LARGE COMPUTATIONAL ERRORS

Mihail Konstantinov, Petko Petkov

Abstract. *We derive bounds for the average rounding errors in double precision binary arithmetic (DPBA) obeying the 754-2019 IEEE Standard for floating-point arithmetic. The results are based on an alternative representation of the sets of normal and subnormal machine numbers. We also consider numerical computations in DPBA with unexpectedly large errors which may lead to numerical catastrophes without warning for the user. This may cause catastrophes with material damage and human casualties. Numerical examples are given which illustrate the theoretical results. The computations are done in MATLAB computing environment.*

Key words: floating-point arithmetic, rounding errors, large error computations, numerical catastrophes.

Mathematics Subject Classification: 65G50

1. Introduction

In this paper we present new bounds for the average rounding errors in double precision binary arithmetic (DPBA) obeying the 754-2019 IEEE Standard for floating-point arithmetic [2]. Note that usually maximal rounding errors are analyzed. At the same time average rounding errors may be more relevant in many practical problems. The results obtained are based on an alternative representation of machine numbers and a new topology of both the sets of normalized and subnormal machine numbers.

We also consider numerical computations in DPBA with unexpectedly large errors. This may lead to numerical catastrophes which appear without any warning for the user. In turn, large error computations may cause real life catastrophes with material damage and human casualties. The computations are done in MATLAB® computing environment.

The DPBA consists of the set of machine numbers (MN) which are binary fractions including zero, and the rules for performing arithmetic operations with MN. For machine arithmetic with different precision the results are similar to these presented below since they are formulated mainly in terms of the rounding unit of DPBA. In what follows the symbol `>>`

denotes the start of a MATLAB® command line, while ":=" means "equal by definition".

We denote by $\mathbb{Z} = \{0, \pm 1, \pm 2, \dots\}$ the set of integers and by $\mathbb{R} = (-\infty, \infty)$ and $\mathbb{R}_+ = [0, \infty) \subset \mathbb{R}$ the sets of real and non-negative real numbers, respectively. For p, $q \in \mathbb{Z}$ and $p \le q$ we set $\mathbb{Z}(p, q) = \{p, p + 1, \dots, q\} \subset \mathbb{Z}$. The set of binary fractions $A \times 2^B$ is denoted as $\mathbb{F}$, where $A, B \in \mathbb{Z}$ and the integer A is odd or zero.

For $a, b \in \mathbb{R}_+$ and $a < b$ the notation $\mathcal{J}(a, b)$ stands for any interval with end points a and b. Let $\mathbb{M} = \mathbb{M}_- \cup \mathbb{M}_+$ be the set of MN, where $\mathbb{M}_+$ is the set of non-negative MN and the symbol Inf. Thus we assume that $0 \in \mathbb{M}_+$ and deal with rounding of non-negative numbers as a function from $\mathbb{R}_+$ to $\mathbb{M}_+$. The set $\mathbb{M}$ contains normal and subnormal numbers. The number of elements of a set M is denoted as $\operatorname{card}(M)$.

Some important MN used throughout this paper are as follows.

1. The rounding unit $\rho = 2^{-53} \simeq 1.1102 \times 10^{-16}$
2. The machine epsilon (denoted as `eps`) $\varepsilon = 2^{-52} \simeq 2.2204 \times 10^{-16}$
3. The number $N = 2^{52} \simeq 4.5036 \times 10^{15}$ of MN in an interval $[2^m, 2^{m+1})$
4. The maximal normal MN (denoted as `realmax`) $R = 2^{1023}(2 - \varepsilon) \simeq 1.7977 \times 10^{308}$
5. The minimal positive normal MN (denoted as `realmin`) $r = 2^{-1022} \simeq 2.2251 \times 10^{-308}$
6. The minimal positive subnormal MN $\eta = 2^{-1075} \simeq 2.4703 \times 10^{-324}$

2. Another View on DPBA

2.1. Normal machine numbers

The normal machine numbers (NMN) from $\mathbb{M}_+$ are binary fractions of the form

$$\mu(m, n) := 2^m(1 + n\varepsilon) = 2^{m-52}(2^{52} + n) \tag{1}$$

where $m \in \mathbb{Z}(-1022, 1023)$ and $n \in \mathbb{Z}(0, N - 1)$. The quantity ε is called machine epsilon of DPBA and is denoted in MATLAB® as `eps`. It has the property that the quantities $\mu = \mu(m, n)$ and

$$\mu^+ := \mu(m, n + 1) = \mu(m, n) + 2^m\varepsilon, \ n \in \mathbb{Z}(0, N - 2) \tag{2}$$

are two consecutive machine numbers, such as $\mu(0,0) = 1$ and $\mu(0,1) = 1+\varepsilon$. The set of NMN is denoted as $\mathbb{M}_{\rm nor}$. If we consider $\mu(m,n)$ as an expression for any $m,n \in \mathbb{Z}$ then for $n = N$ we have $\mu(m,N) = \mu(m+1,0)$. Furthermore, we denote by $\mu^- = \mu(m,n-1) = \mu(m,n) - 2^m\varepsilon$, $n \in \mathbb{Z}(1, N-1)$ the machine number to the left of μ.

Usually NMN are written in binary positional form

$$\mu(m,n) = 2^m\left(1 + \mathrm{bin}(b_1, b_2, \ldots, b_{52})\right),\ b_k \in \{0,1\} \tag{3}$$

where $\mathrm{bin}(b_1, b_2, \ldots, b_{52}) := \sum\limits_{k=1}^{52} b_k 2^{-k} \in [0, 1-\varepsilon]$.

The *rounding unit* of DPBA is the quantity $\rho = \mu(-53,0) = 2^{-53} \simeq 1.1102 \times 10^{-16}$. We have $\rho^{-1} = 2^{53} = 9{,}007{,}199{,}254{,}740{,}992 \simeq 9.0072 \times 10^{15}$. Based on the above definitions, we see that $\mu(N,n) = N + n$, $n \in \mathbb{Z}(0, N-1)$. In particular $2N-1$, $2N$ and $2N+2$ are three consecutive NMN, while $2N+1 = 2^{53}+1$ is *not* a MN and is rounded to $2N$.

