(K+1)}([x_0-\eta, x_0+\eta])}{(K+1)!},$$

then

$$\mathrm{R}_K(x) \in u\cdot[-\eta^{K+1}, +\eta^{K+1}], \quad |\mathrm{R}_K(x)| \leq |u|\cdot\eta^{K+1} =: \tau, \quad \mathrm{R}_K(x) \in [-\tau, +\tau].$$

The interval u is calculated by automatic differentiation using the module `mpitaylor` [4], whereby $[x_0-\eta, x_0+\eta]$ can be divided into sufficiently many subintervals in order to prevent overestimation; in this case $|u|$ is the maximum of of the corresponding subinterval upper bounds $|u_j|$.

In (14) $\mathrm{R}_K(x)$ still has to be divided by $H(x)$. Since with automatic differentiation the confinement of the function value $H(u_j)$ is given for each subinterval, one divides $|u|\cdot\eta^{K+1}$ by the minimum of the values of the calculated $\langle\ |H([x]_j)|\ \rangle$. If this minimum vanishes, the quotient is set to `MaxReal` in order to show that an upper bound of the relative approximation error cannot be calculated.

Estimation of the First Term in (13) resp. in (14)

In (13), with the abbreviation

$$v := \frac{\mathrm{Z}_{M+K}([x_0-\eta, x_0+\eta]-x_0)}{\mathrm{Q}_M([x_0-\eta, x_0+\eta]-x_0)},$$

is valid the estimation

$$\left|\frac{\mathrm{Z}_{M+K}(x-x_0)}{\mathrm{Q}_M(x-x_0)}\right| \leq |v|.$$

If $|v|$ in a too large interval $[x_0-\eta, x_0+\eta]$ becomes too large due to overestimations of the Horner scheme in the polynomial evaluations, then $[x_0-\eta, x_0+\eta]$ must again be divided into sufficiently large number of subintervals.

In (14) the first term in the sum on the right hand side must additionally be divided by $H(x)$ which increases the runtime considerably if more complicated functions are used. This can, however, be prevented by replacing $H(x)$ according to (11) by its already calculated Taylor expansion with remainder:

$$|\varepsilon(x)| \le \left| \frac{\mathrm{Z}_{M+K}(x-x_0)}{[\mathrm{T}_K(x-x_0)+\mathrm{R}_K(x)]\cdot \mathrm{Q}_M(x-x_0)} \right| + \left| \frac{\mathrm{R}_K(x)}{H(x)} \right|, \quad H(x) \neq 0.$$

Since $|\mathrm{R}_K(x)|$ has already been estimated by τ, $\mathrm{R}_K(x) \in [-\tau, +\tau]$ holds, and the first term in the sum can be evaluated for $x \in [x_0-\eta, x_0+\eta]$ with intervals, as described in the previous paragraph.

6 The error functions erf(x) and erfc(x)

Figure 1 shows for the error function

$$\mathrm{erf}(x) := \frac{2}{\sqrt{\pi}} \int_0^x e^{-t^2}\, dt$$

and the complementary error function

$$\mathrm{erfc}(x) := 1 - \mathrm{erf}(x) = \frac{2}{\sqrt{\pi}} \int_x^\infty e^{-t^2}\, dt$$

the corresponding graphs. For these functions with real argument x shall be developed fast algorithms with guaranteed error bounds.

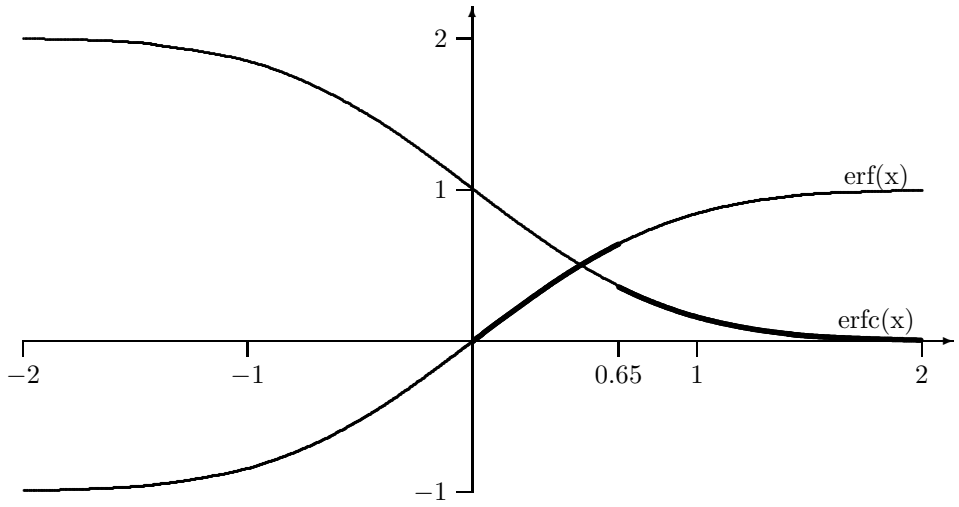


Figure 1: erf(x) and erfc(x)

In the literature (e. g. in [2, 18, 19, 20, 21, 26, 27]) there is a large number of error bounds. But one finds either only absolute error bounds or the relative error is

estimated only asymptotically, so that concrete and guaranteed bounds of the approximation error are not available. If the approximating functions are given by polynomials, then certainly hints on possible extinction effects are made; but an estimate for the error of the polynomial evaluation is not undertaken, so that a guaranteed error bound for the evaluation of erf(x) resp. erfc(x) for a certain data format does not exist.

The algorithms wanted for the functions erf(x) and erfc(x) are implemented in PASCAL-XSC in IEEE-double-format, and it is assumed that the elementary operations are performed only highly accurate (1 ulp accuracy). Together with the calculated error bounds are realized corresponding interval functions.

6.1 The Coarse Algorithm

area	approximation for erf(x)		approximation for erfc(x)
$B_\circ = 0$	0		1
$(B_\circ, B_1]$	0 resp. warning		1
$(B_1, B_2]$	$x \cdot \frac{2}{\sqrt{\pi}}$	$\Longrightarrow$	$1 - x \cdot \frac{2}{\sqrt{\pi}}$
$(B_2, B_3]$	$x \cdot \frac{p_2(x^2) \in P_4}{q_2(x^2) \in P_4}$	$\Longrightarrow$	$1 - x\frac{p_2(x^2)}{q_2(x^2)}$
$(B_3, B_4]$	$1 - e^{-x^2}\frac{p_3(x)}{q_3(x)}$	$\Longleftarrow$	$e^{-x^2}\frac{p_3(x) \in P_5}{q_3(x) \in P_6}$
$(B_4, B_5]$	$1 - e^{-x^2}\frac{p_4(x)}{q_4(x)}$	$\Longleftarrow$	$e^{-x^2}\frac{p_4(x) \in P_5}{q_4(x) \in P_6}$
$(B_5, B_6]$	1		$\frac{1}{x} \cdot e^{-x^2}\frac{p_5(\frac{1}{x^2}) \in P_4}{q_5(\frac{1}{x^2}) \in P_4}$
$> B_6$	1		0 resp. warning
$x < 0$	$\mathrm{erf}(x) = -\mathrm{erf}(\lvert x\rvert)$		$\mathrm{erfc}(x) = 2 - \mathrm{erfc}(\lvert x\rvert)$

The table above shows review–like the various approximations for erf(x) and erfc(x) in the different areas. These are given by the following bounds:

$$\begin{aligned} B_0 &:= 0,\ B_1 := 1.97193 \cdot 10^{-308},\ B_2 := 10^{-10},\ B_3 := 0.65, \\ B_4 &:= 2.2,\ B_5 := 6,\ B_6 := 26.5432\,. \end{aligned}$$

Arrows indicate where the actual elementary approximation is also used slightly modified for the corresponding complementary function. A symbol $p_n(x) \in P_k$ indicates that the numerator polynomial $p_n(x)$ is used in $(B_n, B_{n+1}]$ and has degree k. The symbols for appearing denominator polynomials are to be treated correspondingly.

It is therefore not necessary to give for $\mathrm{erf}(x)$ and $\mathrm{erfc}(x)$ an algorithm for completely all x! Only in the bold–faced areas of Figure 1 have to be given approximating functions for $\mathrm{erf}(x)$ resp. $\mathrm{erfc}(x)$:

$$\begin{aligned} \mathrm{erf}(x) &\approx G(x), & x &\in [0,\, 0.65] & &=: \mathbf{A}, \\ \mathrm{erfc}(x) &\approx g(x), & x &\in [0.65,\, +\infty) & &=: \mathbf{B}. \end{aligned}$$

With the identities

$$\begin{aligned} \mathrm{erf}(x) &\equiv 1 - \mathrm{erfc}(x), & x &\in [0.65,\, +\infty) & &= \mathbf{B}, \\ \mathrm{erfc}(x) &\equiv 1 - \mathrm{erf}(x), & x &\in [0,\, 0.65] & &= \mathbf{A}, \\ \mathrm{erf}(x) &\equiv -\mathrm{erf}(-x), & x &\in (-\infty, 0] & &=: \mathbf{C}, \\ \mathrm{erfc}(x) &\equiv 1 + \mathrm{erf}(-x), & x &\in (-\infty, 0] & &= \mathbf{C} \end{aligned}$$

the functions $\mathrm{erf}(x)$ and $\mathrm{erfc}(x)$ can then be evaluated for all desired arguments x using $G(x)$ and $g(x)$. With respect to $\mathrm{erf}(x)$ resp. $\mathrm{erfc}(x)$ the intervalls $\mathbf{A}$ resp. $\mathbf{B}$ still have to be further partitioned into subintervalls:

$$\begin{aligned} &\mathrm{erf}(x) \text{ in underflow range}, && x \in [0,\, 1.97193 \cdot 10^{-308}\,) && =: \mathbf{A}_0, \\ &\mathrm{erf}(x) \approx G_1(x), && x \in [1.97193 \cdot 10^{-308},\, 10^{-10}] && =: \mathbf{A}_1, \\ &\mathrm{erf}(x) \approx G_2(x), && x \in [10^{-10},\, 0.65] && =: \mathbf{A}_2, \end{aligned}$$

$$\begin{aligned} &\mathrm{erfc}(x) \approx g_1(x), && x \in [0.65,\, 2.2] && =: \mathbf{B}_1, \\ &\mathrm{erfc}(x) \approx g_2(x), && x \in [2.2,\, 6] && =: \mathbf{B}_2, \\ &\mathrm{erfc}(x) \approx g_3(x), && x \in [6,\, 26.5432] && =: \mathbf{B}_3, \\ &\mathrm{erfc}(x) \text{ in underflow range}, && x \in [26.5432,\, +\infty\,) && =: \mathbf{B}_4. \end{aligned}$$

In the error estimates, instead of the interval endpoints which are not representable in number format, are used corresponding confinements of these values. In the subintervals $\mathbf{A}_0$ and $\mathbf{B}_4$ the values of erf(x) and erfc(x) can fall into the denormalized area of the IEEE-double-format, so that the subsequent error estimates referring to the normalized area only are not valid in the denoted intervals. For point arguments in $\mathbf{A}_0$ and $\mathbf{B}_4$ therefore, are produced corresponding error messages while for the interval evaluation of erf(x) and erfc(x) the function values are set to zero.

6.2 Approximation of erf(x) in A=[0, 0.65]

The subinterval $\mathbf{A} = [0,\, 0.65]$ is now subdivided into the two subintervals $\mathbf{A}_1 = [0,\, 10^{-10}]$ and $\mathbf{A}_2 = [10^{-10},\, 0.65]$. First the part where $x \in [0,\, 10^{-10}]$ is examined. Here we proceed from the expansion

$$\mathrm{erf}(x) = \frac{2x}{\sqrt{\pi}} \sum_{n=0}^{\infty} \frac{(-1)^n \cdot x^{2n}}{n!\,(2n+1)} = \frac{2x}{\sqrt{\pi}} \left[1 - \frac{x^2}{3} + \frac{x^4}{10} - + \cdots \right].$$

Since this series [2, Formel 7.1.5] is a Leibniz series for $|x| < 1$, with respect to the approximation

$$G_1(x) := \frac{2}{\sqrt{\pi}} \cdot x \approx \mathrm{erf}(x)$$

the inequality

$$|r(x)| \le \frac{2x^3}{3 \cdot \sqrt{\pi}}$$

is valid for the absolute error $\quad r(x) := \mathrm{erf}(x) - 2x/\sqrt{\pi}$.

