Bounds on Runs of Zeros and Ones for Algebraic Functions

Tomas Lang
Dept. Electrical and Computer Eng.
University of California at Irvine
Irvine, CA 92697 USA
tlang@uci.edu

Jean-Michel Muller
CNRS-Laboratoire LIP
Ecole Normale Superieure de Lyon
46 Allee d'Italie, 69364 Lyon Cedex 07, France
Jean-Michel.Muller@ens-lyon.fr

Abstract

This paper presents upper bounds on the number of zeros and of ones after the rounding bit for algebraic functions. These functions include reciprocal, division, square root, and reciprocal square root, which have been considered in previous work. We here propose simpler proofs for the previously given bounds and generalize to all algebraic functions. We also determine cases for which the bound is achieved for square root.

As is mentioned in the previous work, these bounds are useful for determining the precision required in the computation of approximations in order to be able to perform correct rounding. We consider here rounding to nearest, but the results can be easily extended to other rounding modes.

1. Introduction

This paper presents upper bounds on the number of zeros and ones after the rounding bit for algebraic functions. These functions include reciprocal, division, square root, and reciprocal square root, which have been considered in previous work [1] [2] [5] [4]. We here propose simpler proofs for the bounds given in [1] and generalize to all algebraic functions. We also determine cases for which the bound is achieved for square root. As already discussed for reciprocal square root in [1], except for reciprocal, division, square root, and norms, the bounds obtained are not achieved for precisions up to double precision and it is not known if there is a precision for which they are achieved.

As is mentioned in the previous work, these bounds are useful for determining the precision required in the computation of approximations in order to be able to perform correct rounding. We consider here rounding to nearest, but the results can be easily extended to other rounding modes.

1.1 Basic idea and example

Our approach for the bound on the maximum number of zeros is based on the following:

1. Develop a radix-2 digit-recurrence algorithm for the function[1]. Since we are going to use the algorithm to determine the bounds, to simplify this analysis we concentrate on algorithms of the restoring type.
2. Determine a bound on the minimum value of the (non zero) residual after a one in the rounding bit position is computed.
3. Determine the number of consecutive zeros that are produced by this minimum residual.

This approach can be used for any function for which a digit recurrence algorithm can be developed. We show that this is possible for the class of algebraic functions.

We now develop an example of the use of this procedure and then generalize to the class of algebraic functions. Consider the computation of the cubic root. That is, compute $y = x^{1/3}$ with $1/2 \leq x < 1$ and operand and result of n bits.

We define a radix-2 digit-recurrence algorithm that produces one bit of the result each iteration. That is, after iteration j the partial result is

$$y[j] = \sum_{i=1}^{j} y_i 2^{-i} \quad \text{with } y_i \in \{0, 1\}$$

The residual after iteration j is

$$w[j] = 2^j(x - y[j]^3)$$

[1]Note that the implementation of the digit-recurrence algorithm results in a simple method for correct rounding; this is true even in the case in which the result is obtained in signed-digit representation, where the computation of the rounding includes the sign of the final residual. Consequently, with this implementation the bounds presented in this paper are not necessary; however, this implementation might be too complicated or slow, so that other algorithms based on approximations are preferable.

0-7695-1150-3/01 $10.00 © 2001 IEEE

resulting in the recurrence

$$w[j+1] = 2w[j] - 3y[j]^2 y_{j+1} - 3y[j] y_{j+1}^2 2^{-(j+1)} - y_{j+1}^3 2^{-2(j+1)} \quad (1)$$

with $w[0] = x$. The selection function has to produce the minimum nonnegative $w[j+1]$. Consequently,

$$y_{j+1} = \begin{cases} 0 & \text{if } 2w[j] < 3y[j]^2 + 3y[j]2^{-(j+1)} + 2^{-2(j+1)} \\ 1 & \text{otherwise} \end{cases}$$

