Radix-10 BKM Algorithm for Computing Transcendentals on Pocket Computers

L. Imbert
J.M. Muller
F. Rico

September 1999

Research Report № 99-43
Radix-10 BKM Algorithm for Computing Transcendentals on Pocket Computers

L. Imbert
J.M. Muller
F. Rico
September 1999

Abstract
We present a radix-10 variant of the BKM algorithm. It is a shift-and-add, CORDIC-like algorithm that allows fast computation of complex exponentials and logarithms. This radix-10 version is suitable for implementation in a pocket computer.

Keywords: Elementary functions, CORDIC algorithms, Computer Arithmetic, radix-10 arithmetic

Résumé
Nous proposons une variante de l'algorithme BKM adaptée au calcul en base 10. C'est un algorithme à additions et décalages, qui permet d'évaluer rapidement des exponentielles et logarithmes complexes. Cette version adaptée à la base 10 est destinée à l'implantation sur des calculatrices de poche.

Mots-clés: Fonctions élémentaires, algorithmes CORDIC, Arithmétique des ordinateurs, arithmétique base 10
Radix-10 BKM Algorithm for Computing Transcendentals on Pocket Computers

L. Imbert
Laboratoire d’Informatique de Marseille

J.M. Muller
CNRS-LIP, projet CNRS-ENSL-INRIA Arenaire

F. Rico
Laboratoire d’Informatique de Marseille

September 1999

1 Introduction

Our goal is to design fast shift-and-add algorithms for implementing elementary functions of a complex variable on pocket calculators. As a matter of fact, the most well known shift-and-add algorithm, namely the CORDIC algorithm [10, 11], has been the first algorithm used for evaluating transcendentals on a pocket computer. Recent surveys can be found in [8, 7]. Several authors recently suggested redundant [1] versions of CORDIC, to get faster iterations [9, 5, 4, 6].

In this paper, we aim at designing a radix-10, redundant version of the BKM algorithm. The BKM algorithm was introduced by Bajard, Kla and Muller in 1993 [3]. It is a CORDIC-like algorithm that allows the computation of complex logarithms and exponentials (and therefore of most real elementary functions), without scale factor (even when using redundant arithmetic). Bajard and Imbert have recently suggested some variants of this algorithm [2].

Whereas the original algorithm was suited for radix 2 arithmetic, if we want to design very efficient implementations for pocket computers, we need a radix-10 variant of the algorithm, since most, if not all, pocket computers use a radix-10 arithmetic.

Our algorithm requires a special first step whose purpose is to reduce the initial argument to some value that is small enough, so that the subsequent iterations will converge. After that, the BKM iterations are applied:

\[
\begin{align*}
E_{n+1} &= E_n(1 + d_n10^{-n}) \\
L_{n+1} &= L_n - \ln(1 + d_n10^{-n})
\end{align*}
\]

where \(d_n\) is such that a multiplication by \(d_n\) is straightforwardly performed in radix-10 arithmetic.
1.1 Notations

- For $x \in \mathbb{C}$, we note $\langle x \rangle_p$, the values of the real and imaginary parts of $x$ rounded to the nearest number with $p$ fractional digits. For example, we note $\langle x \rangle_0$ the point in $\mathbb{Z} + i\mathbb{Z}$ closest to $x$.

- For $x \in \mathbb{C}$, we note $[x]_p$, the values of the real and imaginary parts of $x$ truncated down with $p$ fractional digits.

1.2 The BKM algorithm

From the initial values $L_1$ and $E_1$, we perform in parallel the following 2 iterations:

$$
\begin{align*}
L_{n+1} &= L_n - \ln(1 + d_n 10^{-n}) \\
E_{n+1} &= E_n(1 + d_n 10^{-n})
\end{align*}
$$

with:

$$
d_n = d_n^r + i d_n^\theta
$$

$$
-6 \leq d_n^r, d_n^\theta \leq 6
$$

The value $d_n$ depends on which function we wish to calculate. The BKM algorithm has 2 modes:

- **E-mode**: if we find a sequence $d_n$ such that $L_n$ goes to 0, then we will obtain $E_n \rightarrow E_1 e^{L_1}$.