Together with

$$\mathrm{erf}(x) > \frac{2x}{\sqrt{\pi}} \cdot \left(1 - \frac{x^2}{3} \right)$$

we obtain for the relative approximation error $\varepsilon_{\mathrm{app}}(x) := r(x)/\mathrm{erf}(x)$ the inequality

$$|\varepsilon_{\mathrm{app}}(x)| \le \frac{x^2}{3 - x^2} \le \frac{10^{-20}}{3 - 10^{-20}} < 3.3334 \cdot 10^{-21} =: \varepsilon(\mathrm{app})\,.$$

In order to obtain with (3) an error bound for erf(x) in $x \in [0,\, 10^{-10}] \cap S(2, 53)$ one needs the error bound for the machine evaluation of $G_1(x)$:

$$\begin{aligned} \widetilde{G}_1(x) &= \left[\widetilde{\frac{2}{\sqrt{\pi}}} \right] \boxdot x = G_1(x) \cdot (1+\varepsilon)(1+2\varepsilon), \qquad |\varepsilon| \le \varepsilon^* = 2^{-53} \\ &= G_1(x) \cdot (1 + \varepsilon_{G1}); \quad |\varepsilon_{G1}| \le 3.3307 \cdot 10^{-16} =: \varepsilon(G_1)\,. \end{aligned}$$

For the estimation above it is assumed that $2/\sqrt{\pi}$ is stored in $S(2, 53)$ with maximal accuracy. With (3) then follows

$$\widetilde{\mathrm{erf}}(x) = \mathrm{erf}(x)(1+\varepsilon_f); \qquad |\varepsilon_f| \le 3.3308 \cdot 10^{-16} =: \varepsilon(f)\,.$$

As the error estimation above is performed with error bound $2\varepsilon^* = 2^{-52}$, it must be assumed that the values of $G_(x)$ lie in the normalized number domain, i. e.

$$\widetilde{G}_1(x) = G_1(x)\cdot(1+\varepsilon_f) \ge G_1(x)\cdot[1-\varepsilon(f)] = \frac{2x}{\sqrt{\pi}}\cdot[1-\varepsilon(f)] \ge 2^{-1022}$$

must hold. The last inequality is fulfilled if $x \ge 1.97193 \cdot 10^{-308}$.

For the subinterval $\mathbf{A}_1 = [1.97193 \cdot 10^{-308},\, 10^{-10}]$ follows the result

$$\boxed{\widetilde{\mathrm{erf}}(x) = \mathrm{erf}(x)(1+\varepsilon_f); \quad |\varepsilon_f| \le 3.3308 \cdot 10^{-16} = \varepsilon(\mathrm{erf},\mathbf{A}_1), \;\; x \in \mathbf{A}_1 \cap S(2,53)\,.}$$

In case $x < 1.97193 \cdot 10^{-308}$ a corresponding error message is generated if $\mathrm{erf}(x)$ is calculated. For the second subinterval $\mathbf{A}_2 := [10^{-10},\, 0.65]$ the expansion

$$\mathrm{erf}(x) = \frac{2x}{\sqrt{\pi}}\cdot e^{-x^2}\cdot\sum_{n=0}^{\infty} a_n\cdot x^{2n}, \qquad a_n = \frac{2^n}{1\cdot 3\cdot\,\ldots\,\cdot(2n+1)},$$

valid for $|x| < +\infty$ is used. Because $a_{n+1}/a_n < 1$ one obtains with

$$\sum_{n=0}^{\infty} a_n\cdot x^{2n} = \sum_{n=0}^{N} a_n\cdot x^{2n} + \sum_{n=N+1}^{\infty} a_n\cdot x^{2n}$$

the inequality

$$\begin{aligned}\sum_{n=N+1}^{\infty} a_n\cdot x^{2n} &= a_{N+1}\cdot x^{2(N+1)}\left[1+\frac{a_{N+2}}{a_{N+1}}\cdot x^2+\frac{a_{N+3}}{a_{N+1}}\cdot x^4+\cdots\right]\\ &< a_{N+1}\cdot x^{2(N+1)}\sum_{n=0}^{\infty} x^{2n} = \frac{a_{N+1}\cdot x^{2(N+1)}}{1-x^2}, \quad |x|<1\,.\end{aligned}$$

With the auxiliary approximation

$$\mathrm{erf}(x) \approx H(x) := \frac{2x}{\sqrt{\pi}}\cdot e^{-x^2}\sum_{n=0}^{N} a_n\cdot x^{2n}$$

and the inequality

$$\mathrm{erf}(x) \ge \frac{2x}{\sqrt{\pi}}\cdot e^{-x^2}, \quad x \ge 0\,,$$

due to the series expansion, follows for the relative approximation error

$$\begin{aligned}\varepsilon_{\mathrm{app},1} &:= \frac{\mathrm{erf}(x)-H(x)}{\mathrm{erf}(x)};\\ |\varepsilon_{\mathrm{app},1}| &< \frac{2^{N+1}\cdot x^{2(N+1)}}{1\cdot 3\cdot\,\ldots\,\cdot(2N+3)}\cdot\frac{1}{1-x^2} \le \varepsilon(\mathrm{app},1), \quad |x|<1.\end{aligned}$$

For $N = 14$ therefore in $\mathbf{A}_2 = [10^{-10}, 0.65]$ we have

$$\operatorname{erf}(x) \approx H(x) = \frac{2x}{\sqrt{\pi}} \cdot e^{-x^2} \sum_{n=0}^{14} a_n \cdot x^{2n}\,, \qquad \varepsilon(\mathrm{app}, 1) = 7.2149 \cdot 10^{-19}\,.$$

The auxiliary approximating function $H(x)$ in IEEE–Format could actually be used as a machine approximation for $\operatorname{erf}(x)$, but the runtime would be very high because of the appearing exponential function and the high polynomial degree $N = 14$! A lot more effective is the use of a second approximation $G_2(x)$ with

$$\operatorname{erf}(x) \approx x \cdot \frac{\mathrm{P}_4(x^2)}{\mathrm{Q}_4(x^2)} =: G_2(x), \qquad \mathrm{P}_4(x^2) = \sum_{n=0}^{4} p_n \cdot x^{2n}, \quad \mathrm{Q}_4(x^2) = \sum_{n=0}^{4} q_n \cdot x^{2n},$$

n	$p_n :=$ nearest(.)	$q_n :=$ nearest(.)
0	$1.12837916709551256 \cdot 10^{+0}$	$1.00000000000000000 \cdot 10^{+0}$
1	$1.35894887627277916 \cdot 10^{-1}$	$4.53767041780002545 \cdot 10^{-1}$
2	$4.03259488531795274 \cdot 10^{-2}$	$8.69936222615385890 \cdot 10^{-2}$
3	$1.20339380863079457 \cdot 10^{-3}$	$8.49717371168693357 \cdot 10^{-3}$
4	$6.49254556481904354 \cdot 10^{-5}$	$3.64915280629351082 \cdot 10^{-4}$

The decimals in the table above were determined by rational approximation of the auxiliary function $H(x)$ using a computer algebra system. The coefficients $p_n.q_n$ are the IEEE–numbers nearest to the decimals given. For determining the approximation error one may then assume exactly representable approximation coefficients.

For $G_2(x)$ the relative approximation error is given by

$$\varepsilon_{\mathrm{app},2}(x) := \frac{H(x) - G_2(x)}{H(x)}, \qquad x \in \mathbf{A}_2 = [10^{-10}, 0.65].$$

Using the XSC–programs `AppErr` and `ErrBound` a guaranteed upper bound $\varepsilon(\mathrm{app}, 2)$ for $|\varepsilon_{\mathrm{app},2}(x)|$ corresponding to $x \in \mathbf{A}_2$ can be calculated. One obtains

$$H(x) \approx G_2(x) = x \cdot \frac{\mathrm{P}_4(x^2)}{\mathrm{Q}_4(x^2)}; \quad |\varepsilon_{\mathrm{app},2}(x)| \le 1.3594 \cdot 10^{-17} = \varepsilon(\mathrm{app}, 2)\,.$$

With the error bounds $\varepsilon(\mathrm{app}, 1)$, $\varepsilon(\mathrm{app}, 2)$ now available one is now in the position to calculate according to equation (2) an upper bound for the corresponding approximation error

$$\varepsilon_{\mathrm{app}}(x) := \frac{\operatorname{erf}(x) - G_2(x)}{\operatorname{erf}(x)}; \qquad |\varepsilon_{\mathrm{app}}(x)| \le \varepsilon(\mathrm{app}), \quad x \in \mathbf{A}_2 = [10^{-10}, 0.65]$$

with respect to the approximation

$$\operatorname{erf}(x) \approx G_2(x) = x \cdot \frac{\mathrm{P}_4(x^2)}{\mathrm{Q}_4(x^2)}.$$

This results in

$$\boxed{\operatorname{erf}(x) \approx G_2(x) = x \cdot \frac{\mathrm{P}_4(x^2)}{\mathrm{Q}_4(x^2)}; \qquad |\varepsilon_{\mathrm{app}}(x)| \le 1.4316 \cdot 10^{-17} =: \varepsilon(\mathrm{app})\,.}$$

In order to calculate with equation (1) an error bound of $\operatorname{erf}(x)$ in $x \in \mathbf{A}_2 \cap S(2,53)$, we need the relative error bound of the machine evaluation of $G_2(x)$. For this is used that

$$\widetilde{G}_2(x) = x \boxdot \widetilde{\mathrm{P}}_4(x \boxdot x) \boxplus \widetilde{\mathrm{Q}}_4(x \boxdot x) = G_2(x)(1+\varepsilon_{G2}). \tag{15}$$

The error bound wanted is again calculated automatically. For the numerator polynomial follows

$$\boxed{\widetilde{\mathrm{P}}_4(x \boxdot x) = \mathrm{P}_4(x^2)(1+\varepsilon_{P4}); \qquad |\varepsilon_{P4}| \le 2.6230 \cdot 10^{-16} = \varepsilon(\mathrm{P}_4)\,.}$$

For the denominator polynomial one finds

$$\boxed{\widetilde{\mathrm{Q}}_4(x \boxdot x) = \mathrm{Q}_4(x^2)(1+\varepsilon_{Q4}); \qquad |\varepsilon_{Q4}| \le 3.4600 \cdot 10^{-16} = \varepsilon(\mathrm{Q}_4)\,.}$$

For $\widetilde{G}_2(x)$ according to equation (15) we have:

$$\widetilde{G}_2(x) = \frac{x \cdot \mathrm{P}_4(x^2)(1+\varepsilon_{P4})(1+2\varepsilon)^2}{\mathrm{Q}_4(x^2)(1+\varepsilon_{Q4})} = G_2(x)\frac{(1+\varepsilon_{P4})(1+2\varepsilon)^2}{(1+\varepsilon_{Q4})} = G_2(x)(1+\varepsilon_{G2})\,.$$

The error bounds $\varepsilon(\mathrm{P}_4), \varepsilon(\mathrm{Q}_4)$ and $|\varepsilon| \le \varepsilon^* = 2^{-53}$ finally imply for $|\varepsilon_{G2}|$

$$\boxed{\widetilde{G}_2(x) = G_2(x)(1+\varepsilon_{G2}); \qquad |\varepsilon_{G2}| \le 1.0524 \cdot 10^{-15} = \varepsilon(G_2)\,.}$$

With respect to $\widetilde{\operatorname{erf}}(x) = \operatorname{erf}(x)(1+\varepsilon_f)$ one finds at last an estimation for $|\varepsilon_f|$ in $\mathrm{A}_2 = [10^{-10}, 0.65]$ using equation (3):

$$\boxed{\widetilde{\operatorname{erf}}(x) = \operatorname{erf}(x)(1+\varepsilon_f); \quad |\varepsilon_f| \le 1.0668 \cdot 10^{-15} = \varepsilon(\operatorname{erf}, \mathbf{A}_2),\ x \in \mathbf{A}_2 \cap S(2,53)\,.}$$