Since $y < 1$, and by construction $y[j] \leq y$, we obtain $y[j] < 1$. Therefore, for $j \geq 1$

$$3y[j]^2 + 3y[j]2^{-(j+1)} + 2^{-2(j+1)} \leq 61/16 < 4$$

Consequently, we get

$$y_{j+1} = 1 \text{ if } \ (2w[j] \geq 4) \quad (2)$$

We now determine a nonzero lower bound on $w[n+1]$, when $y_{n+1} = 1$. For this, we determine the (maximum) number of fractional bits of $w[n+1]$ (width of $w[n+1]$). The width of $w[j+1]$ is the maximum width of the terms in the recurrence. Each of the terms has the following maximum width:

1. The initial condition is $w[0] = x$, which has n fractional bits. Moreover, the effect of this initial value is "shifted" one position left each iteration, resulting in a term of $n - j - 1$ fractional bits.
2. The term $3y[j]^2$ has $2j$ fractional bits.
3. The term $3y[j]2^{-(j+1)}$ has $2j + 1$ fractional bits.
4. The term $2^{-2(j+1)}$ has $2j + 2$ fractional bits.

Consequently, the width of $w[j+1]$ is

$$widthw[j+1] \leq \max[(n-j-1), (2j+2)]$$

For $j = n$ this is $2n + 2$. Consequently, a nonzero lower bound on $w[n+1]$ is $2^{-(2n+2)}$

So, how many consecutive 0s in y can occur after this minimum $w[n+1]$? Each 0 multiplies the residual by 2. From equation (2) we conclude that a 1 will certainly occur when the value of the residual becomes 2. Consequently[2],

$$maxrun \leq 2n + 3$$

We generalize the idea described above to any algebraic function, as defined below. From this generalization, we obtain upper bounds on the maximum runs of zeros and of ones, after the rounding position. The class of algebraic functions includes the functions considered previously. In Section 4, we compare with the previously reported results.

[2]In Section 3 we analyze whether this bound is achieved

1.2 Algebraic functions

We generalize the idea described above to any algebraic function. An *Algebraic Function* f is a function for which there exists a 2-variable polynomial P with integer coefficients such that

$$y = f(x) \Leftrightarrow P(x, y) = 0 \quad (3)$$

Examples are:

Division, reciprocation $y = a/x$, with $P(x, y) = a - xy$;

Square roots $y = \sqrt{x}$, with $P(x, y) = x - y^2$;

Roots $y = x^{1/q}$, with $P(x, y) = x - y^q$ (q integer);

Square root reciprocal $y = 1/\sqrt{x}$, with $P(x, y) = 1 - xy^2$;

Norms $y = \sqrt{x^2 + v^2}$, with $P(x, y) = x^2 + v^2 - y^2$;

Normalization $y = x/\sqrt{x^2 + v^2}$, with $P(x, y) = x^2 - y^2(x^2 + v^2)$;

We assume that we perform calculations in an n-bit binary number system and that $1/2 \leq x < 1$ and $1/2 \leq y < 1$.[3] We also assume that, in the considered domain of computation the function $y \rightarrow P(x, y)$ is decreasing[4], and that

$$\frac{\partial P}{\partial y}(x, y) \neq 0 \quad (4)$$

To illustrate the development we use as running example the function $y = x^{3/5}$ with $1/2 \leq x < 1$. The corresponding polynomial is

$$P(x, y) = x^3 - y^5$$

1.3 Radix-2 restoring digit-recurrence algorithm for algebraic function

As indicated before, at step j of a digit-recurrence algorithm we have already computed

$$y[j] = 0.y_1 y_2 \ldots y_j = \sum_{i=1}^{j} y_i 2^{-i}$$

We define a *residual* $w[j]$ as

$$w[j] = 2^j P(x, y[j]) \quad (5)$$

Since the representation of y is non-redundant (restoring algorithm), the sequence $y[j]$ goes to y if and only if :

$$y_{j+1} = \begin{cases} 1 & \text{if} \quad y[j] + 2^{-(j+1)} \leq y \\ 0 & \text{otherwise} \end{cases} \quad (6)$$

[3]The generalization to other ranges is discussed later.
[4]Otherwise, it suffices to exchange P and $-P$.

