Extending the Range of the Cody and Waite Range Reduction Method

Peter Kornerup
University of Southern Denmark
Odense, Denmark
E-mail: kornerup@imada.sdu.dk

Jean-Michel Muller
CNRS-LIP-Arénaire
Lyon, France
E-mail: Jean-Michel.Muller@ens-lyon.fr

September 8, 2005

Abstract

This paper shows how the method of Cody and Waite for range-reduction of arguments to transcendental functions may be extended to much larger values of arguments, when a fused multiply-add instruction is available. It is first shown how the multiple N can be determined, such that the reduced argument $x - NC$ can be calculated exactly, satisfying $|x - NC| < 1$, given a finite precision constant C. It is then shown how C must be chosen such that $x - NC$ approximates $x - NK$, with an absolute error less than one ulp of the arithmetic, where K is a given transcendental constant. The method is applicable in a standard precision of the CPU, as well as when a double precision fused multiply-add instruction is emulated by the single precision instructions, thus further extending the range of applicability.

1 Introduction

When determining the value of some transcendental function $f(x)$, it is often most convenient only to approximate the values of $f(\cdot)$ of sufficient accuracy over some quite narrow interval, usually around zero. By exploiting properties of the particular function, other arguments must be range-reduced to fall in that interval. This transformation will often take the form of a translation of the argument x by subtracting an integer multiple of a (often transcendental) constant K, such that $f(x - NK)$ can be approximated, from which the wanted value of $f(x)$ then can be derived.

This paper first extends the "classical" method of range-reduction by Cody and Waite [CW80, Cod82], by exploiting the now more widely available fused multiply-add instruction (FMA), where the result of a computation $R = X \cdot Y + Z$ is only subject to a single final rounding. Let C be an approximation to the constant K, where $C = C_1 + C_2$ and C_1 is exactly representable in the available arithmetic, C_2 providing additional accuracy.

For N and C_1 appropriately chosen, for $\frac{1}{2} < C < 2$ and using the FMA instruction, we shall first show that not only is is possible to calculate $x - NC_1$ exactly; but also such that its value is bounded, $|x - NC_1| < 1$. This may or may not be a perfect reduction, $-\frac{C}{2} \leq x - NC_1 < \frac{C}{2}$, but if not it is easily correctable. We shall then also determine conditions on $C = C_1 + C_2$, as an approximation to K, for the final result to have a relative error less than one unit (ulp) in the last position of $x - N(C_1 + C_2)$.

Our extension to the Cody and Waite method consists in extending the range of x for which the method is applicable. Their method is limited by the requirement that the product NC_1 being exactly representable, the use of an FMA instruction now only requires that x, N, C_1 and $x - NC_1$ are individually exactly representable. E.g., exploiting the 64-bit mantissas of the IEEE-754 double-extended precision internal format, range-reduction now becomes possible for $|x| < 2^{64}$, x given in ordinary double precision. By obvious extension, emulating a double-precision FMA-instruction using the available FMA-instruction, the values of x for which the range reduction is applicable can thus be significantly extended, e.g., now applicable for $|x| < 2^{128}$, x given in ordinary IEEE-754 double precision.

We have previously published [BDK+05] an algorithm applicable for $|x| < 2^{64}$, which is fairly "expensive" in table support. This method only requires a few constants, and covers an even bigger set of input values when emulating a double-precision FMA-instruction. Another paper, [LBD03] also considered the use of FMA-instructions; it obtained more complicated conditions for the selection of constants, but triggered this research.

For the very large, but extremely rare arguments, the best approach still seems to be the method of Payne and Hanek [PH83]

Section 2 provides more detail on the Cody and Waite method, the use of a fused multiply-add (FMA) instruction, and the use of the latter to extend the range of arguments to be reduced. For a given value of C, either in the interval $\frac{1}{2} < C < 1$ or in $1 < C < 2$, Section 3 then deals with the range-reduction of an argument x, where it is shown how to choose the constant N such that the exact value of $x - NC$ can be found, satisfying $|x - NC| < 1$. Section 4 shows how to choose the approximation C of the constant K, and finally Section 5 draws conclusions.

2 Background

The naive range-reduction method consists in directly computing

$$y = x - NK$$

with operands in the target floating-point format. When x is close to a multiple of K, this may result in very inaccurate results. Consider the following example [Mul97]. Assume that we use a radix 10 floating-point number system with 4-digit mantissas, and suppose $x = 88.34$ and $K = \pi/2$. We simply evaluate $y = x - 56C$ in the arithmetic of our number system, $K = \pi/2$ being represented by its closest 4-digit approximation, namely, $C = 1.571$. Assuming that the arithmetic operations return correctly rounded-to-the-nearest results, we get $y = 0.3640$. This gives one significant digit only, since the exact result is $0.375405699485789\cdots$.

For very small arguments, this naive method may return better results if a larger internal format is used and/or if a fused multiply-add instruction (FMA) is available.

Cody and Waite [CW80, Cod82] suggested the following improvement to the naive method. It consists of finding two values C_1 and C_2 that are exactly representable in the floating-point system being used, and such that:

- C_1 is very close to K, and is representable using a few digits only (i.e., C_1 is a machine number containing the first few digits of K). A consequence is that for values of N that are not too large, NC_1 is exactly representable in the floating-point system.
- $K \approx C = C_1 + C_2$ to beyond working precision.

With correctly rounded arithmetic, if NC_1 is exactly representable, it is exactly computed. Moreover, the following result due to Sterbenz shows that $x - NC_1$ is exactly computed.

Lemma 1 (Sterbenz Lemma) *In a floating-point system with correct rounding and subnormal numbers, if a and b are floating-point numbers such that*

$$a/2 \leq b \leq 2a$$

then $a - b$ is computed exactly.

Therefore, instead of evaluating $x - NK$, we evaluate

$$(x - NC_1) - NC_2, \tag{1}$$

hence, since the computation of $z = (x - NC_1)$ is exact (provided that N is not too large), the only errors occurring are:

- the representation of the (assumed infinitely precise) constant K by $C = C_1 + C_2$;
- the multiplication NC_2;
- the subtraction $z - NC_2$.

Again, the possible use of a larger internal format and/or an FMA instruction may make this method much more accurate.