The *normal range* $\mathbb{N} = [r, R] \subset \mathbb{R}_+$ for DPBA consists of the positive real numbers between the machine numbers $r := \mu(-1022, 0) = 2^{-1022} \simeq 2.2251 \times 10^{-308} \in \mathbb{M}$, denoted in MATLAB® as `realmin`, and $R := \mu(1023, N-1) = 2^{1023}(2-\varepsilon) \simeq 1.7977 \times 10^{308} \in \mathbb{M}$, denoted in MATLAB® as `realmax`. Note that $2^{1024} \notin \mathbb{M}$. The MN lying in $\mathbb{N}$ are said to be *normal* and the set of these numbers is denoted as $\mathbb{M}_{\rm nor} = \mathbb{M} \cup \mathbb{N} \subset \mathbb{M}$.

2.2. Subnormal machine numbers

Positive MN between the quantities r and $\omega := \varepsilon\, r = 2^{-1074} \simeq 4.9407 \times 10^{-324} \in \mathbb{M}$ are called *subnormal* [2]. The zero is a machine number which is usually excluded from the set of subnormal numbers. Nevertheless we shall treat 0 as a subnormal number. Therefore the non-negative subnormal MN (SbMN) in this paper are

$$s_n = n\omega,\ n \in \mathbb{Z}(0, N-1) \tag{4}$$

and the set $\mathbb{M}_{\rm sub} \subset \mathbb{M}$ of SbMN has $\mathrm{card}(\mathbb{M}_{\rm sub}) = N$ elements. We have $\mathbb{M} = \mathbb{M}_{\rm nor} \cup \mathbb{M}_{\rm sub}$ and $\mathbb{M}_{\rm nor} \cap \mathbb{M}_{\rm sub} = \emptyset$.

Real numbers which are sufficiently larger than R are rounded in DPBA to the symbol Inf which is the machine analogue of infinity. Positive numbers x which are sufficiently smaller than $s_1 = \omega$ are rounded to $s_0 = 0$.

3. Rounding Errors in the Normal Range

3.1. Rounding function

Let $x \in J_m := [2^m, 2^{m+1}) \subset \mathbb{N}$, $m \in \mathbb{Z}(-1022, 1023)$. The interval J_m contains N machine numbers $2^m(1 + n\varepsilon)$, $n \in \mathbb{Z}(0, N-1)$. Thus there are $2046 = 1023 - (-1022) + 1$ intervals J_m, and $\mathrm{card}(\mathbb{M}_{\mathrm{nor}}) := 2046 \times N \simeq 9.2144 \times 10^{18}$ normal machine numbers in $\mathbb{M}_+$. Since according to (4) there are N subnormal non-negative numbers, the total number of elements of $\mathbb{M}_+$ is $\mathrm{card}(\mathbb{M}_+) = 2047 \times N \simeq 9.2189 \times 10^{18}$.

Since we deal with relative rounding errors then without loss of generality we consider only the interval J_0 which contains the machine numbers $x_n := 1 + n\varepsilon \in J_0$, $n \in \mathbb{Z}(0, N-1)$.

Denote by $\mathrm{flo}(x) \in \mathbb{M}$ the *rounded value* of $x \neq 0$, and let

$$\mathrm{err}(x) = \frac{|\mathrm{flo}(x) - x|}{x} \tag{5}$$

be the relative error in rounding x to the nearest machine number $\mathrm{flo}(x)$. For $x \in \mathcal{J}(a, b)$, where $r \le a < b \le R$, the maximal rounding error for DPBA obeying the IEEE standard satisfies the upper bound

$$\sup\{\mathrm{err}(x)\colon x \in \mathcal{J}(a, b)\} < \rho \tag{6}$$

For particular values of a and b, however, the left-hand side of (6) may be considerably smaller than ρ.

The rounded value $\mathrm{flo}(x)$ of x may also be written as

$$\mathrm{flo}(x) = x + \Delta(x) = x(1 + \delta(x)),\ |\delta(x)| < \rho$$

For moderate values of $x \in \mathbb{R}$ the absolute signed rounding error $\Delta(x) = x\delta(x)$ may be found by the MATLAB® command `>> sym(x,'e')` in the form $\Delta(x) = \mathrm{eps} \times E(x)$, where $E(x)$ is a rational fraction. For example, for $x = 1.1$ we have `>> sym(1.1,'e') = 11/10 + 2*eps/5`. Thus $E(1.1) = 2/5$ and $\delta(1.1) = (8/11)\rho \simeq 0.7273\rho$.

The numbers x_n, and only they, are rounded exactly, i.e. $\mathrm{flo}(x_n) = x_n$ and $\mathrm{err}(x_n) = 0$. More precisely, rounding is performed according to the formulas

$$\mathrm{flo}(x) = \begin{cases} 1 & x_0 = 1 \le x \le x_0 + \rho \\ x_{2p-1} & x_{2p-1} - \rho < x < x_{2p-1} + \rho \\ x_{2p} & x_{2p} - \rho \le x \le x_{2p} + \rho \\ 2 & x_{N-1} + \rho = 2 - \rho \le x < 2 \end{cases} \tag{7}$$

where $p \in \mathbb{Z}(1, N/2-1) = \mathbb{Z}(1, 2^{51}-1)$. Note that numbers $x \in [2-\rho, 2) \subset J_0$ are rounded to $2 \notin J_0$.

For each p denote

$$R_{2p-1} = (x_{2p-1} - \rho, x_{2p-1} + \rho), \ R_{2p} = [x_{2p} - \rho, x_{2p} + \rho] \tag{8}$$

The intervals $R_{2p-1}, R_{2p} \subset [r, R]$ are *sets of attraction* of the points x_{2p-1} and x_{2p}, respectively, since numbers $x \in R_q$ are rounded to x_q.