Since the function $y \to P(x,y)$ is decreasing, (6) is equivalent to

$$y_{j+1} = \begin{cases} 1 & \text{if} \quad P(x,\, y[j] + 2^{-(j+1)}) \geq 0 \\ 0 & \text{otherwise} \end{cases} \tag{7}$$

In order to use (7), let us evaluate $2^{j+1}P(x,\, y[j]+2^{-(j+1)})$. We have

$$\begin{aligned} & 2^{j+1}P(x,\, y[j] + 2^{-(j+1)}) \\ = \; & 2^{j+1}\left[P(x,\, y[j]) + \textstyle\sum_{i\geq 1} \frac{2^{-i(j+1)}}{i!}\frac{\partial^i P}{\partial y^i}(x,\, y[j])\right] \\ = \; & 2w[j] + \frac{\partial P}{\partial y}(x,\, y[j]) + \textstyle\sum_{i\geq 2} \frac{2^{(-i+1)(j+1)}}{i!}\frac{\partial^i P}{\partial y^i}(x,\, y[j]) \\ = \; & 2w[j] - C[j] \end{aligned}$$

Note that this is the definition of $C[j]$.

Consequently, the selection function and the next residual are computed as follows:

- If $2w[j] \geq C[j]$ then $y_{j+1} = 1$ and $w[j+1] = 2w[j] - C[j]$
- If $2w[j] < C[j]$ then $y_{j+1} = 0$ and $w[j+1] = 2w[j]$

For our example

$$C[j] = 5y[j]^4 + 10y[j]^3 2^{-(j+1)} + \ldots.$$

2 Longest run of zeros and of ones for algebraic functions

2.1 Longest run of zeros

As indicated before, we proceed as follows:

1. Determine a lower bound on the (non-zero) value of $w[n+1]$ when $y_{n+1} = 1$.
2. Use the selection function to determine an upper bound on the number of consecutive zeros.

Let us now determine the smallest possible nonzero value of $w[j]$. As said in the introduction, P is a 2-variable polynomial with integer coefficients. Define $A_{p,q}$ as the coefficient of $x^p y^q$ in P, so that

$$P(x,y) = \sum A_{p,q} x^p y^q$$

and $\alpha_{p,q}$ as the largest integer such that $|A_{p,q}| \geq 2^{\alpha_{p,q}}$ (with $\alpha_{p,q}$ not defined for $A_{p,q} = 0$).

Since x is an n-bit normalized fraction and $y[j]$ is a j-bit normalized fraction, the term $A_{p,q}x^p y[j]^q$ is a multiple of $2^{\alpha_{p,q} - np - jq}$. Hence, if we define ν_j as

$$\nu_j = \min_{p,q}\,(\alpha_{p,q} - np - jq) \tag{8}$$

then $w[j] = 2^j P(x, y[j])$, if not equal to zero, is larger or equal to $2^{j+\nu_j}$. In particular,

$$w[n+1] \geq 2^{n+1+\nu_{n+1}}$$

For our example ($P(x,y) = x^3 - y^5$) we get $A_{30} = |A_{05}| = 1$ so that $\alpha_{30} = \alpha_{05} = 0$ and

$$\nu_{n+1} = -5(n+1)$$

Now we use the selection function to determine the maximum number of zeros. This is based on the following two properties:

1. If $y_{j+1} = 0$ then $w[j+1] = 2w[j]$
2. When $2w[j] \geq C[j]$ then $y_{j+1} = 1$.

Consider that k zeros occur after $y_{n+1} = 1$. Then, from the lower bound of $w[n+1]$ we obtain

$$w[n+k] = 2^{k-1}w[n+1] \geq 2^{k+n+\nu_{n+1}}$$

Since the value y_{n+1+k} is still zero, we must have

$$2w[n+k] < C[n+k]$$

which implies

$$2^{k+n+\nu_{n+1}+1} < C[n+k] \tag{9}$$

Hence,

$$k < \log_2 C[n+k] - n - 1 - \nu_{n+1}$$

resulting in

$$k \leq \lceil \log_2 C[n+k] \rceil - n - 2 - \nu_{n+1} \tag{10}$$

where

$$\nu_{n+1} = \min_{p,q}(\alpha_{p,q} - np - (n+1)q)$$

This expression for k is not useful because of the k in the right-hand side. To avoid this we replace $C[n+k+1]$ by its maximum value for $k > 0$. As we will see in the applications, in the practical cases (where n is large) this can be replaced by the maximum of the first partial derivative.