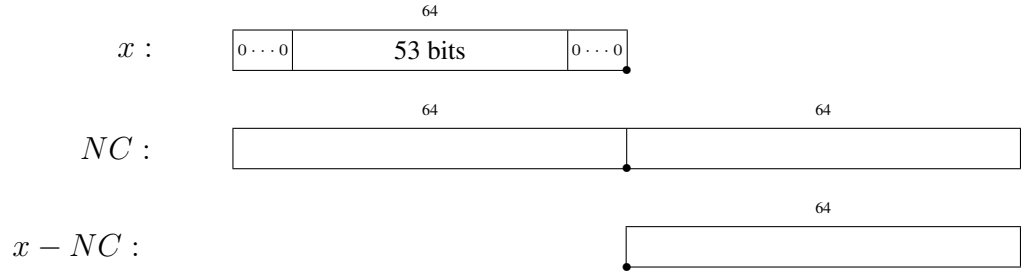
More precisely, if conventional arithmetic operations only (i.e., $\pm$, $\times$) are used, Cody and Waite's method only work if NC_1 is exactly representable (in the target format, or in the internal wider format, if any). Assuming an n-bit internal binary format and a p-bit (with $p < n$) value of C_1, this limits $|N|$ to values less than 2^{n-p}. Hence, a compromise must be found: If p is small, the method will work in a large range, but will be inaccurate (since what makes Cody and Waite's method interesting is that we have a $p + n$-bit approximation to C, instead of an n-bit one with the naive method). If p is large, the method will be accurate, but it will work in a small range only.

When a FMA is available, what only matters is that $x - NC_1$ should be exactly representable. This allows larger values of N. This problem was investigated by Boldo, Daumas and Li [LBD03].

2.1 Using a fused multiply-add instruction

Assume the argument x is given as a double precision (53 bit) FP number, to be reduced by subtracting the integer multiple N of the constant C from x. The arithmetic may then be performed in double-extended format (64 bit), with a fused multiply-add operation (FMA). We assume for the moment that the unit position is at the right end of the register containing x, and is pictured as if in a fixed-point format, assuming $C < 2$.

The operation can be pictured as:



The result $x - NC$ is now assumed by cancellation to be reduced to fall in the interval $|x - NC| < 1$. Under suitable conditions this is possible due to the following strengthening of Sterbenz Lemma:

Lemma 2 (Subtraction Theorem, from [LBD03])
Let a and b be two FP-numbers with $p + \ell$ bits in the mantissa, where the integer $\ell \geq 0$. If

$$b/2 \leq (1 + 2^{-\ell})^{-1} b \leq a \leq (1 + 2^{-\ell})b \leq 2b,$$

then $a - b$ is an FP number, i.e., $a - b$ can be represented exactly by a FP-number with p bits in the significand.

Provided that

$$(1 + 2^{-\ell})^{-1} x \leq NC \leq (1 + 2^{-\ell})x$$

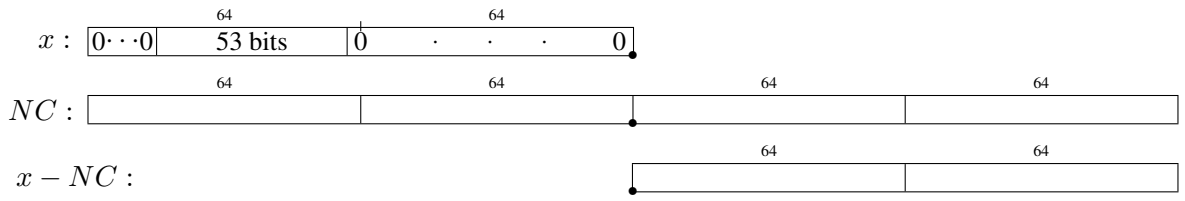
then applying Lemma 2 with $\ell = p = 64$, employing double-extended (64 bit) FP arithmetic with a FMA operation, it follows that $x - NC$ is exactly representable. Note that the value of N is to be obtained from the value of C, where C is given to some sufficient accuracy. In practice this is achieved by multiplying x by some approximation of C^{-1}. Below we shall derive conditions which will assure that $|x - NC| < 1$.

The other matter of concern is the accuracy of C, supposed to be an approximation to K, such that $(x - NC) - (x - NK) = N(C - K)$ is sufficiently small for the evaluation of the function value $f(x)$, based on the reduced argument $x - NC$.

Next we want to extend the range over which the argument can be reduced to the "basic" interval, so N will now become larger. This, of course, requires more precision in the constants C and C^{-1}. The idea is then to emulate a multiple precision arithmetic, now applied to much larger values of x. Similarly, N now also has to be in "double-double-extended" 128-bit representation, and to obtain sufficient precision in C it turns out that it must be in "quadruple" precision.

But to illustrate the idea let $C = C_1 + C_2$ and $N = N_1 + N_2$ where C_1 and N_1 is chosen such that $x - N_1 C_1$ is exact in double-extended precision representation. For the moment we are just assuming that C is sufficiently accurate in "double-double-extended"128-bit representation.

This process may now be shown as:



as it may be realized by emulating 128-bit FMA instructions by means of 64-bit arithmetic. This allows argument reduction for values of x for which $|x| \leq (2^{53} - 1)2^{75} \approx 2^{128}$.

It turns out that there are two different, but very similar, methods for this type of range-reduction, together they cover reduction constants K for $\frac{1}{2} < K < 2$, in practice also covering the uninteresting case of $K = 1$.

In the following we shall use the terminology "double" or "quadruple" precision, where these terms refer to the number of basic "data-units" available in the system. For a radix β, k-digit system these are characterized by being able to represent scaled integers $x\beta^p$ with x in the interval $[0; \beta^k)$ with an attached sign. E.g., when we here employ the FMA instruction internally using the IEEE-754 "extended double precision" (64-bit) significands, our basic data

unit is characterized by the constant $R = 2^{64}$. In the examples we will use 5-digit decimal numbers as our data unit, so here $R = 10^5$.

Our first concern will be to obtain sufficient cancellation using the FMA instruction, only later we shall return to the problem of approximating K by C sufficiently well to allow $x - NC$ to be used for $x - NK$.

3 Obtaining cancellation

Note that the argument reduction is supposed to provide a result $x - NC$, belonging to the interval $-\frac{C}{2} \leq x - NC \leq \frac{C}{2}$, but on the other hand we require that the result is exact as a fixed-point number in the interval $(-1; 1)$, hence we must require $C < 2$.

For K (and hence C) in the interval $[1; 2)$, to gain some additional accuracy, we can write $C = 1 + C'$, and thus $x - NC$ can be calculated by an FMA instruction as $(x - N) - NC'$ for $1 \leq C < 2$. Let c be an approximation to C^{-1}, say $c = \lfloor RC^{-1}/R \rceil$[1], then $\frac{1}{2} < c \leq 1$ and $|N| < |x|$ and N can be represented in the basic word size. It turns out that depending on the situation, N is here must be determined either as the "round-up" or the the "round-down" value of xc.