6.3 Error Bound for erfc(x) in A= [0, 0.65]

Using the machine approximation $\widetilde{\mathrm{erf}}(x)$ determined in the previous section and the corresponding relative error bound, now $\mathrm{erfc}(x) \equiv 1 - \mathrm{erf}(x)$ is calculated on the machine as follows:

$$\widetilde{\mathrm{erfc}}(x) := \begin{cases} 1 & : \quad x \in \mathbf{A}_0 = [0,\, 1.97193 \cdot 10^{-308}\,) \\ 1 \boxminus \widetilde{\mathrm{erf}}(x) & : \quad x \in \mathbf{D} := \mathbf{A}_1 \cup \mathbf{A}_2 = [1.97193 \cdot 10^{-308},\, 0.65] \end{cases} .$$

First the subinterval $\mathbf{D} = [1.97193 \cdot 10^{-308},\, 0.65]$ is considered. With

$$\widetilde{\mathrm{erfc}}(x) := 1 \boxminus \widetilde{\mathrm{erf}}(x) = \mathrm{erfc}(x) \cdot (1 + \varepsilon)$$

one obtains for $|\varepsilon|$ the inequality

$$\begin{aligned} |\varepsilon| &\leq 2 \cdot \varepsilon^* + [1 + 2 \cdot \varepsilon^*] \cdot \frac{\max\limits_{x \,\in\, \mathbf{D}} \{\mathrm{erf}(x)\} \cdot \varepsilon(\mathrm{erf}, \mathbf{A}_2)}{\min\limits_{x \,\in\, \mathbf{D}} \{\mathrm{erfc}(x)\}}; \quad \varepsilon^* = 2^{-53} \\ &= 2 \cdot \varepsilon^* + [1 + 2 \cdot \varepsilon^*] \cdot \frac{\mathrm{erf}(0.65) \cdot \varepsilon(\mathrm{erf}, \mathbf{A}_2)}{1 - \mathrm{erf}(0.65)} . \end{aligned}$$

For estimating the right hand side one needs an upper bound for the value erf(0.65). According to the definition of $\varepsilon_{\mathrm{app},1}$ (compare page 20) for $x \in \mathbf{D}$ follows

$$\mathrm{erf}(x) = \frac{H(x)}{1 - \varepsilon_{\mathrm{app},1}} \leq \frac{H(x)}{1 - \varepsilon(\mathrm{app}, 1)} = \frac{2}{\sqrt{\pi}} \cdot \frac{x \cdot e^{-x^2}}{1 - \varepsilon(\mathrm{app}, 1)} \cdot \sum_{n=0}^{14} a_n \cdot x^{2n} .$$

If one evaluates the term after the last equal sign for $x = [0.65, 0.65]$ with intervals, then one obtains a confinement of the upper bound wanted for erf(0.65). The following XSC–program delivers a guaranteed upper bound of the expression $\frac{\mathrm{erf}(0.65)}{1-\mathrm{erf}(0.65)}$. The upper bound OS has the numerical value

$$\frac{\mathrm{erf}(0.65)}{\mathrm{erfc}(0.65)} < 1.793525 =: \mathrm{OS} .$$

```
program OS;
{*****************************************}
{*  The confinement of an upper bound for  *}
{*  erf(0.65)/( 1-erf(0.65) is calculated  *}
{*****************************************}
use i_ari;
var a                : array[0..14] of interval;
    k                : integer;
    c, x, x2, eps, t : interval;
begin
  a[0] := 1;
  for k := 1 to 14 do
```

```
  a[k] := 2 * a[k-1] / (2*k + 1);
  c := 1 / SQRT( arctan(intval(1)) );
  { c: confinement of 2/SQRT(Pi) }
  x := 65 / intval(100);
  x2 := x * x;
  eps := intval( (<7.2149e-19),(>7.2149e-19) );
  t := a[14];                { Horner-scheme for }
  for k := 13 downto 0 do    { calculating the   }
     t := t * x2 + a[k];     { sum               }
  t := c * x * exp(-x2) * t / (1 - eps);
  t := t / (1 - t);
  { t: confinement of an upper bound }
  { of erf(0.65)/( 1-erf(0.65) )     }
  writeln(sup(t));
end.
```

For $|\varepsilon|$ one obtains with upper limit $\mathrm{OS} = 1.793525$ the estimation

$$|\varepsilon| \leq 2 \cdot \varepsilon^* + [1 + 2 \cdot \varepsilon^*] \cdot \mathrm{OS} \cdot \varepsilon(\mathrm{erf}, \mathbf{A}_2) < 2.1354 \cdot 10^{-15} = \varepsilon(\mathrm{erfc}, \mathbf{D}) \,,$$

i. e.

$$\widetilde{\mathrm{erfc}}(x) = \mathrm{erfc}(x)(1+\varepsilon); \quad |\varepsilon| \leq 2.1354 \cdot 10^{-15} = \varepsilon(\mathrm{erfc}, \mathbf{D}),\ x \in \mathbf{D} \cap S(2, 53) \,.$$

For the remaining second subinterval $\mathbf{A}_0 = [0,\, 1.97193 \cdot 10^{-308}\,)$ one finds for the relative approximation error

$$\varepsilon_{\mathrm{app}}(x) := \frac{1 - \mathrm{erfc}(x)}{\mathrm{erfc}(x)} = \frac{1}{\mathrm{erfc}(x)} - 1 = \frac{\mathrm{erf}(x)}{1 - \mathrm{erf}(x)} < \frac{\mathrm{erf}(1.97193 \cdot 10^{-308})}{1 - \mathrm{erf}(1.97193 \cdot 10^{-308})} \,.$$

The series expansion of $\mathrm{erf}(x)$ on page 19 shows that the relative approximation error is of the order 10^{-308}. As in this subinterval is used as approximating function the (exactly representable) constant 1, no additional evaluation error appears, so the relative error with respect to $\mathrm{erfc}(x)$ in this subinterval is certainly smaller than $\varepsilon(\mathrm{erfc}, \mathbf{D}) = 2.1354 \cdot 10^{-15}$. Thus, it is shown that

$$\boxed{\widetilde{\mathrm{erfc}}(x) = \mathrm{erfc}(x)(1+\varepsilon); \quad |\varepsilon| \leq 2.1354 \cdot 10^{-15} = \varepsilon(\mathrm{erfc}, \mathbf{A}),\ x \in \mathbf{A} \cap S(2, 53).}$$

6.4 Approximation of erfc(x) in $\mathbf{B}_1 \cup \mathbf{B}_2 = [0.65,\, 6]$

In this section the function $\mathrm{erfc}(x)$ is approximated in the interval $[0.65,\, 6]$ by an auxiliary function $H(x)$ which can be realized by a standard function. For the corresponding approximation error a guaranteed upper limit is developed.
First, according to [2, 7.4.11], for $x > 0$ we have the integral

$$\mathrm{erfc}(x) = \frac{2}{\sqrt{\pi}} \int_x^\infty e^{-t^2} dt \equiv \frac{2x \cdot e^{-x^2}}{\pi} \int_0^\infty \frac{e^{-t^2}}{t^2 + x^2}\, dt, \quad x > 0 \,.$$

If the trapezoid rule of numerical integration is applied to the last integral, one receives [19, S. 114], [21, S. 134]:

$$\operatorname{erfc}(x) = \mathrm{T}(x,h) + \mathrm{P}(x,h), \qquad x > 0,$$

$$\begin{aligned}
\mathrm{T}(x,h) &:= \frac{2x\cdot h}{\pi}\cdot e^{-x^2}\left[\frac{1}{2x^2} + \sum_{k=1}^{\infty}\frac{e^{-k^2h^2}}{k^2h^2+x^2}\right], \\
\mathrm{P}(x,h) &:= -\sum_{r=1}^{\infty} G_r, \qquad G_r := \frac{4x\cdot e^{-x^2}}{\pi}\int_0^{\infty}\frac{e^{-t^2}\cdot\cos\,\omega t}{t^2+x^2}\,dt, \quad \omega := \frac{2\pi r}{h}.
\end{aligned}$$

According to [24] the Fourier–integral G_r can be written as

$$G_r = e^{\omega x}\cdot\operatorname{erfc}(x+\frac{\omega}{2}) + e^{-\omega x}\cdot\operatorname{erfc}(x-\frac{\omega}{2}); \qquad \omega := \frac{2\pi r}{h}\,. \tag{16}$$

Using the auxiliary approximation function

$$H(x) := \frac{2x\cdot h}{\pi}\cdot e^{-x^2}\left[\frac{1}{2x^2} + \sum_{k=1}^{N}\frac{e^{-k^2h^2}}{h^2k^2+x^2}\right] \approx \operatorname{erfc}(x)$$

the absolute value of the absolute approximation error Δ can be estimated by

$$|\Delta| := |H(x) - \operatorname{erfc}(x)| \le \frac{2xh\cdot e^{-x^2}}{\pi}\cdot\sum_{k=N+1}^{\infty}\frac{e^{-h^2k^2}}{h^2k^2+x^2} + |\mathrm{P}(x,h)| \tag{17}$$

$$\sum_{k=N+1}^{\infty}\frac{e^{-h^2k^2}}{h^2k^2+x^2} < \sum_{k=N+1}^{\infty}\frac{e^{-h^2k^2}}{h^2k^2} < \frac{1}{h^2(N+1)^2}\sum_{k=N+1}^{\infty}e^{-h^2k^2};$$

$$\begin{aligned}
\sum_{k=N+1}^{\infty}e^{-h^2k^2} &= \sum_{k=0}^{\infty}e^{-h^2([N+1]+k)^2} = e^{-h^2(N+1)^2}\cdot\sum_{k=0}^{\infty}e^{-h^2k(2[N+1]+k)} \\
&< e^{-h^2(N+1)^2}\cdot\sum_{k=0}^{\infty}e^{-4h^2\cdot k} = \frac{e^{-h^2(N+1)^2}}{1-e^{-4h^2}}\,.
\end{aligned}$$

Thus

$$|\Delta| \le \frac{2x\cdot e^{-x^2}}{\pi h\cdot(N+1)^2}\cdot\frac{e^{-h^2(N+1)^2}}{1-e^{-4h^2}} + |\mathrm{P}(x,h)|\,. \tag{18}$$

For estimating $|\mathrm{P}(x,h)|$ one uses because of $\operatorname{erfc}(x) < 2$ first the inequality

$$G_r < e^{\omega x}\cdot\operatorname{erc}(x+\frac{\omega}{2}) + 2\cdot e^{-\omega x} < e^{\omega x}\cdot\operatorname{erfc}(\frac{\omega}{2}) + 2\cdot e^{-\omega x}\,,$$

from which follows with [2, Formel 7.1.13]

$$\operatorname{erfc}\left(\frac{\omega}{2}\right) < \frac{2\cdot e^{-\omega^2/4}}{\sqrt{\pi}\cdot\omega}$$

as well as with $\omega = 2\pi r/h$ and $r \geq 1$, that

$$G_r < \frac{h}{\pi^{3/2}} \cdot e^{-\frac{2\pi r}{h}(\frac{\pi}{2h} - x)} + 2 \cdot e^{-\frac{2\pi x}{h} \cdot r}, \qquad r = 1, 2, \ldots . \tag{19}$$

The necessary summation over r can now be performed under the assumption $2h \cdot x < \pi$ using the geometric series. It follows that

$$|\mathrm{P}(x,h)| < \frac{h}{\pi^{3/2}} \cdot \frac{1}{e^{\frac{2\pi}{h}(\frac{\pi}{2h} - x)} - 1} + \frac{2}{e^{\frac{2\pi \cdot x}{h}} - 1}; \qquad 2h \cdot x < \pi\,. \tag{20}$$

In order to calculate a **relative** error bound, (18) has now only be divided by a lower bound of $\mathrm{erfc}(x)$. With inequality [21, S. 137,(1)]

$$\mathrm{erfc}(x) > \frac{2x}{\sqrt{\pi}} \cdot \frac{e^{-x^2}}{1 + 2x^2}, \qquad x > 0$$

one finally obtains together with (18) and (20) the results:

$$\begin{aligned} \mathrm{erfc}(x) &\approx H(x) = \frac{2xh}{\pi} \cdot e^{-x^2} \left[\frac{1}{2x^2} + \sum_{k=1}^{N} \frac{e^{-h^2k^2}}{h^2k^2 + x^2} \right]; \\ \mathrm{erfc}(x) &= H(x)(1 + \varepsilon_{\mathrm{app},1}); \qquad x \in [0.65,\, 6]; \end{aligned}$$

$$\begin{aligned} \mathrm{U}_1(x,h,N) &:= \frac{1 + 2x^2}{\sqrt{\pi} \cdot h \cdot (N+1)^2} \cdot \frac{e^{-h^2(N+1)^2}}{1 - e^{-4h^2}}; \\ \mathrm{U}_2(x,h) &:= \frac{h \cdot (1 + 2x^2) \cdot e^{x^2}}{2\pi x \cdot \left[e^{\frac{2\pi}{h}(\frac{\pi}{2h} - x)} - 1 \right]}, \qquad 2hx < \pi; \\ \mathrm{U}_3(x,h) &:= \frac{\sqrt{\pi} \cdot (1 + 2x^2) \cdot e^{x^2}}{x} \cdot \frac{1}{e^{\frac{2\pi x}{h}} - 1}; \end{aligned}$$

$$\boxed{\begin{array}{c} |\varepsilon_{\mathrm{app},1}| < \mathrm{U}_1(x,h,N) + \mathrm{U}_2(x,h) + \mathrm{U}_3(x,h) \leq \varepsilon(\mathrm{app},1;x,h,N); \\ x \in [0.65,\, 6]; \qquad 2hx < \pi\,. \end{array}}$$

Finally, in the functions U_i the parameters x, h, N have to be chosen such that one obtains with respect to IEEE–format a relative error bound which is of order 10^{-18} for all $x \in [0.65,\, 6]$.