For our example

$$C[j] = 5y[j]^4 + 10y[j]^3 2^{-(j+1)} + \ldots$$

so that

$$\max(C[j]) = 5$$

Hence the bound is

$$k \leq 3 - n - 2 + 5(n+1) = 3 + 4n + 3 \leq 4n + 6$$

2.2 General ranges

Up to now we have assumed that $1/2 \leq x < 1$ and $1/2 \leq y < 1$. For the general case, we split the domain into subdomains such that $2^i \leq x < 2^{i+1}$ and $2^j \leq y < 2^{j+1}$. Then for each subdomain we apply the method with $X = 2^{-i-1}x$ and $Y = 2^{-j-1}y$. If $P(x,y) = 0$, we then have $P(2^{i+1}X, 2^{j+1}Y) = 0$. This $P(2^{i+1}X, 2^{j+1}Y)$, ONCE DIVIDED by the right power of 2 so that it has integer coefficients that are as small as possible is a new polynomial $Q(X,Y)$ that is used for the method.

For instance, consider the reciprocal function for $1/2 \leq x < 1$. We have $i = -1$ (so $X = x$) and $j = 0$ (so $Y = 2^{-1}y$). Since $P(x,y) = 1 - yx$, we get

$$P(X, 2Y) = 1 - 2YX$$

$$A_{11} = -2,\ \alpha_{11} = 1;\quad A_{00} = 1,\ \alpha_{00} = 0$$

$$\nu_{n+1} = 1 - n - (n+1) = -2n \quad (w[n+1] \geq 2^{-(n-1)})$$

$$C[j] = 2X \quad \max(C[j]) = 2$$

$$k \leq \log(2) - n - 2 + 2n = n - 1$$

2.3 Non-integer coefficients

The bounds on k given in the previous sections were derived assuming P has integer coefficients. In practice, they also hold when the coefficients of P are all multiple of a negative integer power of 2, say $2^{-\mu}$, with μ integer. To prove this, consider the polynomial $Q = 2^{\mu}P$, which has integer coefficients. Directly applying the formulas to Q instead of P will subtract μ from ν_{n+1} and subtract μ from $\log_2(C[n+k])$, so that the obtained value of k is exactly the same. As a consequence, the method can be applied whenever the coefficients of P have a finite binary representation.

2.4 Application to some usual functions

2.4.1 Reciprocal

This is illustrated in Section 2.2, resulting in

$$k \leq n - 1 \tag{11}$$

2.4.2 Division

For division $y = a/x$ and $1/2 \leq a, x < 1$, we have to consider two cases:

- $a < x$. In this case $1/2 \leq y \leq 1$ so

$$P(x,y) = a - yx$$

$$A_{11} = -1,\ \alpha_{11} = 0;\quad A_{00} = a,\ \alpha_{00} = -1$$

$$\nu_{n+1} = 0 - n - (n+1) = -2n - 1$$

$$C[j] = x \quad \max(C[j]) = 1$$

$$k \leq 0 - n - 2 + 2n + 1 = n - 1 \tag{12}$$

- $a > x$. In this case $1 \leq y < 2$. So, $Y = 2^{-1}y$ and we have the same bound as for the reciprocal.

2.4.3 Square root

$$P(x,y) = x - y^2$$

$$A_{02} = -1,\ \alpha_{02} = 0;\quad A_{10} = 1,\ \alpha_{10} = 0$$

$$\nu_{n+1} = -2(n+1)$$

$$C[j] = 2y[j] + 2^{-(j+1)} \quad \max(C[j]) = 2$$

$$k \leq 1 - n - 2 + 2(n+1) = n + 1 \tag{13}$$

2.4.4 q root ($q \geq 2$, integer)

$$P(x,y) = x - y^q$$

$$A_{0q} = -1,\ \alpha_{0q} = 0;\quad A_{10} = 1,\ \alpha_{10} = 0$$

$$\nu_{n+1} = -(n+1)q$$

$$C[j] = qy[j] + \ldots \quad \max(C[j]) = q$$

$$k \leq \lceil \log_2 q \rceil - n - 2 + (n+1)q = \lceil \log_2 q \rceil + (n+1)(q-1) - 1 \tag{14}$$