When K (and hence C) is in the interval $(\frac{1}{2}; 1)$, then $c \in (1; 2)$ can be written as $c = 1 + c'$ with $0 < c' < 1$. Thus N can be formed as $N = \lfloor cx \rceil = \lfloor x + c'x \rceil$, using the FMA instruction, this way allowing one additional digit of accuracy in the specification of c.

We shall now show how it is possible to perform the range-reduction $x - NC$ as an exact process, i.e., delivering the exact value of $x - NC < 1$, given some value of C. First some illustrative examples for $\frac{1}{2} < C < 1$.

Example 1 Assume that we are working in a 10 decimal digit arithmetic, and want to reduce the argument $x = 56789{,}00000$ by multiples of $K = \frac{\pi}{4}$ using the approximation $\frac{\pi}{4} \approx C = 0.78539{,}81634$, with $c = 1 + 0.27323{,}95448$ as the approximation to $\frac{4}{\pi}$. Then $N = \lfloor xc \rceil = 72306{,}00051$ is the rounded value to be used for the reduction. Note that $1 - Cc = 2.761 \cdot 10^{-11}$.

From these values, using a 10 decimal FMA instruction (forming products in 20 decimal digits) it is found that $x - NC = -0.33534{,}63334$. The relative error $\frac{x-NC}{x} = -5.905 \cdots 10^{-11}$ satisfies the condition in the lemma above, such that the reduced value of x is the exact value of $x - NC$ for the provided value of C.

$$\begin{array}{ll} x = & 56789{,}00000. \\ NC = & 56789{,}00000.33534{,}63334 \\ \hline x - NC = & -.33534{,}63334 \end{array}$$

□

Example 2 Now, if only a 5-decimal arithmetic is available, 10-decimal arithmetic can be programmed using the available arithmetic. For the same problem as in Example 1, the components of $C = C_1 + C_2$ and $c = c_1 + c_2$ can be delivered as four constants, and x is assumed to have at most 5 digits, being of the form $x = x'10^5$ with $x' < 10^5$. Then $N = N_1 + N_2$ can be found from x and $c_1 + c_2$, employing the 5-decimal FMA instruction. Note that N_1 must be chosen as the most significant, (rounded) part of N, and N_2 appropriately as $N - N_1$, to ensure that $x - N_1C_1$ has zeroes in its 5 most significant digits.

[1]Note: $\lfloor x \rceil$ denotes the round-to-nearest integer value of x.

Continuing the previous example for $x = 56789{,}00000$ we get:

$$\begin{aligned} C = C_1 + C_2 &= 0.78540 - 0.00000{,}18366 \\ c = c_1 + c_2 &= 1.27324 - 0.00000{,}04552 \\ N = N_1 + N_2 &= 72306{,}00000 + 00051 \end{aligned}$$

where N_1 is found using FMA instructions as

$$\begin{aligned} N_1 &= \lfloor 56789 + 56789 \cdot 0.27324 \rceil 10^5 &= 72306{,}00000 \\ \Delta &= 56789{,}00000 \cdot 1.27324 - N_1 &= 2636 \\ N_2 &= \lfloor 56789 \cdot -0.04552 + \Delta \rceil &= 51 \end{aligned}$$

where the trick of rewriting $c_1 = 1 + 0.27324$ was used so that operands have no more than 5 digits. Note that for larger values of x, N may have 11 digits, and then N_1 may have 6. We will deal with this problem later. Also it is essential that C_1 has been chosen as the rounded value of C with $C_2 = C - C_1$, and similarly that c_1 is determined from C_1 with N_1 determined from c_1, etc., as shown above.

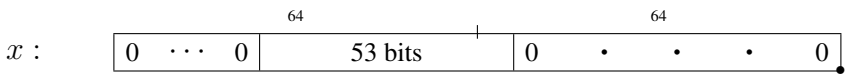
Here x was chosen such that its non-zero digits all fall in the most significant 5-digit word, allowing us to describe the process in a 5-digit fixed-point system. The argument reduction then proceeds as follows:

$x =$	56789	00000.		
$N_1C_1 =$	56789	13240.		
$x - N_1C_1 =$		$-13240.$		
$N_1C_2 =$		$-13279.$	71996	
$N_2C_1 =$		40.	05540	
$x - N_1C_1 - N_1C_2 - N_2C_1 =$			$-.33544$	
$N_2C_2 =$			$-.00009$	36666
$x - NC =$			.33534	63334

where the details of subtracting $N_1C_2 + N_2C_1$ are not shown. The result is then identical to the one of the previous example. However, although the wanted cancellation took place, note that the result is not quite correct as as an argument reduction, since C is not sufficiently close to $K = \pi/4$, more accuracy will be needed. □

3.1 A note on the arithmetic and the FMA instruction

We have so far assumed that we are working with an arithmetic unit having an FMA instruction, where for convenience in the description a fixed-point representation was used. In the examples we use 5 decimal arithmetic with the radix point located conveniently. In practice, with an IEEE 754 compatible ALU supporting the floating point, fused multiply-add instruction (FMA), the assumed fixed point arithmetic is automatically supported, even with the added flexibility that data need not be aligned at fixed positions, e.g., the argument x could as well be located anywhere within the extended "virtual" register like here



The essential property used here is that the product of two (say 64-bit) registers is exactly represented in an internal, double-length register, before the additive term is added. Provided

that the addition does not cause an overflow, the result is exactly represented within the temporary double length register. In the context used here there are situations where we can assure that by cancellation the final result will fit in a single register, hence there will be no loss of information by the final rounding in the floating point FMA instruction. However, there will be other situations where this kind of cancellation does not occur, and hence it will be necessary to recover the "tail" of a result which has been rounded away by the instruction.

3.2 Bounding $|x - NC|$

By a simple rewriting we obtain for $C < 2$ an upper bound when using round-to-nearest, $N = \lfloor xc \rceil$, with an error $\varepsilon = xc - \lfloor xc \rceil$,

$$\begin{aligned} |x - NC| &= |x - C\lfloor xc \rceil| \\ &= |x - C(xc - \varepsilon)| \quad \text{where } |\varepsilon| \le \tfrac{1}{2} \\ &= |x(1 - cC) + C\varepsilon| \qquad (2) \\ &\le |x| \cdot |1 - cC| + \frac{C}{2}, \qquad (3) \end{aligned}$$

thus we cannot in general expect that the choice of N will yield a perfect reduction, where $|x - NC| < \frac{C}{2}$, but must be satisfied that the "high-order" parts cancel such that $|x - NC| < 1$.

