Japan J. Indust. Appl. Math., 25 (2008), 255–279 Area ⟨2⟩

On Computation of a Power Series Root with Arbitrary Degree of Convergence

Takuya KITAMOTO

Faculty of Education, Yamaguchi University
1677–1 Yoshida, Yamaguchi 753–8513, Japan
E-mail: kitamoto@yamaguchi-u.ac.jp

Received September 29, 2005
Revised October 12, 2007

Given a bivariate polynomial $f(x, y)$, let $\phi(y)$ be a power series root of $f(x, y) = 0$ with respect to x, i.e., $\phi(y)$ is a function of y such that $f(\phi(y), y) = 0$. If $\phi(y)$ is analytic at $y = 0$, then we have its power series expansion

$$\phi(y) = \alpha_0 + \alpha_1 y + \alpha_2 y^2 + \cdots + \alpha_r y^r + \cdots . \tag{1}$$

Let $\phi^{(k)}(y)$ denote $\phi(y)$ truncated at y^k, i.e.,

$$\phi^{(k)}(y) = \alpha_0 + \alpha_1 y + \alpha_2 y^2 + \cdots + \alpha_k y^k. \tag{2}$$

Then, it is well known that, given initial value $\phi^{(0)}(y) = \alpha_0 \in \mathbf{C}$, the symbolic Newton's method with the formula

$$\phi^{(2^m-1)}(y) \leftarrow \phi^{(2^{m-1}-1)}(y) - \frac{f\left(\phi^{(2^{m-1}-1)}(y), y\right)}{\frac{\partial f}{\partial x}\left(\phi^{(2^{m-1}-1)}(y), y\right)} \pmod{y^{2^m}} \tag{3}$$

computes $\phi^{(2^m-1)}(y)$ $(1 \leq m)$ in (2) with quadratic convergence (the roots are computed in the order $\phi^{(0)}(y) \to \phi^{(2^1-1)}(y) \to \phi^{(2^2-1)}(y) \to \cdots \to \phi^{(2^m-1)}(y)$). References [1] and [3] indicate that the symbolic Newton's method can be generalized so that its convergence degree is an arbitrary integer p where its roots are computed in the order $\phi^{(0)}(y) \to \phi^{(p-1)}(y) \to \phi^{(p^2-1)}(y) \to \cdots \to \phi^{(p^m-1)}(y)$. Although the high degree convergent formula in [1] and [3] requires fewer iterations than the symbolic Newton's method, it may not be efficient as expected, since one iteration of the formula requires more computations than one in the symbolic Newton's method.

In this paper, we combine the polynomial evaluation method in [9] with the formula of arbitrary degree convergence and propose an algorithm that computes the above power series root $\phi^{(k)}(y)$. We analyze the complexity of the algorithm and give the number of multiplications/divisions required to compute a power series root in an explicit form. It is shown that when the degree of polynomial $f(x, y)$ is high, high degree convergent formula is advantageous over the symbolic Newton's method.

Key words: symbolic Newton's method, power series, high degree convergent, complexity analysis

1. Introduction

The computation of polynomial roots has a long history. One remarkable outcome of such computation of roots is Abel's impossibility theorem, which states that "in general, roots of polynomials with a degree of higher than four can not be expressed in terms of a finite number of additions, subtractions, multiplications, divisions, and root extractions." Because of this theorem, a root $\phi(y)$ of given

polynomial $f(x, y)$ with respect to x can not be expressed explicitly as a function of y when the degree of $f(x, y)$ with respect to x is over four. This motivates us to compute root $\phi(y)$ in the form of power series

$$\phi^{(k)}(y) = \alpha_0 + \alpha_1 y + \alpha_2 y^2 + \cdots + \alpha_k y^k \tag{4}$$

truncated at the k-th degree in y. In the rest of the paper, we refer to the above root as the "k-th power series root."

It is well known that power series roots can be computed by the symbolic Newton's method with quadratic convergence ([7, 8]), where its roots are computed in the order $\phi^{(0)}(y) \to \phi^{(2^1-1)}(y) \to \phi^{(2^2-1)}(y) \to \cdots \to \phi^{(2^m-1)}(y)$. References [1] and [3] propose a generalization of the symbolic Newton's method, where its convergence degree is arbitrary, say p, and its roots are computed in the order $\phi^{(0)}(y) \to \phi^{(p^1-1)}(y) \to \phi^{(p^2-1)}(y) \to \cdots \to \phi^{(p^m-1)}(y)$.

Since the p-th convergent formula requires $\lceil \log_p(k+1) \rceil$ iterations to compute k-th power series roots, when the degree p of convergence is high, the formula in [1] and [3] requires far fewer iterations than the symbolic Newton's method, whose degree p of convergence is two (i.e., $p = 2$). However, this does *not* immediately imply that a high degree convergent formula is more efficient than the symbolic Newton's method, because one iteration in a high degree convergence formula needs more computations than one in the symbolic Newton's method. In fact, numerical experiments often indicate that direct application of a high degree convergent formula is *not* more efficient than the symbolic Newton's method.

In this paper, we combine the high degree convergent formula in [1] with the polynomial evaluation method in [9] and derive a formula that is more efficient than the symbolic Newton's method when degree n of $f(x, y)$ with respect to x is large enough. We subject it to complexity analysis and count the number of multiplications/divisions required to compute a given power series root. We then analyze the behavior of the high degree convergent formula in terms of its complexity (interestingly, the formula is advantageous over the symbolic Newton's method when degree n of $f(x, y)$ with respect to x is large).