- **L-mode**: if we find a sequence $d_n$ such that $E_n$ goes to 1, then we will obtain $L_n \rightarrow L_1 + \ln(E_1)$.

2 The complex exponential function (E-mode)

In this section, we will focus on the problem of finding a sequence $d_n$ such that $L_n$ goes to 0.

If we consider the iteration $L_{n+1} = L_n - \ln(1 + d_n 10^{-n})$, the choice that immediately springs in mind for $d_n$ (but of course it is not implementable) is $d_n = \langle (e^{L_n} - 1)10^n \rangle_0$.

Let us define $T_n = 10^n L_n$. We will first assume that $L_n$ is very small compared to 1. We will later see how to reduce the general case to this one. Our previous “ideal” choice for $d_n$ is equivalent to choosing $d_n = \langle T_n \rangle_0$.

In the following, we suppose that $L_n \ll 1$. We describe one iteration of the algorithm and prove the convergence for $n \geq 3$ in the domain: $L_3 \in [-0.0065, 0.0065] + \hat{v}[-0.0065, 0.0065]$.

2.1 One step of the algorithm

We give an algorithm for which we will be able to show by induction that the real and imaginary parts of $T_n$ have absolute value less than 6 for any $n$.  

2
We suppose that, after \( n \) iterations:

\[
-6 < T^n_x, T^n_y < 6
\]

We will prove that: \(-6 < T^{n+1}_x, T^{n+1}_y < 6\).

The iteration \( L_{n+1} = L_n - \ln(1 + d_n 10^{-n}) \) becomes:

\[
L_{n+1}^x = L_n^x - \frac{1}{2} \ln \left( 1 + 2d_n^x 10^{-n} + (d_n^{x^2} + d_n^{y^2}) 10^{-2n} \right)
\]

\[
L_{n+1}^y = L_n^y - d_n^y \arctan \left( \frac{10^{-n}}{1 + d_n 10^{-n}} \right)
\]

If we consider the Taylor series for \( \ln(1 + x) \), and \( \arctan(x) \), we obtain:

\[
\frac{1}{2} \ln \left( 1 + 2d_n^x 10^{-n} + (d_n^{x^2} + d_n^{y^2}) 10^{-2n} \right) \simeq d_n^x 10^{-n} - d_n^y \arctan \left( \frac{10^{-n}}{1 + d_n 10^{-n}} \right) \simeq d_n^y 10^{-n}
\]

So we need \( d_n^x \) to be very close to \( T^n_x \) and \( d_n^y \) close to \( T^n_y \). We can choose \( d_n^x = \langle |T^n_x|, 2 \rangle \) and \( d_n^y = \langle |T^n_y|, 2 \rangle \). That means that we choose as \( d_n^x \) (resp. \( d_n^y \)) the digit closest to \( T^n_x \) (resp. \( T^n_y \)) considering the first 3 digits of \( T^n_x \) (resp. \( T^n_y \)) only. Note that \(-6 < d_n^x, d_n^y \leq 6\).

Then

\[
|\langle |T^n_x|, 2 \rangle - T^n_x|, |\langle |T^n_y|, 2 \rangle - T^n_y| \leq \frac{1}{100}
\]

and

\[
|d_n^x - \langle |T^n_x|, 2 \rangle|, |d_n^y - \langle |T^n_y|, 2 \rangle| \leq \frac{5}{100}
\]

So

\[
|d_n^x - T^n_x|, |d_n^y - T^n_y| \leq \frac{51}{100}
\]

Let us define:

\[
A_n = \frac{1}{2} \ln \left( 1 + 2d_n^x 10^{-n} + (d_n^{x^2} + d_n^{y^2}) 10^{-2n} \right)
\]

\[
B_n = d_n^y \arctan \left( \frac{10^{-n}}{1 + d_n 10^{-n}} \right)
\]

then

\[
|T^n_{x+1}| \leq 5,1 + (d_n^x 10^{-n} - A_n) 10^{n+1}
\]

\[
|T^n_{y+1}| \leq 5,1 + (d_n^y 10^{-n} - B_n) 10^{n+1}
\]