To get a general bound on $|1 - cC|$ assume that we are working in a k-digit, radix β, arithmetic, so let $R = \beta^k$, and define $c = \lfloor C^{-1}R \rceil / R$ as the rounded approximation to the reciprocal of C, then

$$|1 - cC| = |CC^{-1} - cC| = C|C^{-1} - c| \le \frac{C}{2R}. \qquad (4)$$

Combining (3) and (4) implies that for $\frac{1}{2} < C < 1$ and all x where $|x| < R$, it now follows that $|x - NC| < 1$. However, for larger C there are values of x where $|x - NC| \ge 1$ when N is calculated by rounding as $N = \lfloor xc \rceil$.

Figure 1 illustrates a situation when $1 < C < 2$, for positive x and a fairly large value of C, where the reduced argument $x - NC$ for $N = \lfloor xc \rceil = \lfloor xc \rfloor$ falls inside the interval $(-1; 1)$, whereas the one for $N + 1 = \lceil xc \rceil$ falls outside. Due to the rounding error ε, note that for such a large value of C it is quite unlikely that both points fall in the unit interval, in particular for $|x|$ close to R where $|x| \cdot |1 - cC|$ is close to $\frac{C}{2}$.

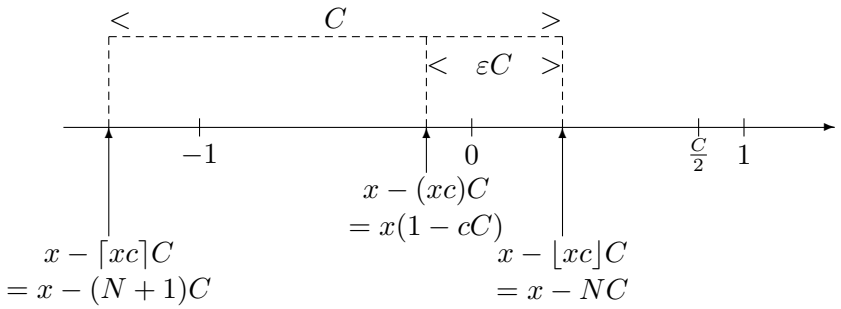


Figure 1: Range reduction for a fairly large value of C, for $x > 0$, $x(1 - cC) < 0$ and $\varepsilon > 0$.

For $1 < C < 2$ using (4) we have $|x| \cdot |1 - cC| < \frac{C}{2}$. Then since $|\varepsilon| < \frac{1}{2}$ assuming that $x(1 - cC) < 0$ and $\varepsilon > 0$, we have (cf. Figure 1) that

$$0 < x - \lfloor xc \rfloor C = x(1 - cC) + C\varepsilon < \frac{C}{2} < 1,$$

so $N = \lfloor xc \rfloor$ is the choice in this situation. By symmetry we find for $x(1 - cC) > 0$ and $\varepsilon < 0$ that xc should be rounded up, so here $N = \lceil xc \rceil$. Combining we find that $N = \lfloor xc \rceil$ is the choice when $x(1 - cC)$ and ε are of opposite signs.

When $x(1 - cC)$ and ε have the same sign, the situation is more complicated. If both are positive we have may situations as shown in Figures 2 and 3.

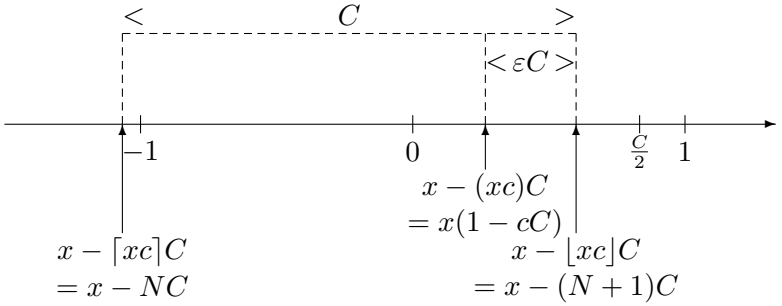


Figure 2: Range reduction when ε and $x(1 - cC)$ are both positive and $x > 0$.

However, note that ε is quite readily available as the fractional part, when calculating $\lfloor xc \rceil$ as a preliminary value of N. But for large values of x where $|x(1 - cC)|$ approaches $\frac{C}{2}$, if εC ($< \frac{C}{2}$) is also large, then $N = \lfloor xc \rceil = \lfloor xc \rfloor$ is not the right choice. Since it is easy to check ε against a bound, let us introduce a parameter α such that for $\varepsilon C \leq \alpha C \leq 1 - \frac{C}{2}$, implying that when $0 < \varepsilon \leq \alpha$ we can choose $N = \lfloor xc \rfloor$, since it is known that $|x(1 - cC)| < \frac{C}{2}$.

On the other hand when $\alpha < \varepsilon < \frac{1}{2}$, to assure that the lefthand value $x - \lceil xc \rceil C$ is inside the unit interval, it is sufficient to require $\alpha C - C \geq -1$ such that $\varepsilon C - C > -1$, allowing the choice $N = \lceil xc \rceil$. Hence we have two conditions on α, $1 - \frac{C}{2} \geq \alpha C \geq C - 1$ which is only possible if $C < \frac{4}{3}$, allowing us to choose $\alpha = \frac{C-1}{C}$ for $1 \leq C < \frac{4}{3}$.

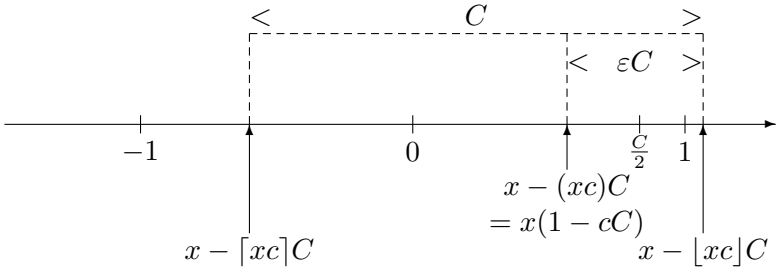


Figure 3: A critical range reduction situation

But for larger values of C an alternative approach must be used. Essentially the condition $x(1 - cC) + \varepsilon C < 1$ should be calculated to full accuracy, but this is equivalent to calculating $x - \lfloor xc \rceil C$. So let us instead of comparing ε against a constant, compare it to an expression which can be calculated at lower precision. Using the stronger condition

$$x(1 - cC) + \varepsilon C < C/2 \tag{5}$$

it is possible to compare against a function $\mu(x)$:

$$\varepsilon < \mu(x) = \tfrac{1}{2} - \left| x \tfrac{1-cC}{C} \right|,$$

where $\frac{1-cC}{C}$, $\left|\frac{1-cC}{C}\right| \leq \frac{1}{2R}$, is a system constant. We then have three algorithms for choosing N:

Theorem 3 *In a k-digit, radix-β arithmetic with $R = \beta^k$ and given constant $1 < C < 2$, let $c = \lfloor C^{-1}R \rceil / R$ and $\varepsilon = xc - \lfloor xc \rceil$, then $|x - NC| < 1$ holds for all $|x| < R$, provided that N is chosen as*

if $\frac{1}{2} < C \leq 1$**:** *Define $N = \lfloor xc \rceil$,*

if $1 < C < \frac{4}{3}$**:** *With $\alpha = \frac{C-1}{C}$ define N by*

if $\text{sign}(x(1 - cC)) \neq \text{sign}(\varepsilon)$
then $N = \lfloor xc \rceil$
else if $(|\varepsilon| < \alpha)$**xor**$(\varepsilon < 0)$ **then** $N = \lfloor xc \rfloor$, **else** $N = \lceil xc \rceil$.

if $\frac{4}{3} \leq C < 2$**:** *With $\mu(x) = \frac{1}{2} - \left| x \frac{1-cC}{C} \right|$ define N by*

if $\text{sign}(x(1 - cC)) \neq \text{sign}(\varepsilon)$
then $N = \lfloor xc \rceil$
else if $(|\varepsilon| < \mu(x))$**xor**$(\varepsilon < 0)$ **then** $N = \lfloor xc \rfloor$, **else** $N = \lceil xc \rceil$.

Proof: First consider the case $\frac{1}{2} < C < 1$ and assume that N is chosen as $N = \lfloor xc \rceil$. Then by the choice of c, we found in (4) that $|1 - cC| < \frac{C}{2R}$, which combined with (3) that $|x - NC| < 1$ is satisfied for all $|x| < R$. The trivial case $C = 1$ also holds since then $c = 1$ and thus $N = \lfloor x \rceil$.

For $1 < C < 2$, when $x(1 - cC)$ and ε are of opposite sign, it is easily seen by (4), that with $N = \lfloor xc \rceil$:

$$|x - NC| = \left| x - xcC + xcC - \lfloor xc \rceil C \right| = \left| x(1 - cC) + \varepsilon C \right| < \tfrac{C}{2} < 1.$$

For $1 < C < \frac{4}{3}$ with $\alpha = \frac{C-1}{C}$, assume that $x(1 - cC)$ as well as ε are non-negative. For $\varepsilon < \alpha$, then $\varepsilon C < \alpha C = C - 1 < \frac{1}{3}$, and by (4) it follows for all x, $|x| < R$ that $x(1 - cC) < \frac{2}{3}$, hence $|x - NC| < 1$ when choosing $N = \lfloor xc \rfloor = \lfloor xc \rceil$. For $\alpha \leq \varepsilon < \frac{1}{2}$ with $N = \lceil xc \rceil = \lfloor xc \rfloor + 1$, assuming that x is positive, then

$$\begin{aligned} x - NC &= x - xcC + xcC - \lceil xc \rceil C \\ &= x(1 - cC) + (xc - \lfloor xc \rfloor - 1)C \\ &= x(1 - cC) + \varepsilon C - C \\ &\geq x(1 - cC) + \alpha C - C = x(1 - cC) - 1 > -1, \end{aligned}$$

and $x - NC = x(1 - cC) + \varepsilon C - C < x(1 - cC) < \frac{C}{2} < 1$ by (4) since $|x| < R$.

When $\frac{4}{3} \leq C < 2$, still assuming that $x(1 - cC)$ as well as ε are non-negative, let the condition for choosing $N = \lfloor xc \rfloor$ be

$$x(1 - cC)C + \varepsilon C < \frac{C}{2} \quad \Longleftrightarrow \quad 0 \leq \varepsilon < \mu(x) = \tfrac{1}{2} - \left| x \tfrac{1-cC}{C} \right|,$$

implying $x - NC = x(1 - cC) + \varepsilon C < \frac{C}{2} < 1$. If $\varepsilon \geq \mu(x) = \frac{1}{2} - \left| x \frac{1-cC}{C} \right|$ let $N = \lceil xc \rceil$, then as above

$$\begin{aligned} x - NC &= x - xcC + xcC - \lceil xc \rceil C \\ &= x(1 - cC) + (xc - \lfloor xc \rfloor - 1)C \\ &= x(1 - cC) + \varepsilon C - C \\ . \quad &\geq x(1 - cC) + \mu(x)C - C = -\tfrac{C}{2} > -1, \end{aligned}$$

and as above $x - NC < 1$. The cases for $x < 0$ are similar.

The equivalent situation where both $x(1 - cC)$ and ε are negative follows similarly, reversing the choice of N (as implemented by the **xor** operation). □

Notes: For a specific value of C a particular algorithm is used, and the appropriate system constants are supplied. All algorithms require the value of $c = \lfloor RC^{-1} \rceil / R$ to full system accuracy, and for $1 < C < 2$ the sign of $1 - cC$ is needed. For $1 < C < \frac{4}{3}$ the constant $\alpha = \frac{C-1}{C}$ must be supplied, alternatively for $\frac{4}{3} \leq C < 2$ the constant $\rho = |\frac{1-cC}{C}|$ must be available for the calculation of $\mu(x) = \frac{1}{2} - |x|\rho$.

For the purpose of the first step of the argument reduction, using a single FMA operation, there must be some bounds on the operands for the reduction to be performed exact. The argument x supplied as the additive constant to the instruction must be representable in the basic system (at-most k digits) and $x - NC$ calculated in double precision, such that it can be computed exact and represented in $2k$ digits. Under the conditions of Theorem 3 it is then assured that the "high-order" k digits of $x - NC$ are all zero, thus its value can be represented exactly in the system, requiring no more than k digits. Thus while x is a number with at-most k digits, when C is a proper fraction of k digits, $\frac{1}{2} < C < 1$, it follows that c will have $k + 1$ digits as $1 < c < 2$, hence $N \approx xc$ may have $k + 1$ digits. This presents a problem if the multiplication by N is to be performed with a single FMA instruction, so for $N \geq R$ let $N' = N - R = \lfloor xc - R \rceil = \lfloor xc' + (R - x) \rceil$, and instead perform the reduction as $(x - RC) - N'C$. Note that this rewriting allows us to operate with one extra digit in the representation of c, which is crucial to minimize the bound on the value of $|1 - cC|$, while operating with k-digit operands. Hence for $C < 1$, whether $N \geq R$ or $N < R$, a single FMA instruction can perform the reduction, respectively with x or $x - RC$ as the additive term, where it may be noted that RC is just a shifted version of C.