This paper is organized as follows: In Section 2 outlines definitions and notations, and Section 3 briefly discusses computations of a truncated power series. In Section 4, we review [1] and explain its high degree convergent formula, and in Section 5, we review [9], and explain its polynomial evaluation method. In Section 6, we combine the methods shown in Sections 4 and 5, and propose an algorithm to compute k-th power series roots $\phi^{(k)}(y)$ of a given polynomial $f(x, y)$ with an arbitrary degree of convergence. We then perform complexity analysis and count the number of multiplications/divisions required to compute k-th power series roots of a given polynomial $f(x, y)$. Section 6 represents the main contribution of the paper. In Section 7, we present the results of numerical experiments to confirm the validity of the complexity analysis in Section 6, and Section 8 outlines the conclusion.

2. Definitions and notations

In this paper, we employ the following notations:

Z: the set of integers
N: the set of natural numbers
R: the set of real numbers
C: the set of complex numbers
$\lceil z \rceil$: the minimum integer equal to or greater than $z \in \mathbf{R}$
$\mathrm{imod}(p, q)$: remainder of division of $p \in \mathbf{Z}$ by $q \in \mathbf{Z}$
$C(r, j)$: binomial coefficient, i.e., $\frac{r!}{j!\,(r-j)!}$

Definition 1. *The formula to compute power series root $\phi^{(k)}(y)$ in the order $\phi^{(0)}(y) \to \phi^{(p-1)}(y) \to \cdots \to \phi^{(p^m-1)}(y) \to \cdots$ is said to be p-th degree convergent.*

3. Computations of truncated power series

In this section, we discuss computations of truncated power series. Given two power series truncated at the k-th degree in y

$$\begin{aligned} a^{(k)} &= \alpha_0 + \alpha_1 y + \cdots + \alpha_k y^k, \\ b^{(k)} &= \beta_0 + \beta_1 y + \cdots + \beta_k y^k, \end{aligned}$$

we can define arithmetic (addition, subtraction, multiplication and division) between $a^{(k)}$ and $b^{(k)}$ as follows: We will find

$$c^{(k)} = \gamma_0 + \gamma_1 y + \cdots + \gamma_k y^k$$

such that

$$a^{(k)} \odot b^{(k)} \equiv c^{(k)} \pmod{y^{k+1}},$$

where $\odot$ is either $+$, $-$, $\times$, $/$. When $\odot = +$ or $\odot = -$, we obviously have

$$\begin{aligned} \gamma_i &= \alpha_i + \beta_i \quad (\text{when } \odot = +) \quad (i = 0, \ldots, k), \\ \gamma_i &= \alpha_i - \beta_i \quad (\text{when } \odot = -) \quad (i = 0, \ldots, k). \end{aligned}$$

When $\odot = \times$, looking at each coefficient of y^i $(i = 0, \ldots, k)$ gives us

$$\gamma_i = \alpha_0\beta_i + \alpha_1\beta_{i-1} + \cdots + \alpha_i\beta_0 \quad (i = 0, \ldots, k). \tag{5}$$

When $\odot = /$, from

$$c^{(k)} = a^{(k)}/b^{(k)} \pmod{y^{k+1}}$$

we obtain

$$a^{(k)} = b^{(k)}c^{(k)} \pmod{y^{k+1}}.$$

Thus, (5) tells us that

$$\begin{aligned}\alpha_0 &= \beta_0\gamma_0,\\ \alpha_1 &= \beta_1\gamma_0 + \beta_0\gamma_1,\\ &\cdots\\ \alpha_k &= \beta_k\gamma_0 + \cdots + \beta_1\gamma_{k-1} + \beta_0\gamma_k.\end{aligned}$$

Therefore, solving the above equation with respect to γ_i $(i = 0, 1, \ldots, k)$, we obtain

$$\gamma_i = \frac{\alpha_i - (\beta_i\gamma_0 + \cdots + \beta_1\gamma_{i-1})}{\beta_0} \quad (i = 0, 1, \ldots, k).$$

Thus, we have the following:

- Addition (subtraction) between the k-th power series requires k additions (subtractions) between numbers.
- Multiplication (division) between the k-th power series requires $\frac{k(k+1)}{2}$ additions (subtractions) and $\frac{(k+1)(k+2)}{2}$ multiplications or divisions between numbers.

We therefore assume the following in this paper:

(i) The computation time for arithmetic between numbers is negligible compared with that between truncated power series.

(ii) The computation time for addition and subtraction between truncated power series is negligible compared with that for multiplication and division.

(iii) The computation time for multiplication between truncated power series is the same as that for division between truncated power series.

In complexity analysis, we therefore count only the number of multiplications/divisions between truncated power series.

Throughout the paper, n, k and p denote the degree of $f(x, y)$ with respect to x, the truncation degree of power series and the degree of convergence, respectively.

NOTE. Theoretically, power series multiplication and division can be performed at lower cost (an FFT-based multiplication algorithm performs multiplication of the k-th degree polynomial with $O(k \log k \log\log k)$ numerical multiplications). However, we employ the above $O(k^2)$ multiplication algorithm in complexity analysis for the following reason: in the paper, we focus on analyzing the performance of the algorithm for common problems, and assume that truncation degree k is in the medium size range (say one hundred or so) where FFT-based algorithms are not so effective (k must be quite large to make such algorithms effective).

4. High degree convergent formula

Let $f(x, y)$ and $\phi(y)$ be a given polynomial and a root of $f(x, y) = 0$ with respect to x. When $f(x, 0)$ is square-free, k-th power series expansion $\phi^{(k)}(y)$ of $\phi(y)$

$$\phi^{(k)}(y) = \alpha_0 + \alpha_1 y + \alpha_2 y^2 + \cdots + \alpha_k y^k \tag{6}$$

can be computed by the symbolic Newton's method

$$\phi^{(2^m-1)}(y) \leftarrow \phi^{(2^{m-1}-1)}(y) - \frac{f\big(\phi^{(2^{m-1}-1)}(y), y\big)}{\frac{\partial f}{\partial x}\big(\phi^{(2^{m-1}-1)}(y), y\big)} \pmod{y^{2^m}} \tag{7}$$