For $\mathrm{U}_3(x,h)$ one therefore seeks for given $h > 0$ an upper bound for the global maximum with respect to $x \in [0.65,\, 6]$.

For calculating a **guaranteed** upper bound the XSC–program for one–dimensional global optimization in [9] can be used, in which the function now denoted with `u3` is to be defined as follows:

```
function u3 (x : DerivType) : DerivType;
var h,pi2,sqrtpi : DerivType;
    c1,c2    : interval;
begin
  h := DerivConst( 93/intval(1000) );
  c1 := ARCTAN( intval(1) );              { pi/4     }
  pi2 := DerivConst( 8*c1 );              { 2*pi     }
  sqrtpi := DerivConst( 2*SQRT(c1) );     { sqrt(pi) }
  u3 := -sqrtpi*(1+2*x*x)*exp(x*x)/(x* (exp(pi2*x/h)-1) );
end;
```

With $h = 0.093$ on obtains the result

$$\mathrm{U}_3(x, 0.093) \le 6.5046 \cdot 10^{-19}, \qquad x \in [0.65,\, 6]\,.$$

Accordingly, also an upper bound for the function $\mathrm{U}_2(x, h)$ can be calculated with the program for global optimization. For preventing overflow, however, the estimation

$$\mathrm{U}_2(x,h) := \frac{h\cdot(1+2x^2)\cdot e^{x^2}}{2\pi x\cdot\left[e^{\frac{2\pi}{h}(\frac{\pi}{2h}-x)}-1\right]} < \frac{h\cdot(1+2x^2)\cdot e^{x^2}}{2\pi x\cdot\left[e^{\frac{2\pi}{h}(\frac{\pi}{2h}-8)}-1\right]}, \quad 2hx<\pi$$

is performed before. For the function `u2`

```
function u2 (x : DerivType) : DerivType;
var h  : DerivType;
    pi : interval;
begin
  h := DerivConst( 93/intval(1000) );
  pi :=4*ARCTAN( intval(1) );               { pi }
  u2 := -h*(1+2*x*x)*exp(x*x)/
        ( 2*pi*x*(exp( 2*pi/h*(pi/(2*h)-8) )-1) )
end;
```

then follows with $h = 0.093$ the upper bound

$$\mathrm{U}_2(x, 0.093) \le 1.0877 \cdot 10^{-246}, \qquad x \in [0.65,\, 6]\,.$$

For estimating $\mathrm{U}_1(x, h, N)$ the function `u1`

```
function u1 (x : DerivType) : DerivType;
var h      : DerivType;
    sqrtpi : interval;
    N      : integer;
begin
  h := DerivConst( 93/intval(1000) );
  sqrtpi :=2*SQRT( ARCTAN(intval(1)) );   { sqrt(pi) }
  N := 70,
  u1 := -(1+2*x*x)*exp(-h*h*(N+1)*(N+1))/
        ( sqrtpi*h*(N+1)*(N+1)*(1-exp(-4*h*h)) )
end;
```

with $h = 0.093$ and $N = 70$ is used. This results in

$$\mathrm{U}_1(x, 0.093, 70) \le 3.0002 \cdot 10^{-19}, \qquad x \in [0.65,\ 6].$$

The calculated global maxima of the three functions U_i now allow to give a guaranteed upper bound $\varepsilon(\mathrm{app}, 1)$ of the relative approximation error for the data set $x \in [0.65, 6]$:

$$\begin{aligned} \mathrm{erfc}(x) &\approx H(x) = \frac{2xh}{\pi} \cdot e^{-x^2} \left[\frac{1}{2x^2} + \sum_{k=1}^{N} \frac{e^{-h^2k^2}}{h^2k^2 + x^2} \right]; \\ \mathrm{erfc}(x) &= H(x)(1 + \varepsilon_{\mathrm{app},1}); \qquad x \in [0.65,\ 6]; \end{aligned}$$

$$|\varepsilon_{\mathrm{app},1}| < 9.5049 \cdot 10^{-19} = \varepsilon(\mathrm{app}, 1), \quad h = 0.093, \quad N = 70\,.$$

For the actual approximation of $\mathrm{erfc}(x)$ on the calculator, the Interval [0.65, 6] is partitioned again into two subintervals. In each of these intervals the auxiliary function $H(x)$ is used for finding an efficient approximation with guaranteed relative error bound.

erfc(x) **in** $\mathbf{B}_1 = [0.65,\ 2.2]$

First is considered the interval $\mathbf{B}_1 = [0.65, 2.2]$. Here, $\mathrm{erfc}(x)$ is approximated on the calculator by

$$\mathrm{erfc}(x) \approx e^{-x^2} \cdot \frac{\mathrm{P}_5(x)}{\mathrm{Q}_6(x)} =: g_1(x); \qquad \mathrm{P}_5(x) = \sum_{n=0}^{5} p_n \cdot x^n, \quad \mathrm{Q}_6(x) = \sum_{n=0}^{6} q_n \cdot x^n.$$

n	$p_n :=$ nearest(.)	$q_n :=$ nearest(.)
0	$9.99999992049799098 \cdot 10^{-1}$	$1.00000000000000000 \cdot 10^{+0}$
1	$1.33154163936765307 \cdot 10^{+0}$	$2.45992070144245533 \cdot 10^{+0}$
2	$8.78115804155881782 \cdot 10^{-1}$	$2.65383972869775752 \cdot 10^{+0}$
3	$3.31899559578213215 \cdot 10^{-1}$	$1.61876655543871376 \cdot 10^{+0}$
4	$7.14193832506776067 \cdot 10^{-2}$	$5.94651311286481502 \cdot 10^{-1}$
5	$7.06940843763253131 \cdot 10^{-3}$	$1.26579413030177940 \cdot 10^{-1}$
6		$1.25304936549413393 \cdot 10^{-2}$

Table 2: Polynomial coefficients with 18 decimals

The given decimal values of the approximation to $H(x)$ were determined again with a computer algebra system. The actually used coefficients p_n, q_n are the IEEE–numbers nearest to each of them.

An upper bound $|\varepsilon_{\text{app},2}(x)|$ for the relative approximation error

$$\varepsilon_{\text{app},2}(x) := \frac{H(x) - g_1(x)}{H(x)}, \qquad x \in \mathbf{B}_1 = [0.65,\, 2.2]$$

can be again determined automatically. One finds

$$H(x) \approx g_1(x) := e^{-x^2} \cdot \frac{\mathrm{P}_5(x)}{\mathrm{Q}_6(x)}; \quad |\varepsilon_{\text{app},2}(x)| \leq 1.5772 \cdot 10^{-16} =: \varepsilon(\text{app}, 2)\,.$$

With the now available error bounds $\varepsilon(\text{app}, 1)$, $\varepsilon(\text{app}, 2)$ one is now able to calculate with respect to the approximation

$$g_1(x) := e^{-x^2} \cdot \frac{\mathrm{P}_5(x)}{\mathrm{Q}_6(x)} \approx \operatorname{erfc}(x)$$

accordin to equation (2) an upper bound for the corresponding approximation error

$$\varepsilon_{\text{app}}(x) := \frac{\operatorname{erfc}(x) - g_1(x)}{\operatorname{erfc}(x)}; \qquad |\varepsilon_{\text{app}}(x)| \leq \varepsilon(\text{app}), \quad x \in \mathbf{B}_1 = [0.65,\, 2.2].$$

It follows

$$\boxed{\operatorname{erfc}(x) \approx g_1(x) = e^{-x^2} \cdot \frac{\mathrm{P}_5(x)}{\mathrm{Q}_6(x)}; \qquad |\varepsilon_{\text{app}}(x)| \leq 1.5868 \cdot 10^{-16} = \varepsilon(\text{app})\,.}$$

In order to be able to calculate with equation (3) an error bound of $\operatorname{erfc}(x)$ in $x \in \mathbf{B}_1 \cap S(2, 53)$, one still needs the relative error bound for the evaluation of $g_1(x)$ on the machine:

$$\begin{aligned}
\widetilde{g}_1(x) &:= \text{EXPx2}(x) \boxdot \left[\widetilde{\mathrm{P}}_5(x) \boxslash \widetilde{\mathrm{Q}}_6(x)\right] \\
&= e^{-x^2} \cdot (1 + \varepsilon_1) \cdot \frac{\widetilde{\mathrm{P}}_5(x)}{\widetilde{\mathrm{Q}}_6(x)} \cdot (1 + 2 \cdot \varepsilon)^2, \quad |\varepsilon_1| \leq \varepsilon(1) \\
&= e^{-x^2} \cdot (1 + \varepsilon_1) \cdot \frac{\mathrm{P}_5(x) \cdot (1 + \varepsilon_{P5})}{\mathrm{Q}_6(x) \cdot (1 + \varepsilon_{Q6})} \cdot (1 + 2 \cdot \varepsilon)^2, \qquad (21) \\
&\qquad |\varepsilon| \leq \varepsilon^* = 2^{-53}, \quad |\varepsilon_{P5}| \leq \varepsilon(\mathrm{P}_5), \quad |\varepsilon_{Q6}| \leq \varepsilon(\mathrm{Q}_6)\,.
\end{aligned}$$

If $\text{EXPx2}(x)$ denotes the e^{-x^2}-function, then according to page 42 is valid with highly accurate arithmetics:

$$\text{EXPx2}(x) = e^{-x^2} \cdot (1 + \varepsilon_1), \qquad |\varepsilon_1| \leq 1.0823 \cdot 10^{-15} =: \varepsilon(1)\,.$$

The calculation of an error bound $\varepsilon(\mathrm{P}_5)$ for the numerator polynomial can again be performed automatically, which results in

$$\widetilde{\mathrm{P}}_5(x) = \mathrm{P}_5(x)(1 + \varepsilon_{P5}); \qquad |\varepsilon_{P5}| \leq 1.2027 \cdot 10^{-15} = \varepsilon(\mathrm{P}_5).$$

Accordingly one finds as error bound $\varepsilon(\mathrm{Q}_6)$ for the denominator polynomial

$$\widetilde{\mathrm{Q}}_6(x) = \mathrm{Q}_6(x)(1+\varepsilon_{Q6}); \qquad |\varepsilon_{Q6}| \le 1.5838 \cdot 10^{-15} = \varepsilon(\mathrm{Q}_6)\,.$$

For the according to equation (21) valid property

$$\widetilde{g}_1(x) = g_1(x) \cdot (1+\varepsilon_1) \cdot \frac{1+\varepsilon_{P5}}{1+\varepsilon_{Q6}} \cdot (1+2\cdot\varepsilon)^2 = g_1(x) \cdot (1+\varepsilon_{g1})$$

follows

$$\widetilde{g}_1(x) = g_1(x)(1+\varepsilon_{g1}); \qquad |\varepsilon_{g1}| \le 4.3129 \cdot 10^{-15} = \varepsilon(g_1).$$