2.4.5 Reciprocal square roots

To obtain results that are valid for any floating-point input, we must find results for x belonging to two consecutive binades. Hence, we consider two cases:

- $y = 1/(2\sqrt{x})$, with $x, y \in [1/2, 1)$. This corresponds to the case of floating-point inputs in intervals of the form $[2^{2i-1}, 2^{2i}]$. We get

$$P(x,y) = 1 - 4xy^2$$

$$A_{00} = 1\ \ \alpha_{00} = 0;\quad A_{12} = -4,\ \alpha_{12} = 2$$

$$\nu_{n+1} = \min_{p,q} \{\alpha_{p,q} - np - (n+1)q\} = -3n.$$

$$C[j] = 8xy[j] + \ldots \quad \max(C[j]) = 8$$

$$k \leq \log_2(8) - n - 2 + 3n = 2n + 1 \tag{15}$$

- $y = 1/\sqrt{2x}$, with $x, y \in [1/2, 1)$. This corresponds to the case of floating-point inputs in intervals of the form $[2^{2i}, 2^{2i+1}]$. We get

$$P(x,y) = 1 - 2xy^2$$

and we deduce the same bound as in the previous case.

2.4.6 Norms

Since $P(x,y) = x^2 + v^2 - y^2$, the bound is the same as for square root, namely,

$$k \leq n+1 \tag{16}$$

2.4.7 Normalization of 2-D vectors

The function to compute is $y = x/\sqrt{x^2+v^2}$ for $1/2 \leq x, v < 1$. We consider the case $1/2 \leq y < 1$, which requires $3x^2 \geq v^2$. The algebraic function is

$$P(x,y) = x^2 - (x^2+v^2)y^2$$

Then

$$A_{20} = |A_{22}| = 1, \quad \alpha_{20} = \alpha_{22} = 0$$

$$|A_{02}| = v^2 \quad \alpha_{02} = 2\log_2 v$$

$$\nu_{n+1} = -4n-2$$

$$C[j] = 2(x^2+v^2)y \quad \max(C[j]) \leq 4$$

$$k \leq 2-n-2+4n+2 = 3n+2 \tag{17}$$

2.5 Longest run of ones

A run of k ones following a zero in $y = f(x)$ corresponds to a run of zeros following a one in function $z = 1 - f(x)$. Hence, it suffices to apply the previous results to the function $z = 1 - f(x)$ to obtain the longest run of ones in $f(x)$. To achieve the range of $1/2 \leq z < 1$ we use $3/2 - f(x)$.

3 Actual maxima for some functions

In the previous section, we have obtained bounds on k. We now explore whether these bounds are attained.

3.1 Reciprocal

For this function it is easy to find a case that attains the bound for the run of zeros. Namely, for $1/2 \leq x, y < 1$ if $x = 1-2^{-n}$ we get $y = 1/x = 1+2^{-n}+2^{-2n}+\ldots$. Since this is larger than 1, it is necessary to normalize resulting in a 1 in position $n+1$ and the next 1 in position $2n+1$. Consequently, there are $k = n-1$ consecutive zeros.

For the run of ones it can be shown that it is attained for n odd. Namely, for $x = 1 - 2^{-(n-1)/2} + 2^{-n}$ results in $y = 1 + 2^{-(n-1)/2} + 2^{-n} - 2^{-2n}\ldots$ This can be easily shown by defining $t = 2^{-(n-1)/2}$ and obtaining the Taylor expansion of $1/(1-t+t^2/2)$.

For n even[5], an analysis similar to the one performed for square root (see Section 3.3) and applied to the algebraic function $P(x,y) = 3x - 1 - xy$, results in

$$e = 2^{2n} - b(3 \times 2^n - (2a+1))$$

. Consequently, the lower bound of $e = 1$ is achieved for b a factor of $2^{2n}+1$ with $2^{n-1} \leq b < 2^n$. By exhaustive search, the bound is achieved for $n = 24$.