For $1 \leq C < 2$ write $C = 1 + C'$ and use the rewriting $x - NC = x - N(1 + C') = (x - N) - NC'$, also allowing a simple use of the FMA instruction when performing the reduction.

Observation 4 *Under the assumptions of Theorem 3, given C and N, the expression $x - NC$ can be calculated exactly in a k-digit system by a single SUB instruction followed by a single FMA instruction. When $C > 1$ using the form $C = 1 + C'$, the reduction is performed as $x - NC = (x - N) - NC'$, and when $C < 1$ the reduction is $(x - RC) - N'C$ where $N = N' + R$. If $N < R$ in the latter case, then even the direct reduction $x - NC$ suffice.*

Let us thus illustrate how to handle larger values of x than used in Examples 1 and 2. Our intention is to use a emulated double-precision arithmetic, in particular the FMA instruction, we will first show a range-reduction over the double-precision range, but of a single-precision operand.

Example 3 Let us first perform an reduction of $x = 98765{,}00000$ in a 10-digit arithmetic, and then the same reduction in a 5-digit arithmetic, this being the reason for having only 5 digits in x. For $K = \pi/4$ we will use the approximation $C = 0.78539{,}81634$ as in the previous examples. We then have $c = 1 + 0.27323{,}95448$ and N' can be calculated by an FMA instruction as $N' = \lfloor x \cdot 0.27323{,}95448 \rceil + x - R = 25751{,}50363$ with then $N = N' + R$. Hence the reduction can be performed in 10-digit arithmetic as:

$$\begin{array}{ll} x = & 98765{,}00000. \\ \underline{RC =} & \underline{78539{,}81634.} \\ x - RC = & 20225{,}18366. \\ N'C = & 20225{,}18365.57904{,}33142 \\ \underline{x - NC =} & \underline{.42095{,}66858} \end{array}$$

Repeating the example, now using 5-digit arithmetic to emulate the 10-digit arithmetic, with the system constants being split accordingly, we have the following constants:

$$\begin{aligned} C = C_1 + C_2 &= 0.78540 - 0.0000018366 \\ c \;=\; 1 + c_1 + c_2 &= 1 + 0.27324 - 0.0000004552 \\ N = R + n_1 + n_2 &= R + 25752{,}00000 - 49637 \end{aligned}$$

Now the first reduction takes the form $(x - RC_1) - n_1C_1$, and proceeds as follows:

$x =$	98765	00000.	
$RC_1 =$	78540	00000.	
$x - RC_1 =$	20225	00000.	
$n_1C_1 =$	20225	62080.	
$x - RC_1 - n_1C_1 =$		−62080.	
$RC_2 =$		−18367.	
$x - RC_1 - n_1C_1 - RC_2 =$		−43714.	
$n_2C_1 =$		−38984.	−89980.
$x - NC_1 - RC_2 =$		−4729	−.10020
$n_1C_2 =$		−4729	−.61232
$x - NC_1 - (R + n_1)C_2 =$			.51212
$n_2C_2 =$			.09250 17221
$x - NC =$			.42095 66858

where not all details are shown, but (of course) delivering the same result. □

For the next example we choose $K = \pi/2$ with $C = \lfloor 10^{10}\pi/2 \rceil / 10^{10} = 1.57079{,}63268$, so now C falls in the interval $\frac{4}{3} < C < 2$ and comparison with the function value $\mu(x)$ is needed. Here $\frac{1}{2} < c < 1$, so $|N| < |x| < R$, but C must now be split.

Example 4 For $R = 10^{10}$ with the given value of C we find the following "system constants":

$$\begin{aligned} c &= \lfloor 10^{10}C^{-1} \rceil / 10^{10} = 0.63661{,}977240 \\ 1 - cC &= -5.41720 \cdots 10^{-11} \\ \left|\tfrac{1-cC}{C}\right| &= 3.44869 \cdots 10^{-11}. \end{aligned}$$

With $x = 98765{,}00000$, we have $\varepsilon = \lfloor xc \rceil - xc = -0.10860$ so $|\varepsilon| < \mu(x) = 0.15939$ hence $x(1 - cC)$ and ε have the same sign and $N = \lfloor xc \rfloor = 62875{,}75182$.

Splitting $C = 1 + C' = 1 + 0.57079{,}63268$ and calculating $x - NC$ as $(x - N) - NC'$ we get

$x =$	98765,00000.	
$N =$	62875,75182.	
$x - N =$	35889,24818.	
$NC' =$	35889,24818.	36444,14776
$x - NC =$		−.36444,14776

The equivalent reduction in 5-digit arithmetic emulating 10-digit arithmetic proceeds as follows:

$x =$	98765	00000.		
$n_1 =$	62876	00000.		
$x - n_1 =$	35889	00000.		
$n_1 C_1 =$	35889	62080.		
$(x - n_1) - n_1 C_1 =$		$-62080.$		
$n_2 =$		$-24818.$		
$x - N(1 + C_1) =$		$-37262.$		
$n_1 C_2 =$		$-23095.$	-61232	
$n_2 C_1 =$		$-14166.$	-11440	
$x - n_1 C_1 - n_1 C_2 - n_2 C_1 =$			$-.27328$	
$n_2 C_2 =$			.09116	14776
$x - NC =$			$-.36444$	-14776

□

Observe that for the cancellations to take place, the value of C as such is irrelevant, but C is supposed to be an approximation to the constant K (like $\pi/2$ or $\ln 2$), hence we want C to be as close as possible to K. The absolute error δ in the reduced argument $x - NC$ is then here obviously $\delta = N(K - C)$, so with "single" precision in the approximation of C to K, and since $|x| < R$ then $\delta < \frac{|x|}{2R} < \frac{1}{2}$, which is clearly not sufficient.