in the order $\phi^{(0)}(y) \to \phi^{(2-1)}(y) \to \phi^{(2^2-1)}(y) \to \cdots \to \phi^{(2^m-1)}(y)$. Under the same conditions, reference [1] presented a high degree convergent formula that computes $\phi^{(k)}(y)$ in the order $\phi^{(0)}(y) \to \phi^{(p-1)}(y) \to \phi^{(p^2-1)}(y) \to \cdots \to \phi^{(p^m-1)}(y)$, where $p\ (\geq 2)$ is an arbitrary integer. The high degree convergent formula is given by

$$\begin{aligned}
&\phi^{(p^m-1)}(y) \\
&\leftarrow \phi^{(p^{m-1}-1)}(y) \\
&\quad - \frac{c_0\big(\phi^{(p^{m-1}-1)}(y)\big)}{c_1\big(\phi^{(p^{m-1}-1)}(y)\big) - c_0\big(\phi^{(p^{m-1}-1)}(y)\big)\frac{\tilde{H}_p(\phi^{(p^{m-1}-1)}(y))}{H_p(\phi^{(p^{m-1}-1)}(y))}} \pmod{y^{p^m}},
\end{aligned} \tag{8}$$

where $\tilde{H}_p(x)$ and $H_p(x)$ are defined by

$$\begin{aligned}
\tilde{H}_p(x) &= \mathrm{Det}\begin{pmatrix} c_2(x) & c_0(x) & 0 & \cdots & 0 \\ c_3(x) & c_1(x) & c_0(x) & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ c_{p-2}(x) & c_{p-4}(x) & c_{p-5}(x) & \cdots & c_0(x) \\ c_{p-1}(x) & c_{p-3}(x) & c_{p-4}(x) & \cdots & c_1(x) \end{pmatrix}, \\
H_p(x) &= \mathrm{Det}\begin{pmatrix} c_1(x) & c_0(x) & 0 & \cdots & 0 \\ c_2(x) & c_1(x) & c_0(x) & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ c_{p-3}(x) & c_{p-4}(x) & c_{p-5}(x) & \cdots & c_0(x) \\ c_{p-2}(x) & c_{p-3}(x) & c_{p-4}(x) & \cdots & c_1(x) \end{pmatrix}, \\
c_r(x) &= \frac{1}{r!}\frac{\partial^r f}{\partial x^r}(x, y)
\end{aligned} \tag{9}$$

(Det M denotes the determinant of matrix M). For details of the formula, refer to [1]. As an example, we show the 3rd-degree convergent formula.

$$\begin{aligned}
&\phi^{(3^m-1)}(y) \\
&\leftarrow \phi^{(3^{m-1}-1)}(y) \\
&\quad - \frac{f\big(\phi^{(3^{m-1}-1)}(y), y\big)}{\frac{\partial f}{\partial x}\big(\phi^{(3^{m-1}-1)}(y), y\big) - \frac{1}{2}f\big(\phi^{(3^{m-1}-1)}(y), y\big)\frac{\frac{\partial^2 f}{\partial x^2}(\phi^{(3^{m-1}-1)}(y), y)}{\frac{\partial f}{\partial x}(\phi^{(3^{m-1}-1)}(y), y)}} \pmod{y^{3^m}}.
\end{aligned} \tag{10}$$

5. Polynomial evaluation

5.1. Polynomial evaluation of $f(x, y)$ and $\frac{\partial f}{\partial x}(x, y)$

5.1.1. Algorithm description

Ordinarily, a polynomial $h(x) = a_n x^n + a_{n-1}x^{n-1} + \cdots + a_0$ is evaluated at $x = x_0$ by Horner's rule

$$h(x_0) = (\cdots(a_n x_0 + a_{n-1})x_0 + \cdots + a_1)x_0 + a_0,$$

which requires n additions and multiplications. With this rule, one iteration of the symbolic Newton's method (7) requires $2n$ power series multiplications or divisions (n multiplications for evaluation of $f(x, y)$ at $x = \phi^{(2^{m-1})}(y)$, $(n-1)$ multiplications for evaluation of $\frac{\partial f}{\partial x}(x, y)$ at $x = \phi^{(2^{m-1})}(y)$ and one division). However, polynomials $f(x, y)$ and $\frac{\partial f}{\partial x}(x, y)$ are closely related, and evaluation of both at the same value can be performed more efficiently. In fact, [9] presents an algorithm to evaluate $h(x)$ and $h'(x)$ at $x = x_0$, which requires $\left(n + 2\left\lceil (n+1)^{\frac{1}{2}}\right\rceil - 1\right)$ multiplications. The algorithm is based on the theorem outlined below.

THEOREM 1 ([9]). *Let $h(x) = a_n x^n + a_{n-1}x^{n-1} + \cdots + a_0$ be a given polynomial. Without loss of generality, we assume that $q = (n+1)^{\frac{1}{2}}$ is an integer (if this is not the case, we let $q = \lceil (n+1)^{\frac{1}{2}} \rceil$ and assume that $h(x)$ is a (q^2-1)-th degree polynomial whose i-th ($i = n+1, \ldots, q^2-1$) coefficients are 0). Defining $T_{j,i}$ by*

$$\begin{aligned}
&T_{-1,i-1} = a_{n-i} x_0^{s(i)} \quad (i = 1, 2, \ldots, n),\\
&T_{j,j} = a_n x_0^{s(0)} \quad (j = 0, 1, \ldots, n-1),\\
&T_{j,i} = T_{j-1,i-1} + T_{j,i-1} x_0^{s(i-j)-s(i-j-1)+1} \quad (j = 0, 1, \ldots, n-1,\ i = j+1, \ldots, n),
\end{aligned}$$

where $s(i) = \mathrm{imod}(n-i, q)$, we have

$$T_{j,i} = x^{s(i-j)} \sum_{r=j}^{i} C(r, j) a_{n-i+r} x^{r-j} \quad (1 \le i \le n,\ -1 \le j < n). \tag{11}$$