Let x_n and x_{n+1} be two consecutive MN and set

$$\xi_n = \frac{x_n + x_{n+1}}{2} = 1 + (2n+1)\rho \notin \mathbb{M} \tag{9}$$

Consider the restriction err_n of the function err on the interval $[x_n, x_{n+1}]$. Here $\mathrm{flo}(x)$ is either x_n or x_{n+1}, see (7), and the graph of the function err_n consists of two hyperbolas $1 - x_n/x$, $x_n \le x \le \xi_n$ and $x_{n+1}/x - 1$, $\xi_n < x \le x_{n+1}$. Hence the function err_n has maximum

$$e_n := \mathrm{err}_n(\xi_n) = \frac{\rho}{\xi_n} = \frac{\rho}{1 + (2n+1)\rho} = \frac{1}{2n + 2N + 1} \tag{10}$$

The variable e_n decreases in n. In particular e_n is maximal for $n = 0$,

$$e_0 = \mathrm{err}(1 + \rho) = \frac{\rho}{1 + \rho} < \rho \tag{11}$$

and minimal for $n = N - 1$,

$$e_{N-1} = \mathrm{err}(2 - \rho) = \frac{\rho}{2 - \rho} = \frac{\rho}{2} + \mathrm{O}(\rho^2) \tag{12}$$

The relations (7) define an increasing step function $\mathrm{flo}\colon \mathbb{N} \to \mathbb{M}$ with jumps of magnitude 2ρ at the points $x = y_n$. The graph of the continuous piece-wise differentiable function $\mathrm{err}\colon \mathbb{N} \to \mathbb{N}$ resembles a saw.

In view of (11), the analysis of rounding errors is usually based [1] on the estimate

$$\mathrm{err}(x) \le \frac{\rho}{1 + \rho} < \rho = 2^{-53}$$

which is almost achievable near the left end of the interval $[1, 2)$. However, close to the right end of this interval the estimate for err is rather of the type

$$\mathrm{err}(x) < \frac{\rho}{2} = 2^{-54}$$

see (12). Moreover, statistically the behavior of the relative rounding errors may not correspond to the upper bounds e_n. Hence it is useful to introduce estimates for *average* rather than for *maximal* errors. Such estimates are inequalities of the form

$$\mathrm{err}(x) \leq E = C\rho + \mathrm{O}(\rho^2) \tag{13}$$

for certain positive constants $C < 1$. Two average bounds that we consider are

$$E_{\max} = \frac{1}{N}\sum_{n=0}^{N-1} e_n \tag{14}$$

and

$$E_{\mathrm{int}} = \int_1^2 \mathrm{err}(x)\,\mathrm{d}x. \tag{15}$$

In what follows we compute the constants C in (13) for the average rounding bounds (14) and (15). In particular we show that $C = C_{\max} = \log(2) \simeq 0.6931$ for the bound (14) and $C = C_{\mathrm{int}} = \log(2)/2 \simeq 0.3466$ for the bound (15).

3.2. Average maximal error

Consider first the average bound (14), where the local maximal errors e_n are defined in (10). We have

$$E_{\max} = \frac{E}{N} = 2\rho E, \quad E = \sum_{n=0}^{N-1} e_n$$

It may be shown that the following relations hold

$$E = \sum_{n=0}^{N-1} \frac{1}{2n+2N+1} = \sum_{k=N+1}^{2N} \frac{1}{2k-1} = \frac{\log(2)}{2} + \mathrm{O}(\rho^2)$$

Hence $2\rho E < \rho\log(2) + \mathrm{O}(\rho^3)$ and

$$E_{\max} = C_{\max}\,\rho + \mathrm{O}(\rho^3),\ C_{\max} = \log(2) \simeq 0.6931 \tag{16}$$

3.3. Average integral error

The average integral rounding error for $a \leq x \leq b$ is defined as

$$E_{\mathrm{int}}(a,b) = \frac{1}{b-a}\int_a^b \mathrm{err}(x)\,\mathrm{d}x$$

In our case $a = 1$, $b = 2$ and hence

$$E_{\mathrm{int}} = E_{\mathrm{int}}(1,2) = \int_1^2 \mathrm{err}(x)\,\mathrm{d}x = \sum_{n=0}^{N-1} E_n$$

where

$$E_n = \int_{x_n}^{x_{n+1}} \mathrm{err}(x)\,\mathrm{d}x,\ n \in \mathbb{Z}(0, N-1) \tag{17}$$

and $x_N := 2$. Furthermore we have $E_n = F_n + G_n$, where

$$F_n = \int_{x_n}^{\xi_n} \frac{x - x_n}{x}\,\mathrm{d}x = \rho - x_n \log\left(\frac{\xi_n}{x_n}\right)$$

and

$$G_n = \int_{\xi_n}^{x_{n+1}} \frac{x_{n+1} - x}{x}\,\mathrm{d}x = x_{n+1} \log\left(\frac{x_{n+1}}{\xi_n}\right) - \rho$$

Setting $\rho_n = \rho/x_n$ we obtain

$$\frac{F_n + G_n}{x_n} = (1 + 2\rho_n)\log\left(1 + 2\rho_n\right)) - 2(1 + \rho_n)\log(1 + \rho_n)$$
$$= \rho_n^2 - \rho_n^3 + \mathrm{O}\big(\rho_n^4\big) = \frac{\rho^2}{x_n^2} - \frac{\rho^3}{x_n^3} + \mathrm{O}\big(\rho^4\big)\,.$$

Hence for small ρ we have the estimate $(F_n + G_n)/\rho < \rho_n + \mathrm{O}\big(\rho^3\big)$ and

$$\frac{1}{\rho}\sum_{n=0}^{N-1}(F_n + G_n) < \sum_{n=0}^{N-1} \rho_n + \mathrm{O}(\rho^2)$$

Next we obtain

$$\sum_{n=0}^{N-1} \rho_n < \frac{1}{2}\log\left(\frac{2N-1}{N-1}\right) + \frac{1}{4(2N-1)} - \frac{1}{4N}$$
$$= \frac{1}{2}\log\left(\frac{2-2\rho}{1-2\rho}\right) - \frac{\rho - 2\rho^2}{4(1-\rho)} = \frac{\log(2)}{2} + \frac{\rho}{4} + \mathrm{O}(\rho^2)$$

Hence

$$E_{\mathrm{int}} \le C_{\mathrm{int}}\rho + \mathrm{O}(\rho^2),\ C_{\mathrm{int}} := \frac{\log(2)}{2} \simeq 0.3466. \tag{18}$$