With respect to $\widetilde{\mathrm{erfc}}(x) = \mathrm{erfc}(x)(1+\varepsilon_f)$ one finds using equation (3) and the error bounds $\varepsilon(\mathrm{app}) = 1.5868{\cdot}10^{-16}$, $\varepsilon(g_1) = 4.3129{\cdot}10^{-15}$ for $|\varepsilon_f|$ the final estimation

$$\widetilde{\mathrm{erfc}}(x) = \mathrm{erfc}(x)(1{+}\varepsilon_f); \quad |\varepsilon_f| \le 4.4716{\cdot}10^{-15} = \varepsilon(\mathrm{erfc},\mathbf{B}_1), \;\; x \in \mathbf{B}_1{\cap}S(2,53)\,.$$

erfc(x) **in** $\mathbf{B}_2 = [2.2,\, 6]$

In the interval $\mathbf{B}_2 = [2.2,\, 6]$ $\mathrm{erfc}(x)$ is approximated on the calculator by

$$\mathrm{erfc}(x) \approx e^{-x^2} \cdot \frac{\mathrm{P}_5(x)}{\mathrm{Q}_6(x)} =: g_2(x); \qquad \mathrm{P}_5(x) = \sum_{n=0}^{5} p_n \cdot x^n, \quad \mathrm{Q}_6(x) = \sum_{n=0}^{6} q_n \cdot x^n.$$

n	$p_n :=$ nearest(.)	$q_n :=$ nearest(.)
0	$9.99921140009714409 \cdot 10^{-1}$	$1.00000000000000000 \cdot 10^{+0}$
1	$1.62356584489366647 \cdot 10^{+0}$	$2.75143870676376208 \cdot 10^{+0}$
2	$1.26739901455873222 \cdot 10^{+0}$	$3.37367334657284535 \cdot 10^{+0}$
3	$5.81528574177741135 \cdot 10^{-1}$	$2.38574194785344389 \cdot 10^{+0}$
4	$1.57289620742838702 \cdot 10^{-1}$	$1.05074004614827206 \cdot 10^{+0}$
5	$2.25716982919217555 \cdot 10^{-2}$	$2.78788439273628983 \cdot 10^{-1}$
6		$4.00072964526861362 \cdot 10^{-2}$

Table 3: Polynomial coefficients with 18 decimals

The decimal values of table 3 were determined by approximating $H(x)$ with a computer algebra system. The actually used coefficients p_n, q_n are the IEEE–numbers nearest to each of the given decimal values.

For the approximation error

$$\varepsilon_{\mathrm{app},2}(x) := \frac{H(x) - g_2(x)}{H(x)}, \qquad x \in \mathbf{B}_2 = [2.2,\, 6]$$

the upper bound $|\varepsilon_{\mathrm{app},2}(x)|$ with respect to $x \in \mathrm{B}_2$ can be calculated again automatically. One finds

$$H(x) \approx g_2(x) := e^{-x^2} \cdot \frac{\mathrm{P}_5(x)}{\mathrm{Q}_6(x)}; \quad |\varepsilon_{\mathrm{app},2}(x)| \leq 1.5282 \cdot 10^{-16} =: \varepsilon(\mathrm{app}, 2)\,.$$

With the now available error bounds $\varepsilon(\mathrm{app}, 1)$, $\varepsilon(\mathrm{app}, 2)$ one is able to calculate for the approximation

$$\mathrm{erfc}(x) \approx g_2(x) = e^{-x^2} \cdot \frac{\mathrm{P}_5(x)}{\mathrm{Q}_6(x)}$$

according to equation (2) an upper bound for the corresponding relative approximation error

$$\varepsilon_{\mathrm{app}}(x) := \frac{\mathrm{erfc}(x) - g_2(x)}{\mathrm{erfc}(x)}; \qquad |\varepsilon_{\mathrm{app}}(x)| \leq \varepsilon(\mathrm{app}), \quad x \in \mathbf{B}_2 = [2.2,\, 6],$$

namely:

$$\mathrm{erfc}(x) \approx g_2(x) = e^{-x^2} \cdot \frac{\mathrm{P}_5(x)}{\mathrm{Q}_6(x)}; \qquad |\varepsilon_{\mathrm{app}}(x)| \leq 1.5378 \cdot 10^{-16} = \varepsilon(\mathrm{app})\,.$$

Before being able to calculate with equation (3) an error bound for $\mathrm{erfc}(x)$ in $x \in \mathbf{B}_2 \cap S(2, 53)$, a relative error bound for the machine evaluation of $g_2(x)$ must be determined, whereby it shall be assumed that the factor e^{-x^2} is evaluated using the function EXPx2(...). One finds

$$\begin{aligned}
\widetilde{g}_2(x) &:= \mathrm{EXPx2}(x) \boxdot \left[\widetilde{\mathrm{P}}_5(x) \boxslash \widetilde{\mathrm{Q}}_6(x)\right] \\
&= e^{-x^2} \cdot (1 + \varepsilon_1) \cdot \frac{\widetilde{\mathrm{P}}_5(x)}{\widetilde{\mathrm{Q}}_6(x)} \cdot (1 + 2 \cdot \varepsilon)^2, \quad |\varepsilon| \leq \varepsilon^* = 2^{-53} \\
&= e^{-x^2} \cdot (1 + \varepsilon_1) \cdot \frac{\mathrm{P}_5(x) \cdot (1 + \varepsilon_{P5})}{\mathrm{Q}_6(x) \cdot (1 + \varepsilon_{Q6})} \cdot (1 + 2 \cdot \varepsilon)^2, \qquad (22) \\
&\qquad\qquad |\varepsilon_{P5}| \leq \varepsilon(\mathrm{P}_5), \quad |\varepsilon_{Q6}| \leq \varepsilon(\mathrm{Q}_6)\,,
\end{aligned}$$

where ε_1 denotes the error imported by the exponential function (cf. page 42).

In by now accustomed manner results as an error bound for the numerator polynomial $\varepsilon(\mathrm{P}_5)$

$$\widetilde{\mathrm{P}}_5(x) = \mathrm{P}_5(x)(1 + \varepsilon_{P5}); \qquad |\varepsilon_{P5}| \leq 1.8701 \cdot 10^{-15} = \varepsilon(\mathrm{P}_5),$$

and as error bound for the denominator polynomial $\varepsilon(\mathrm{Q}_6)$

$$\widetilde{\mathrm{Q}}_6(x) = \mathrm{Q}_6(x)(1+\varepsilon_{Q6}); \qquad |\varepsilon_{Q6}| \le 2.3036 \cdot 10^{-15} = \varepsilon(\mathrm{Q}_6)\,.$$

Equation (22) yields

$$\widetilde{g}_2(x) = g_2(x) \cdot (1+\varepsilon_1) \cdot \frac{1+\varepsilon_{P5}}{1+\varepsilon_{Q6}} \cdot (1+2\varepsilon)^2 = g_2(x) \cdot (1+\varepsilon_{g2}),$$

for which can be shown

$$\widetilde{g}_2(x) = g_2(x)(1+\varepsilon_{g2}); \qquad |\varepsilon_{g2}| \le 5.7002 \cdot 10^{-15} = \varepsilon(g_2).$$

With respect to $\widetilde{\mathrm{erfc}}(x) = \mathrm{erfc}(x)(1+\varepsilon_f)$ one finds using equation (3) and the error bounds

$$\varepsilon(\mathrm{app}) = 1.5378 \cdot 10^{-16}, \qquad \varepsilon(g_2) = 5.7002 \cdot 10^{-15}$$

for $|\varepsilon_f|$ at last the estimation

$$\boxed{\widetilde{\mathrm{erfc}}(x) = \mathrm{erfc}(x)(1+\varepsilon_f); \quad |\varepsilon_f| \le 5.8540 \cdot 10^{-15} = \varepsilon(\mathrm{erfc},\mathbf{B}_2), \quad x \in \mathbf{B}_2 \cap S(2,53)\,.}$$

6.5 Error Bound for erf(x) in $\mathbf{B}_1 \cup \mathbf{B}_2 = [0.65,\, 6]$

First the subinterval [0.65, 2.2] is considered. Because of $\mathrm{erf}(x) \equiv 1 - \mathrm{erfc}(x)$, for the evaluation on the calculator holds

$$\widetilde{\mathrm{erf}}(x) := 1 \boxminus \widetilde{\mathrm{erfc}}(x) = \mathrm{erf}(x) \cdot (1+\varepsilon_0)\,.$$

Thus, one obtains

$$\begin{aligned} |\varepsilon_0| &\le 2 \cdot \varepsilon^* + [1 + 2 \cdot \varepsilon^*] \cdot \frac{\max\limits_{x \in \mathbf{B}_1}\{\mathrm{erfc}(x)\} \cdot \varepsilon(\mathrm{erfc}, \mathbf{B}_1)}{\min\limits_{x \in \mathbf{B}_1}\{\mathrm{erf}(x)\}}; \quad \varepsilon^* = 2^{-53} \\ &= 2 \cdot \varepsilon^* + [1 + 2 \cdot \varepsilon^*] \cdot \frac{[1 - \mathrm{erf}(0.65)] \cdot \varepsilon(\mathrm{erfc}, \mathbf{B}_1)}{\mathrm{erf}(0.65)}. \end{aligned}$$

For further approximation of the right hand side one needs first a lower bound of erf(0.65). According to the definition of $\varepsilon_{\mathrm{app},1}$ (cf. p. 20) follows for $x \in \mathbf{A} = [0,\, 0.65]$:

$$\mathrm{erf}(x) = \frac{H(x)}{1 - \varepsilon_{\mathrm{app},1}} \ge \frac{H(x)}{1 + \varepsilon(\mathrm{app}, 1)} = \frac{2}{\sqrt{\pi}} \cdot \frac{x \cdot e^{-x^2}}{1 + \varepsilon(\mathrm{app}, 1)} \cdot \sum_{n=0}^{14} a_n \cdot x^{2n}.$$

If one evaluates here the term after the last equality sign for $x = [0.65,\, 0.65]$ with intervals, then one obtains a confinement of the wanted lower bound of erf(0.65). The following program uses this lower bound in order to calculate a guaranteed upper bound for the expression $[1 - \mathrm{erf}(0.65)]/\mathrm{erf}(0.65)$.

```
program upper_b2;
{************************************************}
{*  Calculates the confinement of an upper bound  *}
{*  for   [1-erf(0.65)]/erf(0.65)                 *}
{************************************************}
use i_ari;
var a            : array[0..14] of interval;
    k            : integer;
    c,x,x2,eps,t : interval;
begin
  a[0] := 1;
  for k := 1 to 14 do
     a[k] := 2 * a[k-1] / (2*k + 1);
  c := 1 / SQRT( arctan(intval(1)) );
  { c: confinement of 2/SQRT(Pi) }
  x := 65 / intval(100);  x2 := x * x;
  eps := intval( (<7.2149e-19),(>7.2149e-19) );
  t := a[14];             { Horner scheme for }
  for k := 13 downto 0 do { calculating the   }
      t := t * x2 + a[k]; { sum               }
  t := c * x * exp(-x2) * t / (1 + eps);
  t := 1 / t - 1;              writeln(t);
  { t: confinement of an upper bound }
  { for [1-erf(0.65)] / erf(0.65) )  }
end.
```

One finds

$$\frac{1-\mathrm{erf}(0.65)}{\mathrm{erf}(0.65)} = \frac{\mathrm{erfc}(0.65)}{\mathrm{erf}(0.65)} < 0.5575613 =: \mathrm{OS}\,.$$

For $|\varepsilon_0|$ one obtains using the upper bound $\mathrm{OS} = 0.5575613$ the estimation

$$|\varepsilon_0| \le 2\cdot\varepsilon^* + [1+2\cdot\varepsilon^*]\cdot \mathrm{OS}\cdot\varepsilon(\mathrm{erfc},\mathbf{B}_1) < 2.7153\cdot 10^{-15} = \varepsilon(\mathrm{erf},\mathbf{B}_1)\,,$$

so

$$\boxed{\widetilde{\mathrm{erf}}(x) = \mathrm{erf}(x)(1+\varepsilon);\quad |\varepsilon| \le 2.7153\cdot 10^{-15} = \varepsilon(\mathrm{erf},\mathbf{B}_1),\ x\in\mathbf{B}_1\cap S(2,53)\,.}$$