3.2 Division

For the run of zeros the same case as for reciprocal applies. For the run of ones consider the case $q = a/b$ with $A = 2^n a = 3 \times 2^{n-2} + 2$ and $B = 2^n b = 2^{n-1} + 1$. Then, defining $t = 1/(2^{n-1})$, the quotient A/b is $f(t) = (3+2t)/(1+t)$ with Taylor expansion $3-t+t^2-t^3+t^4-\ldots$. Hence $A/B = 3 - 2^{-n+1} + 2^{-2n+2} - \ldots$. So its first bits are: 1.100000..0 (n bits) 0 (round bit) 1111111...1 ($n-1$ ones) 000..

3.3 Square root

Here we show that the actual maximum number of consecutive zeros after $y_{n+1} = 1$ is $k = n-1$. We first tighten the bound and then show the existence of a case that satisfies this bound.

From the bound obtained in the previous section ($w[n+1] \geq 2^{-(n+1)}$) we get

$$w[n+1] = e2^{-(n+1)}, \ e \ integer \tag{18}$$

Moreover, since $y_{n+1} = 1$ (rounding bit), we obtain from the recurrence

$$w[n+1] = 2w[n] - 2y[n] - 2^{-(n+1)} \tag{19}$$

Replacing $w[n]$ by its definition ($w[n] = 2^n(x - y[n]^2)$) results in

$$2^{n+1}(x - y[n]^2) - 2y[n] - 2^{-(n+1)} = e2^{-(n+1)} \tag{20}$$

Calling $(2^n)x = b$ (integer) and $(2^n)y[n] = a$ (integer) we get

$$e+1 = 4(2^n b - a^2 - a) = 4(2^n b - a(a+1)) \tag{21}$$

Since $2^n b$ and $a(a+1)$ are even we get

$$e+1 \geq 8$$

resulting in

$$e \geq 7 \tag{22}$$

[5]This was pointed out to us by one reviewer

That is, the minimum residual is bounded by $7 \times 2^{-(n+1)}$ and consequently $k \leq n-1$.

Now we need to show that this bound is achieved. Making $e = 7$ in (21) we get

$$(2^n)b - a(a+1) = 2$$

with $2^{n-1} \leq a < 2^n$ and $2^{n-2} \leq b < 2^n$

We now need to show that this equation has an integer solution in the range of a and b. It is possible to prove by induction the following solution recurrence:

$$a(2) = 2 \quad b(2) = 2$$

If $b(i)$ is odd THEN $a(i+1) = 2^i + a(i)$

If $b(i)$ even THEN $a(i+1) = 2^{i+1} - a(i) - 1$

Moreover, $b(i+1) = [a(i+1)(a(i+1)+1)+2]2^{-(i+1)}$

Now we can prove by induction that these values are inside the range:

If $2^{i-1} \leq a(i) < 2^i$ then

for $b(i)$ odd $2^i + 2^{i-1} \leq a(i+1) = 2^i + a(i) < 2^{i+1}$ (inside the range)

for $b(i)$ even $2^{i+1} - 2^i - 1 \leq a(i+1) = 2^{i+1} - a(i) - 1 < 2^{i+1} - 2^{i-1} - 1$ (inside the range).

Similarly, we can show that $b(i)$ is inside the required range. As a consequence, there is always a solution inside the range and the bound is achieved.

This method can also be used to obtain an example, even for large precision. For instance for quad precision ($n = 113$) the values are (in hex)

a = '10720461FD6E2F325A24E31B39FA5'

b = '8739AA138358BAE6FC9A76A058E5'

(obtained very fast with Maple).

Finally, for the run of ones two simple cases that achieve $k = n+1$ are sqrt(0.11111...11111) and sqrt(1.0000...000001).

3.4 Cube root

The case of cube root seems quite different from square roots. We have not found a "general formula" for the actual bound. For small values of n, we have performed an exhaustive search, the result of which is given in Table 1.