4. Rounding Errors in the Subnormal Range

4.1. Rounding function

A real non-negative number $x < r$ is rounded by the rounding function $\mathrm{flo}^* : [0, r) \to [0, r)$ to the closest sub-normal MN $s_n = n\omega$ of the form (4) as follows:

$$\mathrm{flo}^*(x) = \begin{cases} s_0 = 0, & 0 \le x \le \eta \\ s_{2k-1}, & s_{2k-1} - \eta < x < s_{2k-1} + \eta \\ s_{2k}, & s_{2k} - \eta \le x \le s_{2k} + \eta \\ r, & r - \eta < x < r \end{cases} \tag{19}$$

Here $k \in \mathbb{Z}(1, N/2)$, the value of $\omega = \varepsilon r$ is 2^{-1074} and $\eta = \omega/2$. Since numbers from the interval $\mathbb{M}_0 := [0, \eta] \subset \mathbb{R}_+$, $\eta = 2^{-1075} \simeq 2.4703 \times 10^{-324}$, are rounded to 0 we shall refer to $\mathbb{M}_0$ as the *machine zero*.

It follows from (19) that the maximum δ_n of the relative rounding error

$$\mathrm{err}^*(x) := \left| \frac{\mathrm{flo}^*(x) - x}{x} \right| \tag{20}$$

for $x \in (s_n, s_{n+1})$ is achieved for $x = \sigma_n := (s_n + s_{n+1})/2$. We have

$$\delta_n = \mathrm{err}^*(\sigma_n) = \frac{1}{2n+1},\ n \in \mathbb{Z}(0, N-1) \tag{21}$$

4.2. Average maximal error

It follows from (21) that the maximal relative error δ_n decreases from $\delta_0 = 1$ to $\delta_{N-1} = \rho/(1-\rho)$. The mean of the maximal errors is

$$E^*_{\max} = \frac{1}{N} \sum_{n=0}^{N-1} \delta_n = \frac{1}{N} \sum_{k=1}^{N} \frac{1}{2k-1}$$

or

$$E^*_{\max} = \rho\left(2\log(2) + \gamma + \log(N) + \mathrm{O}\big(N^{-2}\big)\right) = C(\rho)\rho + \mathrm{O}(\rho^3)$$

where

$$C(\rho) = C_1 - \log(\rho),\ C_1 := \log(2) + \gamma \simeq 1.2704 \tag{22}$$

We see that the mean term $C(\rho)\rho$ in the upper bound for $E^*_{\max}$, defined by (22) and (22), is not a linear function of ρ. This is a major difference in the properties of the rounding function in the subnormal and the normal ranges of DPBA. We stress finally that for DPBA with $\rho = 2^{-53}$ we have $C(\rho) \le 38.0072$ and hence $E^*_{\max} \le 38.01\rho$.

4.3. Average integral error

The average integral rounding error for numbers from the subnormal range $[0, r)$ of the DPBA is

$$E^*_{\rm int} := \frac{1}{r}\int_0^r {\rm err}^*(x){\rm d}x = \sum_{n=0}^{N-1} E^*_n \tag{23}$$

Here

$$E^*_n := \frac{1}{r}\int_{s_n}^{s_{n+1}} {\rm err}^*(x){\rm d}x = F^*_n + G^*_n$$

and

$$F^*_n = \frac{1}{r}\int_{s_n}^{\sigma_n} \left(1 - \frac{s_n}{x}\right){\rm d}x = \rho - 2\rho n \log\left(1 + \frac{1}{2n}\right)$$
$$G^*_n = \frac{1}{r}\int_{\sigma_n}^{s_{n+1}} \left(\frac{s_{n+1}}{x} - 1\right){\rm d}x = -\rho + 2\rho(n+1)\log\left(1 + \frac{1}{1+2n}\right)$$

Therefore $E^*_n = 2\rho\psi(n)$ and

$$E^*_{\rm int} = 2\rho\,\Phi(\rho),\ \ \Phi(\rho) := \sum_{n=0}^{N-1}\psi(n)$$

where

$$\psi(n) := (n+1)\log\left(1 + \frac{1}{1+2n}\right) - n\log\left(1 + \frac{1}{2n}\right)$$

It may be shown that $\psi(n) < 1/(1+2n)$ and

$$\Phi(\rho) < \sum_{k=1}^{N}\frac{1}{2k-1} < \frac{C(\rho)}{2}$$

Hence

$$E^*_{\rm int} \le C(\rho)\rho + {\rm O}(\rho^3) \tag{24}$$

where the expression $C(\rho)$ is determined by (22). We see that the estimates for the maximal average error and the integral average error for rounding of numbers from the subnormal range of DPBA coincide!

5. Rounding Errors in the Supnormal Range

The number $R = 2^{1023}(2 - \varepsilon)$ (written as `realmax` in MATLAB®) is the maximal machine number in DPBA, or the maximal element of $\mathbb{M}$. It may be computed as `>> 2^1023*(2-eps)`. The previous relative to R machine number is $R^- = 2^{1023}(2 - 2\varepsilon)$, while the quantity 2^{1024} is not a machine number and is rounded to Inf. Hence if we try to compute R as `>> 2^1024-2^971` the answer shall be `Inf`.

The quantity $S = 2^{969}(2 - \varepsilon) \in \mathbb{N}$ has the following interesting property. The number $R + S$ is rounded to R while $R + S^+$ is rounded to Inf, where $S^+ = \mu(970, 0)$ is the machine number right next to S. Thus the command `>> realmax+2^969*(2-eps)` gives `realmax`, while the command `>> realmax+2^970` returns `Inf`.