As a corollary of the above theorem, we obtain the following:

COROLLARY 1 ([9]). *Under the same conditions as the above theorem, we have*

$$h^{(j)}(x_0) = j! \frac{T_{j,n}}{x_0^{s(n-j)}} \quad (0 \le j < n). \tag{12}$$

Applying $j = 0, 1$ to the above corollary, we obtain $f(x_0) = T_{0,n}$, $f'(x_0) = \frac{T_{1,n}}{x_0}$. Thus, we obtain the following algorithm:

Algorithm 1. Polynomial evaluation method [9]

Input: $f(x,y) = a_n(y)x^n + \cdots + a_0(y)$, $\phi(y)$;

Output: $f(\phi(y),y)$, $\frac{\partial f}{\partial x}(\phi(y),y)$;

(a) Let $q \leftarrow \lceil (n+1)^{\frac{1}{2}} \rceil$, $N \leftarrow q^2 - 1$

(b) Let $a_i(y) \leftarrow 0$ $(i = n+1, \ldots, N)$ (we regard $f(x,y)$ as an N-th degree polynomial w.r.t. x).

(c) Compute and save the value of $\phi(y)^i$ $(i = 2, \ldots, q)$.

(d) Let $T_{-1,i-1} \leftarrow a_{N-i}(y)\phi(y)^{s(i)}$ $(i = 1, \ldots, N)$, where $s(j) = \mathrm{imod}(N-j, q)$.

(e) Let $T_{j,j} \leftarrow a_N(y)\phi(y)^{q-1}$ $(j = 0, 1)$.

(f) Let

$$T_{j,i} \leftarrow \begin{cases} T_{j-1,i-1} + T_{j,i-1}\phi(y)^q, & \text{when } \mathrm{imod}(i-j,q) = 0 \\ T_{j-1,i-1} + T_{j,i-1}, & \text{when } \mathrm{imod}(i-j,q) \neq 0 \end{cases}$$

$(j = 0, 1,\ i = j+1, \ldots, N)$.

(g) Output $f(\phi(y),y) = T_{0,N}$, $\frac{\partial f}{\partial x}(\phi(y),y) = \frac{T_{1,N}}{\phi(y)}$.

5.1.2. Complexity analysis

As shown in reference [2], the above algorithm requires $\left(n + 2\lceil (n+1)^{\frac{1}{2}} \rceil + M\right)$ power series multiplications or divisions, where M is an integer that is either -1 or 0. To simplify the notation, we fix M to $M = 0$ in the rest of the paper.

5.2. Evaluation of higher derivatives

Algorithm 1 in the previous subsection evaluates $f(\phi(y),y)$, $\frac{\partial f}{\partial x}(\phi(y),y)$. In this subsection, we discuss extension of the algorithm to evaluate higher derivatives $\frac{\partial^r f}{\partial x^r}(\phi(y),y)$ $(r = 2, \ldots, p-1)$.

5.2.1. Algorithm modification

The following modification is enough to compute higher derivatives $\frac{\partial^r f}{\partial x^r}(\phi(y),y)$ $(2 \le r \le p-1)$ with Algorithm 1.

(i) In step (e) and (f), "$j = 0, 1$" should be modified to "$j = 0, 1, \ldots, p-1$."

(ii) In step (g), output $\frac{\partial^r f}{\partial x^r}(\phi(y),y) = r!\,\frac{T_{r,N}}{\phi(y)^{s(n-r)}}$ $(r = 2, \ldots, p-1)$ in addition to $f(\phi(y),y) = T_{0,N}$, $\frac{\partial f}{\partial x}(\phi(y),y) = \frac{T_{1,N}}{\phi(y)}$.

5.2.2. Complexity analysis

We count the number of additional power series multiplications or divisions required for each modification of the above algorithm. To simplify the analysis, we limit the range of convergence degree p to

$$2 \le p \le q, \tag{13}$$

where q is defined by

$$q = \lceil \sqrt{n+1}\, \rceil \tag{14}$$

(in the rest of the paper, q denotes $\lceil \sqrt{n+1}\, \rceil$).

(i) Modification in step (e) does not require any additional multiplications or divisions.

(ii) Modification in step (f) requires $(p-2)(q-1)$ additional multiplications or divisions (multiplication of $T_{j,i-1}$ and $\phi(y)^q$ when $j = 2, \ldots, p-1$, $i = q+j$, $2q+j, \ldots, (q-1)q+j$).

(iii) Modification in step (g) requires $(p-2)$ divisions (division of $\frac{\partial^r f}{\partial x^r}(\phi(y), y)$ by $\phi(y)^r$ where $r = 2, \ldots, p-1$).

Thus the total number of power series multiplications or divisions required for the modified algorithm is $n+2q+(p-2)(q-1)+(p-2) = n+pq = n+(\lceil\sqrt{n+1}\,\rceil)p$.

6. Algorithm to compute power series roots

6.1. Algorithm description

We combine the algorithm for computing power series roots in Section 4 and the polynomial evaluation algorithm in Section 5 to obtain the algorithm below.

Algorithm 2. Power series root computation

Input: $f(x,y)$, $\phi^{(0)}(y)$, $p \in \mathbf{Z}$, $k \in \mathbf{Z}$;

Output: k-th power series root;

Note: The algorithm is of p-th degree convergence;

(a) Let $m \leftarrow 1$.

(b) Compute $\frac{\partial^r f}{\partial x^r}\big(\phi^{(p^{m-1}-1)}(y), y\big)$ $(r = 0, 1, \ldots, p-1)$ with the polynomial evaluation algorithm in Section 5, where the truncation degree of the power series arithmetic is set to $\min(k, p^m - 1)$.

(c) Compute $\phi^{(p^m-1)}(y)$ with the formula (8), where truncation degree of the power series arithmetic is set to $\min(k, p^m - 1)$.