Table 1. Bounds for cube roots for $4 \leq n \leq 20$

n	x (binary)	$x^{1/3}$	k
4	0.01001	$0.1010011110\cdots$	4
5	0.0011101	$0.100111000001\cdots$	5
6	0.0101110	$0.1011011000000001\cdots$	8
7	0.01001111	$0.1010110011111110\cdots$	7
8	0.10011101	$0.11011001100000000001\cdots$	10
9	0.00100010100	$0.1000001101000000000001\cdots$	11
10	0.01000000001	$0.10100001010111111110\cdots$	9
11	0.010101010001	$0.1011000101010000000000000001\cdots$	15
12	0.00111010010100	$0.100111000101100000000000001\cdots$	13
13	0.01100010100111	$0.1011101001000011111111111110\cdots$	14
14	0.011101011000001	$0.1100010101111001111111111110\cdots$	13
15	0.0100100101111001	$0.1010100011011100 1^{17} 0\cdots$	17
16	0.0010011011011110000	$0.100010001000111110^{21} 1\cdots$	21
17	0.10000011001001101	$0.1100110011010111010^{19} 1\cdots$	19
18	0.010101111110000011	$0.101100110011111101 01^{19} 0\cdots$	19
19	0.0111111010001000011	$0.110010100110100001110^{22} 1\cdots$	22
20	0.01101101111011101001	$0.1100000100100011010101^{23} 0\cdots$	23

3.5 Reciprocal square root

As for cube roots, we have not found a general formula for the maximum value. Using an analysis similar to that of square root we obtain that the minimum residual $e = w[n+1]2^{2n+1}$ is given by the expressions

$$e = 2^{3n+2} - 2b(2a+1)^2$$

and

$$e = 2^{3n+2} - 4b(2a+1)^2$$

for the two binades to consider. Using these expressions, by exhaustive search, we have found the actual bounds for n from 4 to 20, and for $n = 24, 32$ and 53 (see Table 2). For $n \leq 20$, we have used a Maple program and for the larger values of n, we have used a dozen of workstations in parallel during a few days, running a program implementing the algorithm presented in [3][6] Note that there are many values of n (at least, $n = 5, 7, 9, 10, 11, 12, 13$ and 18) for which the bound is obtained for $0.111111\ldots110 \times 2^p$, where p is an even exponent (in Table 2, this corresponds to the values of the form $11.1111\ldots110$).

3.6 Normalization of 2-D vectors

Table 3 shows the actual values of k for n from 3 to 10. Although the bound of $3n+2$ is far from achieved, there are

[6]We thank Vincent Lefèvre for the help with this.

Table 2. Bounds for reciprocal square roots for $4 \leq n \leq 20$ and for $n = 24, 32$ and 53 and $1 \leq x < 4$ (which suffices to deduce all cases).

n	x (binary)	$1/\sqrt{x}$	k
4	1.101	$0.110010001\cdots$	3
5	11.110	$0.10000100001\cdots$	4
6	11.0100	$0.10001110000000001\cdots$	9
7	11.11110	$0.100000010000001\cdots$	6
8	10.011011	$0.1010010001111111110\cdots$	9
9	11.1111110	$0.1000000001000000001\cdots$	8
10	11.11111110	$0.100000000010000000001\cdots$	9
11	11.111111110	$0.10000000000001000000000001\cdots$	10
12	11.1111111110	$0.10000000000001000000000001\cdots$	11
13	11.11111111110	$0.1000000000000010^{12}1\cdots$	12
14	1.0010001110011	$0.1110111111011101^{14}0\cdots$	14
15	11.1000101000110	$0.1000100000010000 1^{16}0\cdots$	16
16	10.1111110001000	$0.1001001111111011 10^{15}1\cdots$	15
17	1.0111111000101100	$0.110100011000010111 0^{19}1\cdots$	19
18	11.1111111111111110	$0.100000000000000000010^{17}1\cdots$	17
19	1.100010000011110011	$0.110011101101000100010^{23}1\cdots$	23
20	1.0000101100011111101	$0.1111101010011100111110^{20}1\cdots$	20
24	10.1110100001100011 100011	$0.1001011000100000100 1111 001^{27}0\cdots$	27
32	1.0001111000001101100 01011 0101101	$0.1111001000101101110111010 100101010^{32}1\cdots$	32
53	1.1010011010101001110011 0000 0101011010101111001100 1110	$0.110001110011101111101000010001 1000101000110000100101010 1^{57}0\cdots$	57