As a summary, real numbers $x \in (R, R+S]$ are rounded to $\mathrm{flo}^{**}(x) = R$ with a relative error

$$\left|\frac{x - \mathrm{flo}^{**}(x)}{x}\right| \le \frac{R + S - R}{R + S} = \frac{S}{R + S} = \frac{\rho}{2} \tag{25}$$

In turn, the average integral rounding error in this interval is

$$\frac{1}{S}\int_R^{R+S} \frac{x - R}{x}\,\mathrm{d}x = \frac{1}{S}\left(S - R\log\left(1 + \frac{S}{R}\right)\right) \simeq \frac{\rho}{4} \tag{26}$$

We shall refer to the interval $(R, R+S] \subset \mathbb{R}_+$ as the *supnormal range* for DPBA. Finally, real numbers $x > R + S$ are rounded to `Inf` and the relative error in this case is infinite. The interval $(R, +\infty) \subset \mathbb{R}_+$ is said to be the *infinity range* for DPBA.

6. Summary of Rounding Errors

The above results may be summarized as follows. For this particular standard of machine arithmetic, namely DPBA, a rounding function $\mathrm{flo} : \mathbb{R} \to \mathbb{M}_+ \cup \{\mathrm{Inf}\}$ is defined with the following properties, illustrated for definiteness in the case of non-negative numbers in the next table.

Range	Maximal error	Integral error	Remarks
Machine zero $[0,\eta]$	1	1	Rounding to `0`
Subnormal range (η,r)	$C(\rho)\rho$	$C(\rho)\rho$	$C(\rho)=C_1-\log(\rho)$ $C_1=\log(2)+\gamma\simeq 1.2704$
Normal range $[r,R]$	$C_{\max}\,\rho$	$0.5\,C_{\max}\,\rho$	$C_{\max}=\log(2)\simeq 0.6931$
Supnormal range $(R,R+S]$	$0.5\,\rho$	$0.25\,\rho$	Rounding to `realmax`
Infinity range $(R+S,\infty)$	∞	∞	Rounding to `Inf`

Table 1. Average relative rounding errors

7. Large Error Computations

The exact value of a certain number $x \in \mathbb{R}$ and its rounded value $\mathrm{flo}(x) \in \mathbb{M}$ in DPBA are different unless $x \in \mathbb{M}$. Moreover, the rounding rule $\mathrm{flo} : \mathbb{R} \to \mathbb{M}$ may not be a function in the standard mathematical sense. In particular $\mathrm{flo}(x)$ depends on two informal factors as follows.

1. The way the argument x is written; e.g. for $x = 1 + \mathrm{eps} \in \mathbb{M}$ we have $\mathrm{flo}(1 + \mathrm{eps}) = 1 + \mathrm{eps}$ but $\mathrm{flo}(1 + \mathrm{eps}/2 + \mathrm{eps}/2) = 1$.
2. The properties of the software and hardware realization of the particular DPBA obeying the IEEE Standard for floating-point arithmetic.

These effects and the rounding rules themselves may cause unexpectedly large computational errors without any warning to the user. We recall that the calculations in DPBA are in the following order: function evaluation; calculation in brackets; multiplication (division) and addition (subtraction) from left to right.

7.1. Calculations with large numbers

Interesting rounding effects may be observed for numbers far from the boundaries of the normal range of DPBA, e.g. for large numbers that are close to `realmax`, $R = 2^{1023}(2 - \mathrm{eps}) = 2^{1024} - 2^{971}$. For example, the quantity $A = R + 2^{970} = R + 2^{969} + 2^{969} = 2^{1024} - 2^{970}$ computed by the MATLAB® command `>> realmax+2^970` gives `Inf`. At the same time if we compute A by the command `>> realmax+2^969+2^969` the answer is `realmax` because `realmax+2^969` is rounded twice to `realmax`. Hence we shall get `realmax` if we add any number of quantities `2^969` to `realmax`. A counterpart of the latter result is the sum of 1 and any number of quantities `eps/2`. The result may be arbitrarily large but the computed result is equal to 1 due to the fact that `1+eps/2` is repeatedly rounded to 1. The same is valid for the sum of 2^{53} and any numbers of ones.

Also, 2^{1024} is set to `Inf` in DPBA and the same is valid for quantities written in the form `2^1024-D` for any $D \in \mathbb{N}$. In particular, the exact value of `realmax` is $2^{1024} - 2^{971}$ but, written in this form in DPBA, it will be set to `Inf`. To input the correct value of `realmax` in DPBA (except as `realmax`) we may write it as `>> 2^(1024-m)*(2^m-2^(m-53))` for any $m \in \mathbb{Z}(1, 1023)$. Another brutal example is the calculation $2^{1024} - 2^{1023} - 2^{1023}$. The result is 0 but written in this form in DPBA it will be interpreted as `Inf` and we have the somehow surprising result `Inf = 0`. In turn, the ratio $2^{1024}/2^{1024}$ and the difference $2^{1024} - 2^{1024}$ are interpreted in DPBA as NaN (from Not a Number) instead of 1 and 0, respectively.

7.2. Calculations with moderate numbers

Very often simple computations with quite moderate numbers may be contaminated with large rounding errors which appear without any warning. Consider first an example involving a positive integer parameter n. For $n = 1$ the relative error may reach 1, or 100%, while for $n > 1$ this error is practically unlimited! This example is based on the expression

$$F_n(x) = x^{-n} \left((1+x)^n - \sum_{k=0}^{n-1} \binom{n}{k} x^k \right), \ x \neq 0$$

which is equal to 1. However, the computed value $\mathrm{flo}(F_n(x))$ of $F_n(x)$ exposes strange behavior for values of x far from the boundaries of the normal range of DPBA. This behavior is in full accordance with the arithmetic rules of DPBA.

Let us start with the simplest expression $F_1(x) = (1 + x - 1)/x$, $x \neq 0$. For $x = \rho$ the computed value $\mathrm{flo}(F_1(\rho))$ is 0 and for $x = \rho(1 + 2\rho)$ the computed value $\mathrm{flo}(F_1(\rho(1 + 2\rho)))$ is $2/(1 + 2\rho) \simeq 2$ instead of 1. In both cases the relative error is 100%. The expression $F_1(x)$ may be coded as `>> x=-3*eps:eps/1000:3*eps;F1=(1+x-1)./x;` and further on plotted by the command `>>plot(x,F1,'b',x,ones(1,length(x),'r')`. We should have $F_1(x) = 1$ but in fact $\mathrm{flo}(F_1(x))$ varies from 0 to 2 with

100% relative error. More precisely, for $x \geq 0$ and $k = 0, 1, 2, \ldots$, we have

$$\mathrm{flo}(F_1(x)) = \begin{cases} 0 & 0 \leq x \leq \rho \\ (4+4k)\rho\, x^{-1} & (3+4k)\rho \leq x \leq (5+4k)\rho \\ (2+4k)\rho\, x^{-1} & (1+4k)\rho < x < (3+4k)\rho \end{cases}$$

Similar expressions are valid for $x < 0$.