(d) If $(k \leq p^m - 1)$, then $\phi^{(k)}(y)$ is output and the algorithm finishes. Otherwise let $m \leftarrow m+1$ and go back to step (b).

6.2. Accuracy verification

From formula (8), we see that the numerical errors in coefficients α_i $(i = 0, 1, \ldots, p^{m-1}-1)$ in (4) directly affect to the numerical errors in the coefficients α_i $(i = p^{m-1}, \ldots, p^m - 1)$. This implies that formula (8) is numerically unstable, i.e.,

$$\text{numerical error in coefficient } \alpha_k \text{ accumerates as index } k \text{ increases.} \tag{15}$$

We therefore need to verify the accuracy of the results, when floating point arithmetic is used. Such verification can be performed as follows: The above point (15) implies that if a coefficient α_k is accurate, then so are the lower degree coefficients α_i $(i = 0, 1, \ldots, k-1)$. Hence, to check the numerical accuracy of a power series $\phi^{(k)}(y)$ computed by formula (8), it is enough to examine the highest degree coefficient α_k. Reference [6] presents a method to compute a high degree coefficient α_k accurately using Cauchy's integral formula. Thus, we can verify the accuracy

of a power series $\phi^{(k)}(y)$ computed, comparing the highest degree coefficient α_k of $\phi^{(k)}(y)$ with that computed using the method in [6]. For details of this method of accuracy verification, refer to [6].

6.3. Complexity analysis

6.3.1. The number of power series multiplications/divisions

We count the number of power series multiplications or divisions in each step of Algorithm 2. Steps (a) and (d) require no multiplications or divisions. Step (b) requires $(n + (\lceil\sqrt{n+1}\,\rceil)p)$ multiplications/divisions as indicated in Section 5.2.2. When $p > 2$, we need to evaluate

(i) $\tilde{H}_p\big(\phi^{(p^{m-1}-1)}(y)\big)$, $H_p\big(\phi^{(p^{m-1}-1)}(y)\big)$

and evaluate

(ii) $$\frac{c_0\big(\phi^{(p^{m-1}-1)}(y)\big)}{c_1\big(\phi^{(p^{m-1}-1)}(y)\big) - c_0\big(\phi^{(p^{m-1}-1)}(y)\big)\frac{\tilde{H}_p\left(\phi^{(p^{m-1}-1)}(y)\right)}{H_p\left(\phi^{(p^{m-1}-1)}(y)\right)}}$$

in step (c). Since both matrices in $\tilde{H}_p(x)$ and $H_p(x)$ are in Hessenberg form, each can be evaluated at most with $\frac{p^2-p-6}{2}$ power series multiplications/divisions ($\big(\frac{(p-1)(p-2)}{2} - 1\big)$ for diagonalizations and $(p-3)$ for multiplications of diagonal elements). Thus the above (i) therefore takes (p^2-p-6) multiplications/divisions. It is easy to see that the above (ii) takes three multiplications/divisions and, in total, step (c) requires $(p^2 - p - 3)$ multiplications/divisions. When $p > 2$, each iteration of Algorithm 2 therefore requires

$$n + p(\lceil\sqrt{n+1}\,\rceil) + p^2 - p - 3 \tag{16}$$

power series multiplications or divisions.

When $p = 2$, we have convergent formula (7) (the symbolic Newton's method) instead of (8). We therefore need to evaluate

$$\frac{c_0\big(\phi^{(2^{m-1}-1)}(y)\big)}{c_1\big(\phi^{(2^{m-1}-1)}(y)\big)} \tag{17}$$

in step (c), which requires one division. Therefore, when $p = 2$, each iteration of Algorithm 2 requires

$$n + p(\lceil\sqrt{n+1}\,\rceil) + 1 \tag{18}$$

power series multiplications or divisions. From (16) and (18), each iteration of Algorithm 2 requires

$$\Psi(n,p) \stackrel{\text{def}}{=} \begin{cases} n + p(\lceil\sqrt{n+1}\,\rceil) + 1 & \text{when } p = 2, \\ n + p(\lceil\sqrt{n+1}\,\rceil) + p^2 - p - 3 & \text{when } p > 2. \end{cases} \tag{19}$$

Given $k \in \mathbf{Z}$ (the truncation degree of the power series root) and $p \in \mathbf{Z}$ (the degree of convergence), Algorithm 2 in this section requires $\lceil\log_p(k+1)\rceil$ iterations

to compute a k-th power series root. Hence, the total number of power series multiplications and divisions in the algorithm is $\lceil \log_p(k+1) \rceil \, \Psi(n,p)$.

However, the above number (the number of power series arithmetic steps) does not directly reflect the CPU time required to perform the algorithm as the time taken for one power series multiplication/division depends on its truncation degree k (recall that one power series multiplication/division requires $\frac{(k+1)(k+2)}{2}$ multiplications/divisions of numbers). This implies that to make complexity analysis valid, we need to count the number of numerical multiplications/divisions.

6.3.2. The number of numerical multiplications/divisions

When we perform Algorithm 2, the truncation degree of the power series at the m-th iteration of steps (b) and (c) is

$$\begin{cases} p^m - 1 & \text{when } p^m - 1 < k, \\ k & \text{when } p^m - 1 \geq k. \end{cases}$$

Thus at m-th iteration of steps (b) and (c),

$$\begin{cases} \dfrac{1}{2} p^m (p^m + 1) & \text{when } p^m - 1 < k, \\ \dfrac{1}{2}(k+1)(k+2) & \text{when } p^m - 1 \geq k, \end{cases}$$

numerical multiplications/divisions are required for each power series calculation. Since each iteration of steps (b)–(d) requires $\Psi(n,p)$ power series multiplications/divisions, the number of numerical multiplications/divisions necessary at the m-th iteration is

$$\begin{cases} \dfrac{1}{2} p^m (p^m + 1)\, \Psi(n,p) & \text{when } p^m - 1 < k, \\ \dfrac{1}{2}(k+1)(k+2)\, \Psi(n,p) & \text{when } p^m - 1 \geq k. \end{cases} \tag{20}$$