4 Obtaining sufficient accuracy

Let us then finally return to the accuracy in a "real" range-reduction, i.e., which accuracy is required in approximating the reduction constant K by C. The reduced argument $x - NC$, with $|x - NC| < 1$, must not deviate more from $x - NK$, than their difference yields an acceptable relative error. Thus let $\rho(x)$ be the relative error

$$\rho(x) = \left| \frac{N(C - K)}{x - NK} \right|, \tag{6}$$

we must know how close a floating point number x can be to an exact integer multiple of K, as this determines how small the denominator in (6) can be. This can be determined by an algorithm provided by Kahan [Kah83] (a C-program implementing the algorithm can be found at `http://http.cs.berkeley.edu/~wkahan`). E.g., for $K = \pi/2$ and exponents up to 128, it was found that for the floating point number $6411027962775774 \cdot 2^{-47}$ the distance to a multiple of $\pi/2$ is less than $6.1898 \cdot 10^{-19}$, corresponding to a loss of between 60 and 61 bits of accuracy. Let $\delta(K)$ denote the smallest value this distance can have for the range of possible values of x, then employing "quadruple" accuracy in the approximation C to K

$$\rho(x) = \left| \frac{N(C - K)}{x - NK} \right| < \frac{|x|}{2\delta(K)R^4} < \frac{1}{2\delta(K)R^2},$$

thus if (as very likely) $\delta(K) \geq R^{-1}$ then $\rho(x) < \frac{1}{2R}$, which is sufficient.

The absolute error is easily found to be less than $\frac{1}{2R^2}$, which implies that there is no need to calculate $x - NC$ to more accuracy than R^{-2}.

Example 5 Performing the same argument reduction of $x = 98765{,}00000$ as in the previous example, here not only emulating the 10-digit arithmetic in a 5-digit system, but now also doubling the accuracy in $C = \lfloor 10^{20}\pi/2 \rceil / 10^{20} = 1.57079{,}63267{,}94896{,}61923$, (e.g., using 5-digit, "quadruple" accuracy in the representation of C), C must now be split in five parts:

$$\begin{aligned} C &= 1 + 0.57080 - 0.36732 \cdot 10^{-5} - 0.05103 \cdot 10^{-10} - 0.38077 \cdot 10^{-15} \\ c = c_1 + c_2 &= 0.63662 - 0.22763 \cdot 10^{-5} \\ N = n_1 + n_2 &= 62876{,}00000 - 24818. \end{aligned}$$

The first reduction in 5-digit arithmetic takes the form $(x - N) - n_1 C_1$, such that due to cancellation $\left|(x - N) - n_1 C_1\right| < R$, and the argument reduction as an exact calculation then proceeds as:

$x =$	98765	00000.				
$N =$	62876	−24818.				
$x - N =$	35889	24818.				
$n_1 C_1 =$	35889	62080.				
$x - N - n_1 C_1 =$		−37262.				
$n_2 C_1 =$		−14166.	−11440			
$n_1 C_2 =$		−23095.	−61232			
$x - N - n_1 C_1 - n_2 C_1 - n_1 C_2 =$			−.27328			
$n_2 C_2 =$			.09116	14776		
$x - N(1 + C_1 + C_2) =$			−.36444	−14776		
$n_1 C_3 =$			−.03208	−56228		
$n_2 C_3 =$				01266	46254	
$x - N(1 + C_1 + C_2 + C_3) =$			−.33235	−59814	−46254	
$n_1 C_4 =$				−23941	−29452	
$n_2 C_4 =$					09449	94986
$x - NC =$			−.33235	−35873	−26251	−94986
$x - N\pi/2 =$			−.33235	−35873	−34562	−18541
$\Delta =$					08310	23555

so $x - NC \approx -0.33235{,}35873$ has an absolute error $8.31\ldots 10^{-12}$, corresponding to a relative error of $2.50\ldots 10^{-11}$. Note that the calculations to the right of the vertical line need not be performed, as the absolute error by truncating the calculations at this point can at most contribute an absolute error of less than 10^{-10}, which is sufficient provided that at most 5 digits can be lost due to x being very close to some multiple of $\frac{\pi}{2}$, as only 5 significant digits are expected. □

The difference between the results in Example 4 and Example 5 is exclusively due to the extra accuracy in the representations of $K = \pi/2$, both computations are exact. Actually, the result of Example 4 can be found as an intermediate result in Example 5.

As noted above there is no need to include terms smaller than R^{-2}. In practice using IEEE floating point arithmetic, the input operand would be a double precision (53-bit mantissa), and the arithmetic is assumed to be performed using the extended precision (64-bit) FMA instruction, to emulate "quadruple extended" (128-bit) arithmetic. Hence the absolute errors will be bounded by 2^{-128}, so even after cancellation of up to 61 bits, there is plenty of relative accuracy to deliver a double precision, 53-bit result for the range of operands $|x| < 2^{128}$. A comment on this is provided in the appendix, indicating how the algorithms can be significantly simplified.

5 Conclusions

We have thus essentially shown the following

Observation 5 *Assume given an arithmetic unit with a fixed-point FMA instruction, capable of calculating the exact value of an expression of the form $U = X \cdot Y + Z$, whenever X, Y and Z are representable operands and the result U is representable. Let the underlying number system of representable numbers consist of at most k-digit, radix β, scaled integers X of the form $X = x\beta^i$, where β does not divide x, and $0 \leq |x| < R$, $R = \beta^k$.*

Let C be a given representable quadruple precision constant, $\frac{1}{2} < C < 2$, on the form $C = C_1R^{-1} + C_2R^{-2} + C_3R^{-3} + C_4R^{-4}$, or for $C \geq 1$ the form $C = 1 + C_1R^{-1} + C_2R^{-2} + C_3R^{-3} + C_4R^{-4}$, where C_1, C_2, C_3 and C_4 are representable numbers. Then for any integer x of the form $x = \xi\beta^i$, $i \leq k$, with $|\xi| < R$, the exact value of $x - NC$ can be calculated in emulated double-precision such that $|x - NC| < 1$ with an absolute error less than $\frac{1}{2R^2}$ in approximating $|x - NK|$, whenever C approximates K such that $|C - K| < \frac{1}{2R^4}$, and N is chosen by Theorem 3.

Algorithms for range reduction in the two cases $C < 1$ and $C > 1$ are given in the appendix, assuming the availability of emulated double precision instructions,

References

[BDK+05] N. Brisebarre, D. Defour, P. Kornerup, J.-M. Muller, and N. Revol. A New Range-Reduction Algorithm. *IEEE Transactions on Computers*, 54(3):331–339, March 2005.

[Cod82] W.J. Cody. Implementation and Testing of Function Software. In P.C. Messina and A. Murli, editors, *Problems and Methodologies in Mathematical Software Production*, LNCS-142. Springer Verlag, Berlin, 1982.