Now the second subinterval $\mathbf{B}_2 = [2.2, 6]$ is considered. Because of $\mathrm{erf}(x) \equiv 1-\mathrm{erfc}(x)$ for the evaluation on the calculator is valid

$$\widetilde{\mathrm{erf}}(x) := 1 \boxminus \widetilde{\mathrm{erfc}}(x) = \mathrm{erf}(x)\cdot(1+\varepsilon_0)\,,$$

with which one obtains for the relative error $|\varepsilon_0|$ with respect to addition of erroneous quantities the estimation

$$|\varepsilon_0| \;\le\; 2\cdot\varepsilon^* + [1+2\cdot\varepsilon^*]\cdot\frac{\max\limits_{x\in\mathbf{B}_2}\{\mathrm{erfc}(x)\}\cdot\varepsilon(\mathrm{erfc},\mathbf{B}_2)}{\min\limits_{x\in\mathbf{B}_2}\{1-\mathrm{erfc}(x)\}};\quad \varepsilon^* = 2^{-53}$$

$$= \quad 2 \cdot \varepsilon^* + [1 + 2 \cdot \varepsilon^*] \cdot \frac{\mathrm{erfc}(2.2) \cdot \varepsilon(\mathrm{erfc}, \mathbf{B}_2)}{1 - \mathrm{erfc}(2.2)}.$$

For estimating the right hand side one needs an upper bound for the function value erfc(2.2). According to the definition of $\varepsilon_{\mathrm{app},1}$ (cf. p. 28) follows for $x \in \mathbf{B}_2 = [2.2,\, 6]$:

$$\mathrm{erfc}(x) = \frac{H(x)}{1 - \varepsilon_{\mathrm{app},1}} \leq \frac{H(x)}{1 - \varepsilon(\mathrm{app}, 1)} = \frac{2xh \cdot e^{-x^2}}{\pi \cdot [1 - \varepsilon(\mathrm{app}, 1)]} \left[\frac{1}{2x^2} + \sum_{k=1}^{70} \frac{e^{-h^2k^2}}{h^2k^2 + x^2} \right].$$

If one evaluates here the term after the last equality sign for $x = [2.2,\, 2.2]$ with intervals, then one obtains a guaranteed upper bound for erfc(2.2). With these considerations one notices that the following XSC–program gives out a guaranteed upper bound for erfc(2.2)]/[1 − erfc(2.2)].

```
program upper_b3;
{***********************************************}
{*  Calculates the confinement of a guaranteed  *}
{*                  upper bound for             *}
{*              erfc(2.2) / [1-erfc(2.2)]       *}
{***********************************************}
use i_ari;
var h,h2,pi,x,x2,eps,a,b,s,t : interval;
    k                        : integer;
begin
  h   := 93 / intval(1000);        h2 := h * h;
  pi  := 4 * arctan( intval(1) );
  x   := 11 / intval(5);
  x2 := x * x;
  eps := intval( (<9.5049e-19),(>9.5049e-19) );
  a   := 2 * x * h * exp(-x2) / (pi * (1-eps) );
  b   := 1 / (2 * x2);    s := 0,
  For k := 1 to 70 do
  s := s + exp(-h2 * k * k) / (h2 * k * k + x2);
  t   := a * (b + s);
  t   := t / (1 - t);
  { t :     confinement of a guaranteed       }
  { upper bound for  erfc(2.2)/[1-erfc(2.2)]  }
  writeln(t);
end.
```

One obtains

$$\frac{\mathrm{erfc}(2.2)}{1 - \mathrm{erfc}(2.2)} < 1.866323 \cdot 10^{-3} = \mathrm{OS}\,.$$

For $|\varepsilon_0|$ using the upper bound $\mathrm{OS} = 1.866323 \cdot 10^{-3}$ this implies the estimation

$$|\varepsilon_0| \leq 2 \cdot \varepsilon^* + [1 + 2 \cdot \varepsilon^*] \cdot \mathrm{OS} \cdot \varepsilon(\mathrm{erfc}, \mathbf{B}_2) < 2.3298 \cdot 10^{-16} = \varepsilon(\mathrm{erf}, \mathbf{B}_2)\,,$$

and thus

$$\boxed{\widetilde{\mathrm{erf}}(x) = \mathrm{erf}(x)(1+\varepsilon); \quad |\varepsilon| \le 2.3298 \cdot 10^{-16} = \varepsilon(\mathrm{erf}, \mathbf{B}_2),\ x \in \mathbf{B}_2 \cap S(2,53)\,.}$$

6.6 Approximation of erfc(x) in $\mathbf{B}_3 = [6,\ 26.5432]$

First the arguments x are bounded from above by 26.5432 in order to make sure that the function values erfc(x) lie in the **normalized** number domain of the IEEE double-format:

$$x \in \mathbf{B}_3 = [6,\ 26.5432] \quad \Longrightarrow \quad \mathrm{erfc}(x) > 2^{-1022} = 2.2250738\ldots \cdot 10^{-308}$$

For $x \ge 6$ erfc(x) is approximated by its assymptotic [2, Formeln 7.1.23, 7.1.24]

$$\mathrm{erfc}(x) = \frac{e^{-x^2}}{\sqrt{\pi}\cdot x}\left[1 - \frac{1}{(2x^2)^1} + \frac{1\cdot 3}{(2x^2)^2} - + \ldots + (-1)^N \cdot \frac{1\cdot 3\cdots(2N-1)}{(2x^2)^N} + r\right]$$

$$|r| \equiv |r(x,N)| \le \frac{1\cdot 3\cdot 5\cdots(2N+1)}{(2x^2)^{N+1}}; \quad N = 1,2,3,\ldots$$

For $N = 35$ one obtains using interval calculus

$$|r(x,35)| \le |r(6,35)| = \frac{1\cdot 3\cdot 5\cdots 71}{72^{36}} < 3.276507\cdot 10^{-16}\,.$$

The relative approximation error is then given by

$$|\varepsilon_{\mathrm{app},1}(x)| := \frac{r(6,35)}{\mathrm{erfc}(x)} < \frac{e^{-x^2}}{\sqrt{\pi}\cdot x}\cdot\frac{3.276507\cdot 10^{-16}}{\mathrm{erfc}(x)}.$$

The function erfc(x) has according to [19, S. 201] for $x > 0$ the lower bound

$$\frac{2}{\sqrt{\pi}}\cdot\frac{x\cdot e^{-x^2}}{2x^2+1} < \mathrm{erfc}(x), \qquad \text{which implies}$$

$$\begin{aligned} |\varepsilon_{\mathrm{app},1}(x)| &< \left[1+\frac{1}{2x^2}\right]\cdot 3.276507\cdot 10^{-16} \le \left[1+\frac{1}{72}\right]\cdot 3.276507\cdot 10^{-16} \\ &< 3.322015\cdot 10^{-16} = \varepsilon(\mathrm{app},1); \qquad x \ge 6, \quad N = 35. \end{aligned}$$

$$\boxed{\begin{aligned} \mathrm{erfc}(x) &\approx H(x) := \frac{e^{-x^2}}{\sqrt{\pi}\cdot x}\left[1 - \frac{1}{(2x^2)^1} + \frac{1\cdot 3}{(2x^2)^2} - + \ \ldots\ - \frac{1\cdot 3\cdots 69}{(2x^2)^{35}}\right] \\ \mathrm{erfc}(x) &= H(x)\cdot[1+\varepsilon_{\mathrm{app},1}(x)] \\ &|\varepsilon_{\mathrm{app},1}(x)| < 3.322015\cdot 10^{-16} = \varepsilon(\mathrm{app},1), \qquad x \ge 6. \end{aligned}}$$

n	$p_n :=$ nearest(.)	$q_n :=$ nearest(.)
0	$5.64189583547756078 \cdot 10^{-1}$	$1.00000000000000000 \cdot 10^{+0}$
1	$8.80253746105525775 \cdot 10^{+0}$	$1.61020914205869003 \cdot 10^{+1}$
2	$3.84683103716117320 \cdot 10^{+1}$	$7.54843505665954743 \cdot 10^{+1}$
3	$4.77209965874436377 \cdot 10^{+1}$	$1.12123870801026015 \cdot 10^{+2}$
4	$8.08040729052301677 \cdot 10^{+0}$	$3.73997570145040850 \cdot 10^{+1}$

Table 4: decimal approximations for the IEEE–coefficients p_n, q_n

Since the numerical evaluation of $H(x)$ would be too tedious for runtime reasons because of $N = 35$, erfc(x) is approximated on the calculator by

$$\operatorname{erfc}(x) \approx \frac{e^{-x^2}}{x} \cdot \frac{\mathrm{P}_4(\frac{1}{x^2})}{\mathrm{Q}_4(\frac{1}{x^2})} =: g_3(x); \quad \mathrm{P}_4(\frac{1}{x^2}) = \sum_{n=0}^{4} p_n \cdot x^{-2n}, \quad \mathrm{Q}_4(\frac{1}{x^2}) = \sum_{n=0}^{4} q_n \cdot x^{-2n}.$$

The decimal values in the table above are determined as rational approximation to $H(x)$. The coefficients p_n, q_n are the IEEE–numbers closest to the given decimal values. For the approximation the relative approximation error is given as

$$\varepsilon_{\mathrm{app},2}(x) := \frac{H(x) - g_3(x)}{H(x)}, \qquad x \in [6,\, 27]\,.$$

With $u := 1/x^2$ and

$$\mathrm{A}(u) := \frac{1}{\sqrt{\pi}} \cdot \left[1 - \frac{1 \cdot u}{2^1} + \frac{1 \cdot 3 \cdot u^2}{2^2} - + \ \ldots \ - \frac{1 \cdot 3 \cdot \ \ldots \ \cdot 69 \cdot u^{35}}{2^{35}}\right] \qquad \text{holds}$$

$$\varepsilon_{\mathrm{app},2}(x) \equiv \varepsilon_2(u) := \frac{\mathrm{A}(u) - \mathrm{P}_4(u)/\mathrm{Q}_4(u)}{\mathrm{A}(u)}, \qquad u \in [27^{-2}, 6^{-2}]\,.$$

Using the module `mpitaylor` the PASCAL-XSC function $H(x)$ can be realized by the more simple expression $\mathrm{A}(u)$:

```
VAR c : mpinterval;            { c = 1 /sqrt(Pi) }

FUNCTION H(u: mpi_taylor): mpi_taylor[0..ub(u)];
var k   : integer;
    s,a : mpi_taylor[0..ub(u)];
BEGIN
    u:= u / 2;
    s[0]:= 1;
    For k:= 1 to ub(u) do s[k]:= 0,
    a:= s;
    For k:= 1 to 35 do
```

```
    begin
      a:= -a * (2*k-1) * u;
      s:= s + a;
    end;
    H:= s * c;
END;  { H }
. . .

BEGIN
  setprec(8); { Definiert ca. 8*8 genaue Stellen }
  c := arctan(_mpinterval(1.0));
  c := 1 / ( 2 * sqrt(c) );     { c = 1/sqrt(Pi) }
END.
```

Now it is possible again to calculate automatically a guaranteed upper bound for $|\varepsilon_{\mathrm{app},2}(x)|$ bzgl. $x \in \mathbf{B}_3$.

$$H(x) \approx g_3(x) := \frac{e^{-x^2}}{x} \cdot \frac{\mathrm{P}_4(x^{-2})}{\mathrm{Q}_4(x^{-2})}; \quad |\varepsilon_{\mathrm{app},2}(x)| \leq 9.0000 \cdot 10^{-17} = \varepsilon(\mathrm{app}, 2)\,.$$

With the now available error bounds $\varepsilon(\mathrm{app}, 1)$, $\varepsilon(\mathrm{app}, 2)$ one is able to calculate with respect to the approximation

$$\mathrm{erfc}(x) \approx g_3(x) := \frac{e^{-x^2}}{x} \cdot \frac{\mathrm{P}_4(x^{-2})}{\mathrm{Q}_4(x^{-2})}$$

according to equation (2) an upper bound for the corresponding relative approximation error

$$\varepsilon_{\mathrm{app}}(x) := \frac{\mathrm{erfc}(x) - g_3(x)}{\mathrm{erfc}(x)}; \qquad |\varepsilon_{\mathrm{app}}(x)| \leq \varepsilon(\mathrm{app}), \quad x \in [6,\, 27].$$