Let

$$t = \lceil \log_p(k+1) \rceil, \tag{21}$$

then the condition $(p^m - 1 < k)$ is equivalent to $m \leq t-1$. Thus, adding the above (20) for $m = 1, 2, \ldots, t-1, t$, we obtain

$$\begin{aligned} \Omega(n,k,p) &\stackrel{\text{def}}{=} \sum_{m=1}^{t-1} \left(\frac{1}{2} p^m (p^m + 1)\, \Psi(n,p) \right) + \frac{1}{2}(k+1)(k+2)\, \Psi(n,p) \\ &= \frac{\Psi(n,p)}{2} \left\{ \sum_{m=1}^{t-1} (p^{2m} + p^m) + (k+1)(k+2) \right\} \\ &= \frac{\Psi(n,p)}{2} \left\{ \frac{p^{2t} - p^2}{p^2 - 1} + \frac{p^t - p}{p - 1} + (k+1)(k+2) \right\}, \end{aligned}$$

which is the number of numerical multiplications/divisions required to perform Algorithm 2.

6.4. Analysis of Algorithm 2

Having obtained function $\Omega(n,k,p)$, which gives the complexity of Algorithm 2, it is natural to ask what convergence degree p is the most efficient (i.e., which p gives the minimum of $\Omega(n,k,p)$) for a given n (the degree of $f(x,y)$ with respect to x) and k (the truncation degree of the power series root). However, the behavior of the function $\Omega(n,k,p)$ is rather chaotic due to the non-continuous properties of $\lceil \log_p(k+1) \rceil$ and $\lceil \sqrt{n+1}\, \rceil$. We therefore approximate function $\Omega(n,k,p)$ and, by analyzing the properties of the approximated function, we examine the asymptotic behavior of $\Omega(n,k,p)$.

6.4.1. Approximation of $\Omega(n,k,p)$

To analyze the complexity, we approximate t and $\Psi(n,p)$ in (21) with

$$\tilde{t} \stackrel{\text{def}}{=} \log_p(k+1) \tag{22}$$

and

$$\tilde{\Psi}(n,k) \stackrel{\text{def}}{=} n + p(\lceil \sqrt{n+1}\, \rceil) + p^2 - p - 3 \tag{23}$$

respectively, and define approximation $\tilde{\Omega}(n,k,p)$ of $\Omega(n,k,p)$ by

$$\tilde{\Omega}(n,k,p) \stackrel{\text{def}}{=} \frac{\tilde{\Psi}(n,p)}{2}\left\{\frac{p^{2\tilde{t}}-p^2}{p^2-1} + \frac{p^{\tilde{t}}-p}{p-1} + (k+1)(k+2)\right\} \tag{24}$$

($\tilde{\Omega}(n,k,p)$ is the same as $\Omega(n,k,p)$ except that t and $\Psi(n,p)$ are replaced by $\tilde{t}$ and $\tilde{\Psi}(n,p)$, respectively). Since we have $\tilde{t} \le t$ and $\tilde{\Psi}(n,p) < \Psi(n,p)$, where the equality holds when k is in the form of

$$k = p^r - 1, \quad (r \in \mathbf{Z},\ p > 2) \tag{25}$$

it is easy to see that

$$\tilde{\Omega}(n,k,p) \le \Omega(n,k,p), \tag{26}$$

where the equality holds when k is in the form of (25).

Since we have $p^{\tilde{t}} = k+1$, $\tilde{\Omega}(n,k,p)$ can be written as

$$\begin{aligned}\tilde{\Omega}(n,k,p) &= \frac{\tilde{\Psi}(n,p)}{2}\left\{\frac{(k+1)^2-p^2}{p^2-1} + \frac{(k+1)-p}{p-1} + (k+1)(k+2)\right\} \\ &= \frac{kp(kp+3p+1)\,\tilde{\Psi}(n,p)}{2(p^2-1)}.\end{aligned} \tag{27}$$

6.4.2. Analysis of $\tilde{\Omega}(n, k, p)$

We define $\lambda(p)$ by

$$\lambda(p) \stackrel{\text{def}}{=} \tilde{\Omega}(n, k, p) \tag{28}$$

and regard $\lambda(p)$ as a function of p with parameters n and k. We have the theorem shown below on $\lambda(p)$.

THEOREM 2. *Let n and k be integers satisfying $n \geq 3$, $k \geq 1$. Then, in $\{p \mid 4/3 \leq p \leq q\}$, $\lambda(p)$ is either monotonically decreasing or has exactly one local minimum.*

To prove the above theorem, we need the lemma outlined below.

LEMMA 1. *Under the same condition as the above theorem, we have*

$$\lambda''(p) > 0 \quad (p \geq 4/3). \tag{29}$$

Proof. From the above definition, we have

$$\lambda''(p) = \frac{kz(p)}{(p^2-1)^3},$$

where $z(p)$ is a function defined by

$$\begin{aligned} z(p) \stackrel{\text{def}}{=} {} & (k+3)p^6 - 3(k+3)p^4 + \{n+(k+3)q-k-5\}p^3 + 3\{k(n-1)+3n+q-4\}p^2 \\ & + 3\{n+(k+3)q-k-5\}p + k(n-3) + 3n + q - 10 \end{aligned} \tag{30}$$