So, we need:

\[
|d_n^x 10^{-n} - A_n| 10^{n+1} \leq 0.9
\]

\[
|d_n^y 10^{-n} - B_n| 10^{n+1} \leq 0.9
\]
In order to prove (2), let us define: \( x = 2d_n^x10^{-n} + (d_n^{x^2} + d_n^{x^2})10^{-2n} \).

If \( n \geq 3 \), we have: \( |x| \leq \frac{1}{d_n^x} \).

We define: \( \alpha = 75\ln \left( \frac{d_n^x}{d_n^{x^2}} \right) \), and \( \beta = 75 \ln \left( \frac{d_n^{x^2}}{d_n^{x^2}} \right) \).

We know that (if \( x \geq 0 \)): \[
\alpha x \leq \ln(1 + x) \leq x
\]
Thus:
\[
d_n^x(\alpha - 1)10^{-n} + \frac{\beta}{2}(d_n^{x^2} + d_n^{x^2})10^{-2n} \leq A_n - d_n^x10^{-n} \leq \frac{\beta}{2}(d_n^{x^2} + d_n^{x^2})10^{-2n}
\]

If \( n \geq 3 \) and \( |d_n^x|, |d_n^{x^2}| \leq 6 \), as \( \alpha \geq 0.993 \) we have
\[
|A_n - d_n^x10^{-n}|10^{n+1} \leq 0.9
\]
similarly if \( x \leq 0 \),
\[
\beta x \leq \ln(1 + x) \leq x
\]
and as \( \beta \leq 1.007 \)
\[
|A_n - d_n^x10^{-n}|10^{n+1} \leq 0.9
\]
for (3), we can say that (if \( x \geq 0 \)):
\[
x - \frac{x^3}{3} \leq \arctan(x) \leq x
\]
and then, if \( n \geq 3 \):
\[
d_n^x\frac{10^{-n}}{1 + d_n^x10^{-n}} - \frac{d_n^{x^2}}{3} \left( \frac{10^{-n}}{1 + d_n^{x^2}10^{-n}} \right)^3 \leq B_n \leq d_n^{x^2}\frac{10^{-n}}{1 + d_n^{x^2}10^{-n}}
\]
As \( |d_n^x|, |d_n^{x^2}| \leq 6 \),
\[
|B_n - d_n^x10^{-n}| \leq 0.037 \times 10^{-n}.
\]
If \( x \leq 0 \) we have
\[
x \leq \arctan(x) \leq x - \frac{x^3}{3}
\]
and the same result.

So the domain of convergence of the algorithm is
\[
P = [-0.0065, 0.0065] + i[-0.0065, 0.0065]
\]
we can extend this domain to
\[
P = [-0.025, 0.035] + i[-0.035, 0.035]
\]
since if we reduce the number of values for \( d_n^x \), and \( d_n^{x^2} \), we can obtain \( L_2 \in P \). We have plotted the surfaces corresponding to \( z = |d_n^x10^{-2} - A_2|10^3 \), \( z = |d_n^{x^2}10^{-2} - B_2|10^3 \), and the plane \( z = 0.9 \), for \( -2 \leq d_n^x \leq 3 \), and \( -3 \leq d_n^{x^2} \leq 3 \).

We can see in figure 1 that:
\[
|d_n^x10^{-2} - A_2|10^3 \leq 0.9
\]
\[
|d_n^{x^2}10^{-2} - B_2|10^3 \leq 0.9
\]

Unfortunately, this allow the computation of \( \exp(x) \) in a tiny domain only. In the next section, we will give a method to reduce \( x \) to the convergence domain.
2.2 Range reduction for the exponential

We assume that we wish to evaluate \( \exp(x) \), where \( x = a + ib \), and \( -\frac{1}{2} \leq a, b \leq \frac{1}{2} \). Reduction to this domain is fairly straightforward.