Table 3. Bounds for normalization of 2D-vectors (x, v) (that is, $x/\sqrt{x^2+v^2}$) for $3 \leq n \leq 10$, with $1 \leq x, v < 2$.

n	x (binary)	v (binary)	$x/\sqrt{x^2+v^2}$	k
3	0.111	0.101	$0.1101000001\cdots$	5
4	0.1010	0.1000	$0.110001111110\cdots$	6
5	0.11001	0.10101	$0.11000100000001\cdots$	7
6	0.100000	0.110010	$0.10001001^{9}0\cdots$	9
7	0.1111011	0.1011100	$0.110011001^{12}0\cdots$	12
8	0.10010110	0.10101000	$0.1010101010^{13}1\cdots$	13
9	0.110010001	0.111011000	$0.10100101101^{16}0\cdots$	16
10	0.1101010000	0.1111111111	$0.101000110101^{17}0\cdots$	17

some values which are significantly larger than n, which is the value expected by probabilistic considerations.

4 Comparison with results given in [1] and conclusions

Table 4 shows a comparison with the results reported in [1]. As can be seen the specific results are essentially the same, with minor differences for reciprocal, square root, and reciprocal square root. As indicated, we provide expressions for the bound for the class of algebraic functions.

The main issue that is still pending is to obtain better bounds for those cases, such as reciprocal square root, for which the bounds are far from being achieved for specific values of n and show whether they are achieved for some n.

In Section 1.3 we have given a radix-2 restoring digit-recurrence algorithm for algebraic functions. As is well known for division and square root, this can be extended by the use of a redundant digit set for y_j (usually signed-digit) to simplify the selection function. Then, the arguments of the selection function are low precision estimates of the residual and of $C[j]$. The feasibility of the implementation depends on the complexity of the selection function and of the computation of $C[j]$.

Acknowledgment. Special thanks to the reviewer that did an extraordinary job. He/she checked every detail finding a few mistakes, gave us suggestions on presentation, and even contributed some additional results. As Jean-Michel said: "what a reviewer!".

References

[1] C. S. Iordache and D. W. Matula, *Infinitely Precise Rounding for Division, Square root, and Square Root Reciprocal*, Proc. 14th IEEE Symp. on Computer Arithmetic, 1999, pp. 233-240.

[2] V. Lefevre, J-M. Muller, and A. Tisserand, *Towards Correctly Rounded Transcendentals*, Proc. 13th IEEE Symp. on Computer Arithmetic. 1997, pp. 132-137.

[3] V. Lefevre and J-M. Muller, *Worst Case for Correct Rounding of the Elementary Functions in Double Precision*, submitted to ARITH15.

Table 4. Comparison with results given in [1]

Function	run of ones		run of zeros	
	I-M	Our	I-M	Our
Reciprocal	$\leq n-1$	$= n-1$(all n odd) $= n-1$(some n even)	$= n-1$	$= n-1$
Division	$= n-1$	$= n-1$	$= n-1$	$= n-1$
Square root	$= n+1$	$n+1$	$\leq n-1$	$= n-1$
Rec. square root	$\leq 2n-1$	$\leq 2n+1$	$\leq 2n-1$	$\leq 2n+1$
Norm	–	$\leq n+1$	–	$\leq n+1$
q-root	–	$\leq \lceil \log_2 q \rceil + (n+1)(q-1) - 1$	–	(same)
2D normalization	–	$\leq 3n+2$	–	$\leq 3n+2$
algebraic	–	$\leq \lceil \log_2(max(C[j])) \rceil - n - 2 - \nu_{n+1}$	–	(same)

[4] P. W. Markstein, *Computation of elementary Functions on the IBM RISC System/6000 Processor*, IBM J. Res. Develop., vol. 34, no. 1, Jan. 1990, pp. 111-119.

[5] M. Schulte and E.E. Swartzlander, *Exact Rounding of Certain Elementary Functions*, 11th Symp. on computer Arithmetic, 1993, pp. 138-145.