(q is given by (14)). It is easy to see that the following is enough to prove the lemma:

$$z(p) > 0 \quad (p \geq 4/3). \tag{31}$$

From $z^{(4)}(p) = (360p^2 - 72)k + 1080p^2 - 216$, it is easy to see that $z^{(4)}(p) > 0$ $(p \geq 4/3)$. This and the fact that

$$z^{(3)}(4/3) = \frac{2}{9}\{27n + (27q + 821)k + 81q + 2409\} > 0$$

imply $z^{(3)}(z) > 0$ $(p \geq 4/3)$. This and the fact that

$$z^{(2)}(4/3) = \frac{2}{27}\{(81k + 351)n + (108q + 227)k + 405q + 384\} > 0$$

imply $z^{(2)}(z) > 0$ $(p \geq 4/3)$. This and the fact that

$$\begin{aligned} z'(4/3) = {} & \frac{1}{81}[\{648(k-1) + 3267\}(n-3) \\ & + (675q + 365)(k-1) + 3348q + 1487] > 0 \end{aligned}$$

imply $z'(p) > 0$ $(p \geq 4/3)$. This and the fact that

$$z(4/3) = \frac{1}{729}[\{4617(k-1)+23112\}(n-3) + \{4644q+316\}(k-1)+23193q+1291] > 0$$

imply (31), which proves the lemma. □

Proof of Theorem 2. Note that we have

$$\lambda'(4/3) = -\frac{k}{294}[\{648(k-1)+3267\}(n-3) + \{528(q-1)+256\}(k-1)+2760(q-1)+1152] < 0. \quad (32)$$

If $\lambda'(q) \geq 0$, then (32) and the intermediate value theorem imply that

$$\exists p \ (4/3 \leq p \leq q), \quad \lambda'(p) = 0,$$

which prove the existence of a local minimum. The uniqueness of the local minimum follows from Lemma 1.

If $\lambda'(q) < 0$, then Lemma 1 implies that

$$\forall p \ (4/3 \leq p \leq q), \quad \lambda'(p) < 0,$$

which indicates that $\lambda(p)$ is monotonically decreasing. This completes the proof. □

The above theorem implies that given n $(\in \mathbf{Z})$ and k $(\in \mathbf{Z})$, p such that $\tilde{\Omega}(n,k,p)$ $(4/3 \leq p \leq q)$ takes its minimum, is uniquely determined and is a function of n and k. We denote such p by $\tilde{p}(n,k)$. In other words, $\tilde{p}(n,k)$ is a function of n and k such that

$$\min_{4/3 \leq p \leq q} \tilde{\Omega}(n,k,p) = \tilde{\Omega}(n,k,\tilde{p}(n,k)). \quad (33)$$

Since we have $\lambda'(p) = \frac{\partial \tilde{\Omega}}{\partial p}(n,k,p)$, the proof of Theorem 2 implies that the above $\tilde{p}(n,k)$ can be written as

$$\tilde{p}(n,k) = \begin{cases} p \text{ such that } \dfrac{\partial \tilde{\Omega}}{\partial p}(n,k,p) = 0, & \text{when } \dfrac{\partial \tilde{\Omega}}{\partial p}(n,k,q) \geq 0, \\ q, & \text{when } \dfrac{\partial \tilde{\Omega}}{\partial p}(n,k,q) < 0. \end{cases} \quad (34)$$

By the definition, $\tilde{p}(n,k)$ denotes optimal p in $(4/3 \leq p \leq q)$ for the given n and k from the viewpoint of computational complexity.

Regarding $\tilde{p}(n,k)$ as a function of n with parameter k, we have the theorem outlined below concerning the behavior of $\tilde{p}(n,k)$ when $n \to \infty$.

THEOREM 3. *Let k be a constant number satisfying $k \geq 1$. We then have*

$$\lim_{n\to\infty} \tilde{p}(n,k) = \infty. \tag{35}$$

Proof. From (34), $\tilde{p}(n,k)$ is equal to either q or p such that $\frac{\partial\tilde{\Omega}}{\partial p}(n,k,p) = 0$. When $\tilde{p}(n,k) = q$, the theorem easily follows from (14). Thus, we will prove the theorem, assuming that

$$\tilde{p}(n,k) = \left(p \text{ such that } \frac{\partial\tilde{\Omega}}{\partial p}(n,k,p) = 0\right) \tag{36}$$

when $n \to \infty$. From Lemma 1, we have

$$\lambda''(p) = \frac{\partial^2\tilde{\Omega}}{\partial p^2}(n,k,p) > 0 \quad (n \geq 3,\ k \geq 1),$$

which implies that $\frac{\partial\tilde{\Omega}}{\partial p}(n,k,p_0)$ $(n \geq 3,\ k \geq 1)$ is an increasing function of p. This and (36) imply that

$$p_0 < \tilde{p}(n,k) \iff \frac{\partial\tilde{\Omega}}{\partial p}(n,k,p_0) < 0. \tag{37}$$

To prove (35), it is enough to show that

$$\forall M,\ \exists N \text{ such that } (n > N \to \tilde{p}(n,k) > M). \tag{38}$$

Given an integer $M \in \mathbf{Z}$, let N be

$$N = \max\{M^5, 2(3k+10), 32(k+3)^{10}\}. \tag{39}$$

Suppose that n is an integer satisfying $n > N$. Then

$$\frac{\partial\tilde{\Omega}}{\partial p}\big(n,k,n^{\frac{1}{5}}\big) = \frac{k\eta(n,k)}{2(p^2-1)^2}, \tag{40}$$

where $\eta(n,k)$ is defined by

$$\begin{aligned}\eta(n,k) \;=\; & -n^{\frac{7}{5}} - (2k+6)n^{\frac{6}{5}} + (2k+5)n + \{(k+3)q - k - 2\}n^{\frac{4}{5}} \\ & - 4(k+3)n^{\frac{3}{5}} - 3(k+3)(q-1)n^{\frac{2}{5}} + 2(3k-q+10)n^{\frac{1}{5}} + 3.\end{aligned} \tag{41}$$