[CW80] W. Cody and W. Waite. *Software Manual for the Elementary Functions*. Prentice Hall, Englewood Cliffs, NJ, 1980.

[Kah83] W. Kahan. Minimizing $q^*m - n$. text accessible electronically at http://http.cs.berkeley.edu/~wkahan/. At the beginning of file "nearpi.c", 1983.

[LBD03] R.C. Li, S. Bondo, and M. Daumas. Theorems on Efficient Argument Reduction. In *Proc. 16th IEEE Symposium on Computer Arithmetic*. IEEE Computer Society, 2003.

[Mul97] J.-M. Muller. *Elementary Function Evaluation: Algorithms and Implementation*. Birkhäuser, Boston, Basel, Berlin, 1997.

[PH83] M. Payne and R. Hanek. Radian Reduction for Trigonometric Functions. *SIGNUM Newsletter*, 18, 1983.

A Algorithms

We can now formulate algorithms for the argument reduction, first assuming that $\frac{1}{2} < C < 1$. To simplify the description we will describe it using double-precision arithmetic, assuming this being emulated in the base arithmetic, e.g., using a double double-extended (128-bit mantissas) floating point FMA instruction, $\texttt{fma}(a, b, c) = a + b * c$, emulated by the basic (64-bit) FMA and other emulated instructions like $\texttt{sub}(a, b) = a - b$ and $\texttt{mul}(a, b) = a * b$, producing the most significant rounded (128-bit) result.

Algorithm 6 (Extended range argument reduction for $\frac{1}{2} < \mathbf{K} < 1$)

Stimulus: *A single-precision (k-digit) argument $x = \xi \cdot \beta^k$ satisfying $|x| < R = \beta^{2k}$, and a quadruple precision constant, $\frac{1}{2} < C < 1$, on the form $C = C_1 + C_2$, where C_1 and C_2 are double precision representable numbers such that $|C - K| < \frac{1}{2R^4}$ where K is a given constant. Also needed is $c = 1 + c'$ where $c' = \lfloor C^{-1}R^2 \rceil / R^2 - 1$, to be used to derive $N = \lfloor xc \rceil$ as specified by Theorem 3.*

Response: *The exact value of the reduced argument $x - NC$ satisfying $|x - NC| < 1$, where the absolute error satisfies $\big|(x - NC) - (x - NK)\big| = \big|N(C - K)\big| < \frac{1}{2R^2}$.*

Method:

$N := \mathtt{fma}(x, x, c');$

if $|N| < R$ **then**

$s := \mathtt{fma}(x, -N, C_1);$	$\{s = x - NC_1,\ \|s\| < 1$, *canceling integer part*$\}$
$s := \mathtt{fma}(s, -N, C_2);$	$\{s = x - NC_1 - NC_2 = x - NC\}$

else

$s := x - R;$	
$N' := \mathtt{fma}(s, x, c');$	$\{N' = \lfloor x + xc' \rceil - R,\ N' < R\}$
$s := \mathtt{sub}(x, R * C_1);$	
$s := \mathtt{fma}(s, N', C_1);$	$\{s = x - NC_1,\ \|s\| < 1\}$, *canceling integer part*
$s := \mathtt{sub}(s, R * C_2);$	$\{s = x - NC_1 - RC_2,\ \|s\| < 1\}$
$t_1 := \mathtt{mul}(N', C_2);$	$\{t_1 = \lfloor R^2 N' C_2 \rceil / R^2,\ \|t_1\| < 1\}$
$s := \mathtt{sub}(s, t_1);$	
$t_2 := \mathtt{fma}(-t_1, N', C_2);$	$\{t_2 = N'C_2 - \lfloor R^2 N' C_2 \rceil / R^2,\ \|t_2\| < \frac{1}{2}R^{-2}\}$
$s := \mathtt{sub}(s, t_2);$	$\{s = x - N(C_1 + C_2) = x - NC,\ \|s\| < 1\}$

Where: *The underlying number system of double precision, representable numbers consists of at most $2k$-digit, radix β, scaled integers of the form $z = \xi\beta^i$, where β does not divide ξ, and $0 \le |\xi| < R^2 = \beta^{2k}$.*

Algorithm 7 (Extended range argument reduction for $\mathbf{1 < K < 2}$)

Stimulus: *A single-precision (k-digit) argument $x = \xi \cdot \beta^k$ satisfying $|x| < R = \beta^{2k}$, and a quadruple precision constant, $1 < C < 2$, on the form $C = 1 + C_1 + C_2$, where C_1 and C_2 are double precision representable numbers such that $|C - K| < \frac{1}{2R^4}$ where K is a given constant. Also needed is c where $c = \lfloor C^{-1}R^2 \rceil / R^2$, to be used to derive N as specified by Theorem 3.*

Response: *The exact value of the reduced argument $x - NC$ satisfying $|x - NC| < 1$, where the absolute error satisfies $\big|(x - NC) - (x - NK)\big| = \big|N(C - K)\big| < \frac{1}{2R^2}$.*

Method:

$t_0 := \mathtt{sub}(x, N);$	$\{t_0 = x - N,\ \|t_0\| < R^2\}$
$s := \mathtt{fma}(t_0, -N, C_1);$	$\{s = x - N(1 + C_1),\ \|s\| < 1\}$
$t_1 := \mathtt{mul}(N, C_2);$	$\{t_1 = \lfloor R^2 N C_2 \rceil / R^2,\ \|t_1\| < 1\}$
$s := \mathtt{sub}(s, t_1);$	
$t_2 := \mathtt{fma}(-t_1, N, C_2);$	$\{t_2 = NC_2 - \lfloor R^2 N C_2 \rceil / R^2,\ \|t_2\| < \frac{1}{2}R^{-2}\}$
$s := \mathtt{sub}(s, t_2);$	$\{s = x - N(1 + C_1 + C_2) = x - NC,\ \|s\| < 1\}$

Where: *The underlying number system of double precision, representable numbers consists of at most $2k$-digit, radix β, scaled integers of the form $z = \xi\beta^i$, where β does not divide ξ, and $0 \le |\xi| < R^2 = \beta^{2k}$.*

Note: Both algorithms can be simplified, as t_2 need not be calculated and subtracted, since its contribution is of the same order as the error in approximating K by C. Thus the last four instructions of the algorithms may be substituted by $s := \mathtt{fma}(s, -N', C_2)$ resp. $s := \mathtt{fma}(s, -N, C_2)$, with an error bound of $\frac{1}{2R^2}$, hence the total absolute error is bounded by R^{-2}.