One finds

$$\mathrm{erfc}(x) \approx g_3(x) := \frac{e^{-x^2}}{x} \cdot \frac{\mathrm{P}_4(x^{-2})}{\mathrm{Q}_4(x^{-2})}; \qquad |\varepsilon_{\mathrm{app}}(x)| \leq 4.2221 \cdot 10^{-16} = \varepsilon(\mathrm{app}).$$

In order to be able to calculate for $\mathbf{B}_3 = [6,\, 26.5432]$ an error bound for $\mathrm{erfc}(x)$ in $x \in \mathbf{B}_3 \cap S(2, 53)$ using equation (3), one needs the relative error bound for $g_3(x)$, whereby shall be assumed that the factor e^{-x^2} is calculated using the function EXPx2(...).

$$\begin{aligned} \widetilde{g}_3(x) &:= \left[\mathrm{EXP\_X2}(x) \boxdot \widetilde{\mathrm{P}}_4(x \boxdot x)\right] \boxslash \left[x \boxdot \widetilde{\mathrm{Q}}_4(x \boxdot x)\right] \\ &= \frac{e^{-x^2} \cdot (1+\varepsilon_1) \cdot \mathrm{P}_4(x^2) \cdot (1+\varepsilon_{P4}) \cdot (1+2\cdot\varepsilon)^2}{x \cdot \mathrm{Q}_4(x^2) \cdot (1+\varepsilon_{Q4}) \cdot (1+\varepsilon_k)}, \\ &\qquad |\varepsilon_1| \leq \varepsilon(1), \quad |\varepsilon| \leq \varepsilon^* = 2^{-53}, \quad |\varepsilon_{P4}| \leq \varepsilon(\mathrm{P}_4), \quad |\varepsilon_{Q4}| \leq \varepsilon(\mathrm{Q}_4). \end{aligned} \tag{23}$$

As error bound $\varepsilon(\mathrm{P}_4)$ for the evaluation of the numerator polynomial on the machine this results in

$$\widetilde{\mathrm{P}}_4(x^{-2}) = \mathrm{P}_4(x^{-2})(1+\varepsilon_{P4}); \qquad |\varepsilon_{P4}| \le 6.2806 \cdot 10^{-16} = \varepsilon(\mathrm{P}_4)$$

and the error bound is $\varepsilon(\mathrm{Q}_4)$ for the evaluation of the denominator polynomial on the machine one finds

$$\boxed{\widetilde{\mathrm{Q}}_4(x^{-2}) = \mathrm{Q}_4(x^{-2})(1+\varepsilon_{Q4}); \qquad |\varepsilon_{Q4}| \le 6.4250 \cdot 10^{-16} = \varepsilon(\mathrm{Q}_4)\,.}$$

Equation (23) yields

$$\widetilde{g}_3(x) = g_3(x) \cdot \frac{(1+\varepsilon_1)(1+\varepsilon_{P4})(1+2\cdot\varepsilon)^2}{(1+\varepsilon_{Q4})(1+2\cdot\varepsilon)} = g_3(x)\cdot(1+\varepsilon_{g3}),$$

for this can be shown that

$$\widetilde{g}_3(x) = g_3(x)(1+\varepsilon_{g3}); \qquad |\varepsilon_{g3}| \le 3.0190 \cdot 10^{-15} = \varepsilon(g_3).$$

With respect to $\widetilde{\mathrm{erfc}}(x) = \mathrm{erfc}(x)(1+\varepsilon_f)$ one finds using equation (3) and the error bounds $\varepsilon(\mathrm{app}) = 4.2221 \cdot 10^{-16}$, $\varepsilon(g_3) = 3.0910 \cdot 10^{-15}$ for $|\varepsilon_f|$ the estimation

$$\boxed{\widetilde{\mathrm{erfc}}(x) = \mathrm{erfc}(x)(1+\varepsilon_f); \quad |\varepsilon_f| \le 3.4413 \cdot 10^{-15} = \varepsilon(\mathrm{erfc},\mathbf{B}_3), \;\; x \in \mathbf{B}_3 \cap S(2,53)\,.}$$

6.7 Error Bound for erf(x) in $\mathbf{B}_3 \cup \mathbf{B}_4$, i. e. for $x \ge 6$

Here $\mathrm{erf}(x)$ is approximated by 1, i. e.

$$\mathrm{erf}(x) \approx 1\,.$$

The relative approximation error $\varepsilon_{\mathrm{app}}(x)$ is then monotonically decreasing and we have

$$\varepsilon_{\mathrm{app}}(x) := \frac{1-\mathrm{erf}(x)}{\mathrm{erf}(x)} = \frac{1}{\mathrm{erf}(x)} - 1 = \frac{\mathrm{erfc}(x)}{1-\mathrm{erfc}(x)} \le \frac{\mathrm{erfc}(6)}{1-\mathrm{erfc}(6)},$$

where a guaranteed upper bound for $\mathrm{erfc}(6)/[1-\mathrm{erfc}(6)]$ can be calculated with the program `upper_b3` on page 34 if one replaces only the command `x:=11/intval(5)` by `x:=6`. The result is

$$\frac{\mathrm{erfc}(6)}{1-\mathrm{erfc}(6)} < 2.1520 \cdot 10^{-17} \quad \Longrightarrow \quad \varepsilon_{\mathrm{app}}(x) < 2.1520 \cdot 10^{-17} = \varepsilon(\mathrm{app}) \quad \Longrightarrow$$

$$\boxed{\widetilde{\mathrm{erf}}(x) = \mathrm{erf}(x)(1+\varepsilon_f); \quad |\varepsilon_f| \le 2.1520 \cdot 10^{-17} = \varepsilon(\mathrm{erf}, x \ge 6)\,.}$$

6.8 Approximation of erfc(x) in $\mathbf{C} = (-\infty, 0]$

Due to the identity $\quad \mathrm{erfc}(x) \equiv 1 - \mathrm{erf}(x) \quad$ we have in $\mathbf{C} = (-\infty, 0]$

$$\begin{aligned} \widetilde{\mathrm{erfc}}(x) &:= 1 \boxminus \widetilde{\mathrm{erf}}(x) = \mathrm{erfc}(x) \cdot (1 + \varepsilon_f), \qquad \text{with} \\ |\varepsilon_f| &\leq 2 \cdot \varepsilon^* + (1 + 2 \cdot \varepsilon^*) \cdot \max_{-x \geq 0} \left[\frac{1 \cdot 0 + \mathrm{erf}(-x) \cdot \varepsilon(\mathrm{erf})}{1 + \mathrm{erf}(-x)} \right]. \end{aligned}$$

Since $[\ldots]$ increases for $(-x) \to +\infty$ monotonically, immediately follows

$$|\varepsilon_f| \leq 2 \cdot \varepsilon^* + (1 + 2 \cdot \varepsilon^*) \cdot 0.5 \cdot 2.7153 \cdot 10^{-15} < 1.5797 \cdot 10^{-15} \quad \Longrightarrow$$

$$\boxed{\widetilde{\mathrm{erfc}}(x) = \mathrm{erfc}(x)(1+\varepsilon_f); \quad |\varepsilon_f| \leq 1.5797 \cdot 10^{-15} = \varepsilon(\mathrm{erfc}, \mathbf{C}),\ x \in \mathbf{C} \cap S(2,53)\,.}$$

6.9 Summary of the results for erf(x) and erfc(x)

By this is performed for the functions $\mathrm{erf}(x), \mathrm{erfc}(x)$ the error approximation with respect to the IEEE-double-format, assuming only a **highly accurate** basic arithmetics (1 ulp) in all subintervals. The relative error bounds are the maxima of the upper bounds calculated in each part:

$$\boxed{\varepsilon(\mathrm{erf}) = 2.7153 \cdot 10^{-15}; \qquad |x| \in [1.97193 \cdot 10^{-308},\ +\infty) \cap S(2,53)}$$

$$\boxed{\varepsilon(\mathrm{erfc}) = 5.8540 \cdot 10^{-15}; \qquad x \in (-\infty, 26.5432] \cap S(2,53)}$$

If one evaluates the functions $\mathrm{erf}(x), \mathrm{erfc}(x)$ using the same algorithm in IEEE double-format in **maximally accurate** arithmetics (i. e. rounding to the nearest floating point number), one finds correspondingly the even smaller error bounds

$$\boxed{\varepsilon(\mathrm{erf}) = 1.5643 \cdot 10^{-15}; \qquad |x| \in [1.97193 \cdot 10^{-308},\ +\infty) \cap S(2,53)}$$

$$\boxed{\varepsilon(\mathrm{erfc}) = 3.2952 \cdot 10^{-15}; \qquad x \in (-\infty, 26.5432] \cap S(2,53)\,.}$$

The given numerical error bounds correspond to arguments in IEEE-double-format. The presented method of error estimation can also be performed for other data formats (cf. e. g. [5]).

A The Auxiliary Function e^{-x^2}

As the function $f = e^{-x^2}$ plays an important role in implementing the error function and in Dawsons integral, in this section is given for f a suitable algorithm with corresponding error estimation assuming the following:

- The basic operations in IEEE double-format are performed **highly accurate**, i. e. $a \boxdot b = (a \cdot b)(1+\varepsilon), \quad |\varepsilon| \le 2 \cdot \varepsilon^* = 2^{-52} = 2.220446\ldots \cdot 10^{-16}$
- It is used the fast exponential function in IEEE double-format, implemented according to the table–procedure, (cf. [10, 11]), i. e.
  $\mathrm{EXP}(x) = e^x \cdot (1+\varepsilon_{\exp}), \quad |\varepsilon_{\exp}| \le 2.3580 \cdot 10^{-16} =: \varepsilon(\exp)$

In order to prevent the function values $f(x)$ from falling into the denormalized number domain of the IEEE double-format the inequality

$$\begin{aligned} e^{-x^2} &\ge 2^{-1022} \quad \text{(smallest positive normalized floating point number)} \quad \Longleftrightarrow \\ |x| &\le \sqrt{1022 \cdot \ln(2)} = 26.615717509251260202\ldots \end{aligned}$$

must be fulfilled. Because $f(-x) \equiv f$ one may restrict to nonnegative arguments x, i. e. it is considered only

$$0 \le x < 26.615717.$$

First the question has to be answered why $f = e^{-x^2}$ cannot be realized on the calculator by calling up the already implemented standard exponential function $\mathrm{EXP}(\ldots)$ with perturbed argument $-x \boxdot x$. The corresponding error estimation will show that this method is not appropriate.

If $\mathrm{EXP}(x)$ denotes the exponential function in IEEE-format, implemented according to the table–procedure, then we first have

$$\mathrm{EXP}(x) = e^x \cdot (1+\varepsilon_{\exp}), \qquad |\varepsilon_{\exp}| \le 2.3580 \cdot 10^{-16} =: \varepsilon(\exp)$$

$$\mathrm{EXP}(-x \boxdot x) = e^{-x^2} \cdot (1+\varepsilon_1), \qquad |\varepsilon_1| \le \varepsilon(1) = ?$$

Because of

$$x \boxdot x = x^2 \cdot (1+\varepsilon_x), \quad |\varepsilon_x| \le 2 \cdot \varepsilon^* \le 2.220447 \cdot 10^{-16} = \varepsilon(x),$$

$$x \in [0, 26.615717] \quad \Longrightarrow \quad x^2 \in [0, 708.397]$$

the function $\mathrm{EXP}(\ldots)$ is called with perturbed arguments. Using the XSC–program `ErrBound` one obtains with input

Interval of exact arguments $[x1, x2] = ?$	$[0, 708.397]$
Relative bound for perturbed arguments	$2.220447 \cdot 10^{-16}$
Rel. error bound for function in screen S(B,k) = ?	$2.3580 \cdot 10^{-16}$

the unnecessarily large bound

$$\varepsilon(1) = 1.5754 \cdot 10^{-13}\,.$$

An error bound smaller by about the order two is given by the following algorithm whose runtime compared with the fast exponential function is about double:

```
var x,m : real;
    z   : integer;
begin
  z := trunc(x);  { z: integer part of argument }
  m := x - z;     { m:        }
  if m > 0.5 then
  begin
    z := z + 1;   { z = 0,1,2,....,27                    }
    m := m - 1;   { m <= 0.5;                 x = z + m; }
  end; ...
end.
```