Define \( b' \) as the number obtained by truncating \( b \) to two fractional digits. From \( b' \), we look up in a table the number of the form \( \beta_k = \log(1 + ik_0 \cdot 10^{-2}) \), with \( 55 \leq k_0 \leq 55 \) that is closest to \( b' \). The number \( y = x - \log(1 + ik_0 \cdot 10^{-2}) \) has an imaginary part of absolute value less than \( \frac{1}{100} + \frac{1}{2} \max |\Im(\beta_{k+1} - \beta_k)| = 0.01 + 0.0049998333433326191032 \).

The real part of \( y \) is between \(-\frac{1}{2} - 0.133\) and \(-\frac{1}{2}\). Define \( a' \) as \( \Re(y) \) truncated to two fractional digits. From \( a' \) we look up in a table the number of the form \( \alpha_k = \log(1 + ik_0 \cdot 10^{-2}) \), with \( -47 \leq k_0 \leq 65 \) that is closest to \( a' \). The number \( z = y - \log(1 + ik_0 \cdot 10^{-2}) \) has the same imaginary part as \( y \), and its real part has an absolute value less than \( \frac{1}{100} + \frac{1}{2} \max |\alpha_{k+1} - \alpha_k| = 0.01 + 0.00934606656067627473 \).

We will choose \( z \) as the reduced argument. Its real part has an absolute value less than 0.02, and its imaginary part has an absolute value less than 0.015.

3 The complex logarithm function (L-mode)

In this section, we will focus on the problem of finding a sequence \( d_n \) such that \( E_n \) goes to 1.

If we note: \( S_n = 10^n(E_n - 1) \), the choice that immediately springs in mind for \( d_n \) (but
of course it is not implementable) is \( d_n = -\lfloor \frac{S_n}{E_n} \rfloor \). ¹ It is clear, that if \( E_n \approx 1 \), this is equivalent to choosing \( d_n = -\lfloor S_n \rfloor \).

In the following, we suppose \( E_n \approx 1 \). We describe one iteration of the algorithm and prove the convergence for \( n \geq 2 \). In section 3.2, we will explain the reduction step that gives \( E_n \approx 1 \).

### 3.1 One step of the algorithm

We suppose that, after \( n \) iterations:

\[ -6 < S_n^e, S_n^y < 6 \]  

(4)

The iteration \( E_{n+1} = E_n(1 + d_n 10^{-n}) \) becomes:

\[
\begin{align*}
S_{n+1}^e &= 10 \left( S_n^e + d_n^e + (d_n^e S_n^e - d_n^y S_n^y) 10^{-n} \right) \\
S_{n+1}^y &= 10 \left( S_n^y + d_n^y + (d_n^e S_n^e + d_n^y S_n^y) 10^{-n} \right)
\end{align*}
\]

so we need \( d_n^e \) to be very close to \( -S_n^e \) and \( d_n^y \) to be very close to \( -S_n^y \). We can choose \( d_n^e = -\lfloor |S_n^e| 2 \rfloor \) and \( d_n^y = -\lfloor |S_n^y| 2 \rfloor \). That means that we choose as \( d_n^e \) (resp. \( d_n^y \)) the digit closest to \( -S_n^e \) (resp. \( -S_n^y \)) considering only the first 3 digits of \( S_n^e \) (resp. \( S_n^y \)).