Weird behavior demonstrates the expression $\mathrm{flo}(F_n(x))$ for $n \geq 2$ since its values may differ drastically from the exact value $F_n(x) = 1$. In particular we have the awful result $\mathrm{flo}(F_n(x)) = -n/x$, $-\rho/2 \leq x \leq \rho$, $x \neq 0$, with unlimited relative error. For $0 < x \leq \rho$ the sign of the computed result is wrong but this cannot make things worse.

For example $\mathrm{flo}(F_2(\rho)) = -2/\rho \simeq -1.8014 \times 10^{16}$ instead of the exact answer $F_2(\rho) = 1$. More generally, for moderate values of $m = 1, 2, \ldots$ we have $\mathrm{flo}(F_2(2m\rho)) = 0$ and

$$\mathrm{flo}(F_2(4m-1)\rho) \simeq \frac{2}{(2m-1)^2\rho}, \ \mathrm{flo}(F_2(4m-3)\rho) \simeq \frac{-2}{(2m-1)^2\rho}$$

These astonishing errors are due neither to the computer system used nor to possible hardware defects. The are completely in accordance with the rules governing the DPBA. In addition, the computed values of $F_n(x)$ strongly depend on the order of summation in the corresponding formula.

In particular, for $n = 2$ the three expressions $y_1 = ((1+x)^2 - 1 - 2x)/x^2$, $y_2 = ((1+x)^2 - 2x - 1)/x^2$ and $y_3 = ((1+x)^2 - (1+2x))/x^2$ for computing $F_2(x)$ are equivalent. However, in DPBA they produce quite different results for certain values of x due to violation of the associative rule for addition. Some numerical results are presented at the table below.

x	y_1	y_2	y_3
$2^{26} \times \varepsilon$	1	1	1
$2^{25.5} \times \varepsilon$	2.751	2.000	2.000
$2^{25} \times \varepsilon$	0	0	0
$\varepsilon/2$	-1.801×10^{16}	-1.801×10^{16}	-1.801×10^{16}
$\varepsilon/3$	-2.702×10^{16}	-2.027×10^{16}	-4.053×10^{16}
$\varepsilon/4$	-3.603×10^{16}	-3.603×10^{16}	0
$\varepsilon/5$	-4.504×10^{16}	-5.630×10^{16}	0
$\varepsilon/8$	-7.206×10^{16}	0	0
2^{-537}	-8.998×10^{161}	0	0
2^{-538}	$-$Inf	NaN	NaN

Table 2. Unlimited chaotic errors

In order to avoid such catastrophic computations, attempts to “cheat” the DPBA may sometimes be applied by preliminary symbolic simplification of the corresponding formulas. For example, the simplification of $F_n(x)$ as a symbolic expression directly gives the correct answer $F_n(x) = 1$. If however similar expressions arise in the execution of complicated computational algorithms this may not be possible.

Consider next the following numerical test. Choose an integer $p > 1$ such that $p + 1$ is not an integer power of 2 and let V be a p–vector with rational elements $V_k = 1+kh$ between 1 and 2, i.e. $V = [1+h, 1+2h, \ldots, 1+ph]$, where $h = 1/(p+1)$. The vector V may be computed by the commands `>> h=1/(p+1);V=1+h:h:2-h;` The aim of the numerical test is to estimate the mean value $U(p) := \frac{1}{p}\sum_{k=1}^{p} U_k(p)$ of the quantities $U_k(p) = |\mathrm{flo}(V_k) - V_k|/(V_k\rho)$. The computed values `Up` of $U(p)$ are expected to be close to the coefficient C in the bound $C\rho$ for the average rounding errors in the normal range of DPBA. The experiment is performed by the MATLAB® command

```
>>Up=double(mean(abs(sym(V,'e')./V-ones(1,length(p))))*2/eps)
```

We recall that the MATLAB® function `sym(a,'e')` computes the rounded value $\mathrm{flo}(a)$ of the array a plus an approximation of the form $\mathrm{eps} \times A$ of the absolute rounding error, where A is an array of the type of a with rational elements of order 1. For instance, for the 2-vector $a = [1 + 1/3, 1 + 2/3]$ the command `>> sym([1+1/3,1+2/3],'e')` gives `[4/3-eps/3,5/3-2*eps/3]`, i.e. in this case $A = [-1/3, -2/3]$. The result is different if we use the command `>> sym([4/3,5/3],'e')` which gives `[4/3-eps/3,eps/3+5/3]` with $A = [-1/3, 1/3]$.

The results of this numerical experiment are shown at the next two tables. We stress that if $p+1$ was an integer degree of 2 then the rounding errors would be zero and $U(p) = 0$. The same is true if we choose V by a quasi-random number generator, e.g. `V=1+rand(1,p)` since in this case the elements of V are machine numbers.

p	10	20	30	40	50	60	70	80	90	100
$U(p)$	.503	.408	.387	.392	.507	.377	.371	.418	.375	.361

Table 3. Coefficients for relative rounding errors

p	9	99	999	999
h	0.1	0.01	0.001	0.0001
$U(p)$	0.613	0.363	0.672	0.360

Table 4. More coefficients for relative rounding errors

7.3. Automatic voting calculations

Consider the allocation of M seats among n parties $P_1, P_2, \dots, P_n$ with votes $v_1, v_2, \dots, v_n$. According to the method of Hamilton fractional seats $m_k = v_k M/V$ are calculated, where $V = v_1 + v_2 + \cdots + v_n$. Denote $m_k = n_k + r_k$, where $n_k = \mathrm{fix}(m_k)$ is the integer part of m_k and $r_k \in [0, 1)$ is the remainder m_k. Initially each party P_k obtains n_k seats. If there are p seats left then p parties with largest remainders obtains one additional seat. If this procedure cannot be realized because of equal remainders then a tie is carried out. In bi-proportional systems as in Bulgaria several ties may be necessary and the calculations may be done automatically. Moreover, possible ties may be done before the election results are known by ordering the parties preliminary.