With the integers m and z calculated in this way, e^{-x^2} is then calculated according to

$$e^{-x^2} = e^{-z^2} \cdot e^{-(2z)\cdot m} \cdot e^{-m^2}. \tag{24}$$

The constants e^{-z^2} are calculated using the module `mp_ari` and can be stored for $z \in \{0, 1, \ldots, 26\}$ maximally accurate in IEEE-double-format:

$$\widetilde{e^{-z^2}} = e^{-z^2} \cdot (1 + \varepsilon); \qquad |\varepsilon| \le \varepsilon^* = 2^{-53} = 1.1102230\ldots \cdot 10^{-16}.$$

Because $e^{-27^2} = e^{-729} < 2^{-1022}$, the constant e^{-27^2} cannot be stored maximally accurate in the normalized number domain of the IEEE-double-format. If, however, one factorizes e^{-729} into the two factors:

$$e^{-729} = \left[2^{64} \cdot e^{-729}\right] \cdot 2^{-64},$$

then the first factor can be stored maximally accurate because of $[2^{64} \cdot e^{-729}] > 2^{-1022}$, and the multiplication with 2^{-64} can be performed very fast and without rounding error. The relative error bound $\varepsilon(k)$ is thus valid for all factors e^{-z^2}.
As the exponent of the second factor in equation (24) is calculated without rounding error, we have

$$\mathrm{EXP}(-(z \boxplus z) \boxdot m) = \mathrm{EXP}(-(z + z) \cdot m) = e^{-(2z)\cdot m} \cdot (1 + \varepsilon_{\exp})$$

$$|\varepsilon_{\exp}| \le 2.3580 \cdot 10^{-16} = \varepsilon(\exp)\,.$$

The exponential function is therefore called up in the algorithm presented here with the **unperturbed** argument $-(2z) \cdot m$. The last factor e^{-m^2} in (24) is evaluated using

the fast exponential function which is called up with the perturbed argument $-m \boxdot m$, which is, however, bounded with very much smaller absolute values.

$$\mathrm{EXP}(-m \boxdot m) = e^{-m^2}(1+\varepsilon_2), \quad |\varepsilon_2| \le \varepsilon(2) = ?$$

Using the program `ErrBound` with the now significantly smaller argument interval this results in

$$\text{Interval of exact arguments } [x1, x2] = ? \;\Big|\; [-0.25,\, 0]$$

(the other input data agree with those given in the calculation of $\varepsilon(1)$) the wanted upper bound for the absolute value of the relative calculation error ε_2

$$\varepsilon(2) = 2.9132 \cdot 10^{-16}\,.$$

Thus, the error bounds for the three factors in (24) are estimated, and one obtains with

$$\widetilde{e^{-x^2}} = e^{-z^2}(1+\varepsilon) \cdot e^{-(2z)\cdot m}(1+\varepsilon_{exp}) \cdot e^{-m^2}(1+\varepsilon_2) \cdot (1+2\varepsilon)^2 = e^{-x^2}(1+\varepsilon_3)$$

for $|\varepsilon_3|$, using again the program `ErrBound` assuming a **highly accurate** arithmetics, as upper bound for the total error

$$\boxed{\widetilde{e^{-x^2}} = e^{-x^2}(1+\varepsilon_3);\;\; |\varepsilon_3| \le 1.0823 \cdot 10^{-15},\;\; |x| \in [0,\, 26.615717] \cap S(2,53)\,.}$$

Comparison with the error bound $\varepsilon(1) = 1.5754 \cdot 10^{-13}$ shows that instead of about 13 digits, the function values e^{-x^2} can be calculated with at least 15 correct decimals.

Assuming **maximally accurate arithmetics** one obtains by using otherwise the same algorithm, the again improved total error bound

$$\boxed{\widetilde{e^{-x^2}} = e^{-x^2}(1+\varepsilon_3);\;\; |\varepsilon_3| \le 8.3243 \cdot 10^{-16},\;\; |x| \in [0,\, 26.615717] \cap S(2,53)\,.}$$

B A Simple Test Program, Numerical Results

The following program uses the relation

$$\mathrm{erf}(x) + \mathrm{erfc}(x) - 1 = 0$$

for a simple test. The evaluation with intervals of the left hand side must result in an interval containing zero.

The user has only to include the module `erf_mod` with the command `use erf_mod` into his PASCAL-XSC program in order to have available the erf as well as the erfc function for interval arguments.

```
PROGRAM erf_test;
USE i_ari, erf_mod;
VAR x, fx: interval;
BEGIN
  writeln;
  writeln('           ****************************************');
  writeln('           *    Computing  erf(x)  and  erfc(x)     *');
  writeln('           ****************************************');
  writeln;
  writeln('Program termination using <Ctrl> <C> ');
  REPEAT
    writeln;
    write('x = [xlb, xub] = ?  '); read(x); writeln;
    writeln('Argument interval:', x); writeln;
    fx := erf(x);
    writeln('erf(x)  = [', fx.inf:23:0:-1, ' , ', fx.sup:23:0:+1, ' ]');
    fx := erfc(x);
    writeln('erfc(x) = [', fx.inf:23:0:-1, ' , ', fx.sup:23:0:+1, ' ]');
    writeln('erf(x) + erfc(x) - 1: ', erf(x) + erfc(x) - 1 );
  UNTIL FALSE

END.
```

Output of test program:

```
                      ****************************************
                      *    Computing  erf(x)  and  erfc(x)     *
                      ****************************************
Program termination using <Ctrl> <C>

x = [xlb, xub] = ?
Argument interval:[  1.000000000000000E+000,  1.000000000000000E+000 ]

erf(x)  = [ 8.427007929497132E-001 ,  8.427007929497166E-001 ]
erfc(x) = [ 1.572992070502843E-001 ,  1.572992070502860E-001 ]
erf(x) + erfc(x) - 1: [               -2.6E-015,               2.7E-015 ]


x = [xlb, xub] = ?
Argument interval:[  5.000000000000000E+000,  5.000000000000000E+000 ]

erf(x)  = [ 9.999999999984608E-001 ,  9.999999999984643E-001 ]
erfc(x) = [ 1.537459794428029E-012 ,  1.537459794428042E-012 ]
erf(x) + erfc(x) - 1: [               -1.7E-015,               1.8E-015 ]

x = [xlb, xub] = ?
Argument interval:[ -1.000000000000000E+000, -1.000000000000000E+000 ]
```

```
erf(x)  = [-8.427007929497166E-001 , -8.427007929497132E-001 ]
erfc(x) = [ 1.842700792949708E+000 ,  1.842700792949722E+000 ]
erf(x) + erfc(x) - 1: [               -8.3E-015,               8.5E-015 ]

x = [xlb, xub] = ?
Argument interval:[ -2.000000000000000E+000, -2.000000000000000E+000 ]

erf(x)  = [-9.953222650189544E-001 , -9.953222650189510E-001 ]
erfc(x) = [ 1.995322265018946E+000 ,  1.995322265018960E+000 ]
erf(x) + erfc(x) - 1: [               -8.3E-015,               8.5E-015 ]


x = [xlb, xub] = ?
Argument interval:[  2.000000000000000E+002,  2.000000000000000E+002 ]

erf(x)  = [ 9.999999999999967E-001 ,  1.000000000000000E+000 ]
erfc(x) = [ 0.000000000000000E+000 ,  4.450147717014403E-308 ]
erf(x) + erfc(x) - 1: [               -3.3E-015,               2.3E-016 ]
```

The results essentially reflect the size of the error bounds for implementations of erf and erfc. Also in all tested cases, zero lies inside the calculated resulting interval.

References

[1] G. Alefeld, J. Herzberger: *Einführung in die Intervallrechnung.* Reihe Informatik/12; BI, 1974.

[2] M. Abramowitz and I.A. Stegun: – *Handbook of Mathematical Functions, Graphs, and Mathematical Tables*. Dover Publications, Inc., New York.

[3] F. Blomquist: *Automatische a priori Fehlerabschätzungen.* Inst. für Angewandte Mathematik, Universität Karlsruhe, 1995.

[4] F. Blomquist: *Mathematische Funktionen für Intervallargumente.* Interner Abschlussbericht eines gleichnamigen Projektes, Inst. für Angewandte Mathematik, Universität Karlsruhe, 1997.

[5] F. Blomquist: *Dezimalversion von PASCAL-XSC.* Inst. für Angewandte Mathematik, Universität Karlsruhe, erscheint 1998.

[6] K. D. Braune: *Hochgenaue Standardfunktionen für reelle und komplexe Punkte und Intervalle in beliebigen Gleitpunktrastern.* Dissertation, Universität Karlsruhe 1987.

[7] B.D. Fried and S.D. Conte: *The plasma dispersion function.* Academic Press, New York, N.Y. and London, England, 1961.

[8] W. Gautschi: *Error function and Fresnel integrals.* Chapter 7 in [2].

[9] R. Hammer, M. Hocks, U. Kulisch, D. Ratz: *Numerical Toolbox for Verified Computing I.* Springer Series in Computational Mathematics 21, 1993.

[10] W. Hofschuster, W. und Krämer: *Ein rechnergestützter Fehlerkalkül mit Anwendung auf ein genaues Tabellenverfahren.* Preprint 96/5 des IWRMM, Universität Karlsruhe, 35 Seiten, 1996.

[11] W. Hofschuster, W. Krämer: *A Computer Oriented Approach to Get Sharp Reliable Error Bounds*, *Reliable Computing 3*, pp. 239-248, 1997.

[12] W. Krämer: *Inverse Standardfunktionen für reelle und komplexe Intervallargumente mit a priori Fehlerabschätzungen für beliebige Datenformate.* Dissertation, Universität Karlsruhe, 1987.

[13] W. Krämer: *Multiple-Precision Computations with Result Verification*, in: *Scientific Computing with Automatic Result Verification*, Adams, E., Kulisch, U.(editors), Academic Press, pp. 311–343, 1992.

[14] W. Krämer: *Sichere und genaue Abschätzung des Approximationsfehlers bei rationalen Approximationen.* Forschungsschwerpunkt Computerarithmetik, Intervallrechnung und Numerische Algorithmen mit Ergebnisverifikation, Bericht 3/1996, Karlsruhe, 1996.

[15] W. Krämer: *Constructive Error Analysis.* Journal of Universal Computer Science (JUCS), Vol. 4, No. 2, pp. 147-163, 1998.

[16] W. Krämer:" *A priori Worst Case Error Bounds for Floating-Point Computations.* IEEE Transactions on Computers, Vol. 47, No. 7, July 1998.

[17] B. Lohmander, S. Rittsten: *Table of the Function* $y = e^{-x^2} \int_0^x e^{t^2} dt$, Kungl. Fysiogr. Sällsk. i. Lund Förh., V 28, 1958, pp. 45–52.

[18] Y.L. Luke: *The Special Functions and their Approximations.* Volume I, Academic Press, NEW YORK and London, 1969.

[19] Y.L. Luke: *The Special Functions and their Approximations.* Volume II; Academic Press, NEW YORK and London, 1969.

[20] Y.L. Luke. *Algorithms for the Computation of mathematical Functions* Academic Press, NEW YORK SAN FRANCISCO LONDON, 1977.

[21] Y.L. Luke: *Mathematical Functions and their Approximations* Academic Press, NEW YORK SAN FRANCISCO LONDON, 1975.

[22] *Algebrasystem Mathematica.* Wolfram Research, Inc., Champaign, Illinois.

[23] W.L. Miller and A.R. Gordon: *Numerical evaluation of infinit series.* Jn. Phys. Chem., V. 35, 1931, especially part V, p. 2856-2857, 2860-2865.

[24] F. Oberhettinger: *Tabellen zur Fourier Transformation.* Springer, Berlin, 1957.

[25] PASCAL–XSC: A PASCAL Extension for Scientific Computation and Numerical Data Processing. Numeric Software GmbH, D–76492 Baden–Baden, Germany.

[26] J.B. Rosser: *Theorie and Application of* $\int_0^z e^{-x^2}dx$ *and* $\int_0^z e^{-p^2y^2}dy \int_0^y e^{-x^2}dx$. *Part I.* Methods of Computation, NEW YORK, 1948.

[27] H.E. Salzer: *Formulas for calculating the error function of a complex variable.* Math. Tables and Other Aids to Computation 5, 67-70, (1951).

[28] E.C. Titchmarsh: *Introduction to the Theory of Fourier Integrals.* Oxford, 1937, p. 60-64.

[29] *IEEE Standard for Binary Floating-Point Arithmetic, ANSI–IEEE Standard 754-1985, 1985.*