Then

\[
|\lfloor S_n^e \rfloor - S_n^e|, |\lfloor S_n^y \rfloor - S_n^y| \leq \frac{1}{100}
\]

and

\[
|d_n^e + |S_n^e|2|, |d_n^y + |S_n^y|2| \leq \frac{5}{100}
\]

So

\[
|d_n^e + S_n^e|, |d_n^y + S_n^y| \leq \frac{51}{100}
\]

and then

\[
\begin{align*}
|S_{n+1}^e| &\leq 10 \left( \frac{51}{100} + (|d_n^e S_n^e| + |d_n^y S_n^y|) 10^{-n} \right) \\
&\leq 5,1 + (|d_n^e S_n^e| + |d_n^y S_n^y|) 10^{-n+1} \\
|S_{n+1}^y| &\leq 5,1 + (|d_n^e S_n^e| + |d_n^y S_n^y|) 10^{-n+1}
\end{align*}
\]

If \( n \geq 3 \) as \( |S_n^e|, |S_n^y| < 6 \) and \( |d_n^e|, |d_n^y| < 6 \),

\[
10^{-n+1} (|d_n^e S_n^e| + |d_n^y S_n^y|) \leq 0.72 < 0.9
\]

\[
10^{-n+1} (|d_n^e S_n^e| + |d_n^y S_n^y|) \leq 0.72 < 0.9
\]

If \( n = 2 \) and if \( |S_2^e|, |S_2^y| < 2 \) then, \( |d_2^e|, |d_2^y| < 2 \), and

\[
10^{-1} (|d_2^e S_2^e| + |d_2^y S_2^y|) \leq 0.8 < 0.9
\]

\[
10^{-1} (|d_2^e S_2^e| + |d_2^y S_2^y|) \leq 0.8 < 0.9
\]

We obtain:

¹Actually, \( E_n(1 - \frac{S_n}{E_n} 10^{-n}) = 1 \)
• if \( S_2 \in [-2, 2] + i[-2, 2] \) then \( \| S_2 \|_{\infty} < 6 \),
• and for \( n \geq 3 \), by induction \( \| S_n \|_{\infty} < 6 \).

As a result, we have an algorithm which converge to \( \ln(x) \) for
\[
x \in P = [0.98, 1.02] + i[-0.02, 0.02]
\]

Unfortunately, this allows the computation of \( \ln(x) \) in a tiny domain only. In order to allow the computation for more values, we need to reduce the argument.

### 3.2 Range reduction for the logarithm

Using a method presented in [2], we can reduce the argument to
\[
S = \{ x + iy; \ x > 0 \text{ and } -\frac{2}{5} x \leq y \leq \frac{2}{5} x \}
\]

From \( S \), it is easy to reduce to \( T \) (one shift and one multiplication by 2)
\[
T = \{ x + iy; \ 0.98 \leq x \leq 2 \text{ and } -\frac{2}{5} x \leq y \leq \frac{2}{5} x \}
\]

So, our goal is to find a reduction from \( T \) to \( P \).

In the following, we will explain this step in greater detail.

• start with \( E_1 \in T \)
• choose \( d = \langle |E_1|_3 \rangle_1 \)
• read in a table \( z = \left( \frac{1}{1+i\theta} \right)_2 \)
• compute \( \hat{z} = E_1 \times z \)
• choose \( d' = \langle |\hat{z}|_4 \rangle_2 \)
• read in a table \( z' = \left( \frac{1}{1+i\theta'} \right)_2 \)
• compute \( E_2 = \hat{z} \times z' \ (E_2 \in P) \)

**Proof of the reduction**

First, we define \( R \), a rectangle such that \( T \subset R \). We split \( R \) into 187 squares depending of the different choices for \( d \) (see figure 2(a)). Then, we have computed the image of \( R \) by the first step of the argument reduction (that is to say the image of each square by a similitude). We can see on figure 2(b) that this image is enclosed into the square:
\[
\hat{R} = [0.925, 1.075] + i[-0.075, 0.075].
\]

The second step is similar. We split \( \hat{R} \) into 225 squares, and compute by the same way, the image of \( \hat{R} \) (see figure 2(c), and 2(d)).

Consequently, all the computed points \( (E_2) \) are enclosed into \( P \).
Figure 2: argument reduction for the logarithm function
4 Conclusion

We have presented a radix-10 variant of the BKM algorithm. Extensions of this work to other radices (not too large, though) should be straightforward. A radix-10 BKM algorithm could be implemented into pocket calculators, to get fast complex elementary functions.

References