Surprisingly, rounding errors with relative accuracy $\rho \simeq 10^{-16}$ may contaminate voting calculations although the number V does not exceed 10^7. The problem is in the MATLAB command `>>[a,b] = max(u)`, where a is the maximum element of the vector u and b is its index. In case of equal elements b is the minimal index. However, when the vector u is computed in DPBA, elements which should be equal in fact differ within quantities of order ρ. For example, the vector $[4/3 - 1, 1/3]$ has equal elements $1/3$ and the correct value of the index b is 1. At the same time the command `>> [a,b] = max([4/3-1,1/3])` gives $a = 0.3333, b = 2$. The reason is that the first element $4/3 - 1$ is rounded in DPBA as $1/3 - \mathrm{eps}/3$ which is less then the rounded value $1/3 - \mathrm{eps}/12$ of the second element. As a result MATLAB® "decides" that the index of the maximal element is 2. This leads to wrong determination of the index b and to wrong allocation of seats among political parties. This may have dangerous political consequences.

Such cases illustrate the influence of rounding on voting calculations. They took place in experiments with the software for voting calculations in Bulgaria. The solution is to use integer remainders $R_k = Vr_k = Mv_k$ instead of fractional remainders r_k.

7.4. Evaluation of Lipschitz functions

Let $\mathbf{f} : \mathbb{R}^n \to \mathbb{R}^n$ be a vector function of vector argument $\mathbf{x} \in \mathbb{R}^n$. The function evaluation $\mathbf{y} = \mathbf{f}(\mathbf{x}) \neq \mathbf{0}$ for large arguments $\mathbf{x}$ may be accompanied by large errors. Indeed let $\mathbf{x}^*$ be the rounded value of the argument $\mathbf{x}$ such that $\|\mathbf{x}^* - \mathbf{x}\| \leq \Gamma\|\mathbf{x}\|\rho$. where the constant Γ depends on n and on the norm used. For the Euclidean norm we have $\Gamma = \sqrt{n}$. Let the function $\mathbf{f}$ be Lipschitz continuous with constant $\Lambda > 0$, i.e.

$$\|\mathbf{f}(\mathbf{x}_1) - \mathbf{f}(\mathbf{x}_2)\| \leq \Lambda \|\mathbf{x}_1 - \mathbf{x}_2\|$$

Even if the machine calculation $\mathbf{y}^* = \mathbf{f}(\mathbf{x}^*)$ is done exactly, we have

$$\frac{\|\mathbf{y}^* - \mathbf{y}\|}{\|\mathbf{y}\|} \leq L\rho, \ L = \frac{\Lambda\Gamma\|\mathbf{x}\|}{\|\mathbf{y}\|} \tag{27}$$

see [1]. The constant L is the *relative condition number* of the computational problem $\mathbf{y} = \mathbf{f}(\mathbf{x})$. This constant will be large if Λ and/or $\|\mathbf{x}\|$ are large and if $\|\mathbf{y}\|$ is small (note that Γ is usually not large).

The computational problem is *well conditioned in DPBA* if $L\rho \ll 1$. In this case one may expect about $-\log_{10}(L\rho)$ true decimal digits in the computed solution. On the contrary, when $L\rho$ is close to 1 the problem is *badly conditioned* and there may be no true digits in the computed solution.

This leads to the next important rule in numerical computations. *If the initial data x and some of the intermediate computational results are large and the result y is small then large relative errors in the computed result are to be expected.* The famous cancellation arguments in the analysis of the subtraction of close numbers are particular cases of this rule.

As an example consider the computation of the quantities $\alpha_m = \sin\left(\pi\left(0.5 + 2^m\right)\right)$ and $\beta_m = \cos\left(\pi 2^m\right)$ for positive integers m. The quantities α_m, β_m should be equal to 1 but they are not. Some results are presented at the next table.

m	48	49	50	51	52	53	54
s_m	0.9998	0.9903	0.9783	0.9883	−0.5240	−0.8926	−0.8049
c_m	0.9994	0.9976	0.9905	0.9622	0.8517	0.4509	−0.5934

Table 5. Function evaluation for large arguments

7.5. Evaluation of Hölder functions

Consider the function evaluation $\mathbf{y} = \mathbf{f}(\mathbf{x}) \neq \mathbf{0}$, where the function $\mathbf{f}$ is Hölder continuous with constant $\Lambda > 0$ and exponent $\alpha \in (0, 1)$, i.e.

$$\|\mathbf{f}(\mathbf{x}_1) - \mathbf{f}(\mathbf{x}_2)\| \leq \Lambda \|\mathbf{x}_1 - \mathbf{x}_2\|^\alpha$$

The machine computation of $\mathbf{y}$ may be accomplished with large errors especially when α is close to 0. Such cases arise in e.g. the computation of multiple zeros ξ of a smooth function F, where $F(\xi) = F'(\xi) = \cdots =$

$F^{m-1}(\xi) = 0$ $(m \geq 2)$ and $\alpha = 1/m$. For Hölder continuous functions the analogue of (27) is

$$\frac{\|\mathbf{y}^* - \mathbf{y}\|}{\|\mathbf{y}\|} \leq H\rho^\alpha, \quad H = \frac{\Lambda\Gamma^\alpha \|\mathbf{x}\|^\alpha}{\|\mathbf{y}\|} \tag{28}$$

To obtain meaningful numerical results it is necessary to have $H\rho^\alpha < 1$. But even in this case the computational errors may be relatively large since ρ^α may not be small enough. For example, if $\alpha = 1/4$ then $\rho^\alpha \simeq 10^{-4}$.

To illustrate the above concepts consider the complex polynomial of degree $n \geq 3$. Note that for $n = 2$ there are specialized algorithms for finding the roots with high accuracy.

$$F(z) = z^n + c_1 z^{n-1} + \cdots + c_n = (z - r_1)(z - r_2)\cdots(z - r_n) \tag{29}$$

Set $\mathbf{c} = [c_1; c_2; \ldots; c_n]$ and $\mathbf{r} = [r_1; r_2; \ldots; r_n]$. The vectors $\mathbf{c}$ and $\mathbf{r}$ are connected by the Viéte map $\Sigma\colon \mathbb{C}^n \to \mathbb{C}^n$, namely $\mathbf{c} = \Sigma(\mathbf{r})$, where $c_k = \Sigma_k(\mathbf{r}) = (-1)^k \sigma_k(r_1, r_2, \ldots, r_n)$, $k \in \mathbb{Z}(1, n)$. Here σ_k is the k-th symmetric function of n variables. The function Σ is polynomial and hence Lipschitz continuous. The inverse function $\mathbf{f} = \Sigma^{-1}$ is of complicated algebraic structure, implicit for $n \geq 5$ and is Hölder continuous of exponents as small as $\alpha = 1/n$.

Computational problems arising in solving Hölder problems such as algebraic equations with multiple roots are considered below. The MATLAB® code for solving algebraic equations of type (29) is `roots(c0)`, where $c_0 = [1; c_1; c_2; ...; c_n] \in \mathbb{C}^{n+1}$. It produces a column vector with the computed roots. The code computes the eigenvalues of the companion matrix of the polynomial (29) by the QR algorithm.

The polynomial $(z - 1)^n$ has zero $r_1 = 1$ of multiplicity n. The implementation of the code `roots` for several values of n gives the following results. For a given n we denote by $\mathbf{r}^*$ the vector of computed roots, by $\mathbf{r}$ the vector of exact roots equal to 1, and

$$H = \frac{\|\mathbf{r}^* - \mathbf{r}\|}{\|\mathbf{r}\|}\rho^{-1/n}, \ \rho = 2^{-53}$$

We have

n	3	4	5	6	7
H	2.1848	2.1153	1.7577	2.1880	1.7852

Table 6. Solving algebraic equations of degree n

The relative error in the computed roots is of order $2\rho^{1/n} = 2^{1-53/n}$ and it may be unacceptable even for small values of the degree n. At the same time the code `roots` works with degrees n of order up to several thousands and this is an important advantage.

Note that together with the code `roots` we may find the zeros of a polynomial also by the codes `solve` and `fzero` from MATLAB. The latter two codes are intended for solving general finite equations. They produce solutions close to the nearest machine numbers.

Below we compare the main advantages and disadvantages of the codes `roots`, `solve` and `fzero` for solving algebraic equations of degree n. The code `solve(...)` is used as `double(solve(...))`.

Code	Advantages	Disadvantages
`roots`	Works with n up 5000. Relatively fast.	Large errors for multiple roots even for low multiplicity.
`solve`	Very accurate including for multiple roots.	Works with n up 500. May be slow for large n.
`fzero`	Very accurate and fast.	Finds only one root. May not work properly for roots of even multiplicity.

Table 7. Comparison of three solvers for algebraic equations

7.6. Subtraction without cancellation

In this section we consider the errors in subtraction of finite binary fractions of type $A \times 2^B$, where $A, B \in \mathbb{Z}$ and A is either an odd number or zero. These fractions may or may not be machine numbers. Denote the set of such fractions as $\mathbb{F}$. Obviously $\mathbb{M} \subset \mathbb{F}$ and the set $\mathbb{F}$ is closed under addition and multiplication. For example, the quantities $1/\rho + 1$ and $1 + \rho$ are not machine numbers in DPBA while $1/\rho - 1$ and $1 - \rho$ are.

The machine subtraction of binary fractions may be done either exactly or with rounding errors. Usually the subtraction $b - a$, either approximate or exact, of close numbers a, b is considered as a dangerous operation due to *cancellation* of the information coded in the left-most digits of a and b. At the same time rounding is often neglected as an immediate reason for loss of accuracy since the machine subtraction of close machine numbers is done exactly. The key words here are "machine numbers". If binary fractions are subtracted in DPBA and at least one of them is not a machine number then the relative error of the subtraction may be unlimited. The next example illustrates this phenomenon.

Setting $x^* = \mathrm{flo}(x) \in \mathbb{M}_+$ for $x \in \mathbb{N}$, the machine subtraction of $a, b \in \mathbb{N} \cap \mathbb{F}$ $(a < b)$ is done as $\mathrm{flo}(b - a) = (b^* - a^*)^*$. Let $a = 1 + \rho$ and $b = 1+\rho(1+\rho(1+2\rho)) = 1+\rho+\rho^2+2\rho^3$. The exact result is $b-a = \rho^2(1+2\rho)$. Next we obtain $a^* = 1$, $b^* = 1 + 2\rho$ and $\mathrm{flo}(b - a) = b^* - a^* = 2\rho$. Thus $\mathrm{flo}(b-a) = \beta(\rho)(b-a)$, where $\beta(\rho) = 2/\rho + \mathrm{O}(1)$. The computed difference is 1.8014×10^{16} times larger than the exact difference! We stress again that this is not a result of cancellation of left-most digits but of "pure" rounding.

A lighter version of this phenomenon is the subtraction of close numbers a, b from the set of attraction of a given machine number. Here $a^* = b^*$ and $\mathrm{flo}(b - a) = 0$ with 100% relative error.

Bibliography

[1] N. Higham, M. Konstantinov, V. Mehrmann, P. Petkov, The sensitivity of computational control problems, *IEEE Control Systems Magazine*, Vol. 24, 2004, 28–43, doi:10.1109/MCS.2004.1272744

[2] IEEE, *754–2019 IEEE Standard for Floating-Point Arithmetic*, 2019, standards.ieee.org/standard/754-2019.html

[3] MathWorks, *MATLAB, Version R2019a*, MathWorks Inc, 2019, Natick, Massachusetts, mathworks.com/products/matlab

Mihail Konstantinov[1,*], Petko Petkov[2]
[1] University of Architecture, Civil Engineering and Geodesy, 1421 Sofia,
[2] Technical University of Sofia, 1157 Sofia, `php@tu-sofia.bg`
[*] `Corresponding author: misho.konstantinov@gmail.com`