Note that (39) implies that

$$n^{\frac{1}{5}} > M, \quad n > 2(3k+10), \quad n^{\frac{1}{10}} > \sqrt{2}(k+3), \tag{42}$$

and from $q = \lceil \sqrt{n+1} \rceil$, we obtain

$$1 < \sqrt{n} < q < \sqrt{2n}. \tag{43}$$

Thus, we have

$$\begin{aligned}
\eta(n,k) &= -n^{\frac{7}{5}} - (2k+5)\big(n^{\frac{6}{5}} - n\big) - n^{\frac{6}{5}} + (k+3)qn^{\frac{4}{5}} - \big\{(k+2)n^{\frac{4}{5}} - 3\big\} \\
&\quad - 4(k+3)n^{\frac{3}{5}} - 3(k+3)(q-1)n^{\frac{2}{5}} + 2(3k+10)n^{\frac{1}{5}} - 2qn^{\frac{1}{5}} \\
&< -n^{\frac{7}{5}} - n^{\frac{6}{5}} + (k+3)qn^{\frac{4}{5}} + 2(3k+10)n^{\frac{1}{5}} \quad (\because n \geq 3,\ k \geq 1) \\
&< -n^{\frac{7}{5}} - n^{\frac{6}{5}} + \sqrt{2}(k+3)n^{\frac{13}{10}} + 2(3k+10)n^{\frac{1}{5}} \quad (\because (43)) \\
&= -\big\{n^{\frac{1}{10}} - \sqrt{2}(k+3)\big\}n^{\frac{13}{10}} - \{n - 2(3k+10)\}n^{\frac{1}{5}} \\
&< 0 \quad (\because (42)).
\end{aligned}$$

This and (40) imply that $\frac{\partial \tilde{\Omega}}{\partial p}\big(n,k,n^{\frac{1}{5}}\big) < 0$. Thus, (37) with $p_0 = n^{\frac{1}{5}}$ implies that

$$\tilde{p}(n,k) > n^{\frac{1}{5}} > M \quad (\because (42)), \tag{44}$$

and (38) is shown. This completes the proof. □

6.4.3. Observations on $\tilde{\Omega}(n,k,p)$

Summarizing Theorems 2 and 3, we obtain the following observations on $\tilde{\Omega}(n,k,p)$:

(A1) As a function of p, $\tilde{\Omega}(n,k,p)$ is either monotonically decreasing or has exactly one local minimum in $\{p \mid 4/3 \leq p \leq q\}$ for any $n\ (\geq 3) \in \mathbf{N}$ and $k\ (\geq 1) \in \mathbf{N}$.

(A2) Let $\tilde{p}(n,k)$ denote p such that $\tilde{\Omega}(n,k,\tilde{p}(n,k))$ gives the minimum of $\tilde{\Omega}(n,k,p)$ above. Then, as a function of n, $\tilde{p}(n,k)$ satisfies $\lim_{n\to\infty} \tilde{p}(n,k) = \infty$.

6.5. Analysis of $\Omega(n,k,p)$

In this subsection, we examine the properties of $\Omega(n,k,p)$. First, we recall the following relationships between $\Omega(n,k,p)$ and its approximation $\tilde{\Omega}(n,k,p)$

$$\Omega(n,k,p) \geq \tilde{\Omega}(n,k,p), \tag{45}$$

$$\Omega(n,k,p) = \tilde{\Omega}(n,k,p) \quad (k = p^r - 1,\ r \in \mathbf{Z},\ p > 2). \tag{46}$$

Thus, as a function of p, $\tilde{\Omega}(n,k,p)$ supports $\Omega(n,k,p)$ from the bottom and they coincide at $p = \sqrt[r]{k+1}$ $(r \in \mathbf{Z},\ p > 2)$. As an illustrative example, we plot $\Omega(100,30,p)$ $(2 \leq p \leq q)$ (denoted by the solid line) and $\tilde{\Omega}(100,30,p)$ $(2 \leq p \leq q)$ (denoted by the dashed line) in Fig. 1. From the figure, it is clear that $\Omega(n,k,p)$ does not inherit property (A1) of $\tilde{\Omega}(n,k,p)$ (obviously, it has plural local minimums as shown in Fig. 1). However, it has a property similar to (A2). To state this property, let $p(n,k)$ be the same as $\tilde{p}(n,k)$, except $\tilde{\Omega}(n,k,p)$ is replaced with $\Omega(n,k,p)$. In other words, $p(n,k)$ is a function of n and k such that

$$\Omega(n,k,p(n,k)) = \min_{2 \leq p \leq q} \Omega(n,k,p). \tag{47}$$

We then have the theorem outlined below.

THEOREM 4. *Let k be a constant number satisfying $k \geq 2$. We then have*

$$\lim_{n\to\infty} p(n,k) = k+1. \tag{48}$$

Proof. It is enough to prove that

$$\exists M \in \mathbf{R}, \quad (n > M) \implies p(n,k) = k+1, \tag{49}$$

where M is a large enough number determined by k. From the definition of $p(n,k)$, the above is equivalent to

$$(n > M) \implies \min_{2\leq p\leq q} \Omega(n,k,p) = \Omega(n,k,k+1). \tag{50}$$

Note that if we set M so that it satisfies $M > (k+1)^2$, then we have

$$(n > M) \implies q = \lceil \sqrt{n+1}\, \rceil > \sqrt{(k+1)^2+1} > k+1, \tag{51}$$

which implies that

$$\begin{aligned} (n > M) \implies & \min_{2\leq p\leq q} \Omega(n,k,p) \\ & = \min\{\Omega(n,k,2), \ldots, \Omega(n,k,k+1), \ldots, \Omega(n,k,q)\} \\ & \leq \Omega(n,k,k+1). \end{aligned} \tag{52}$$

Since we obviously have

$$\min_{2\leq p} \Omega(n,k,p) \leq \min_{2\leq p\leq q} \Omega(n,k,p), \tag{53}$$

if we prove

$$(n > M) \implies \min_{2\leq p} \Omega(n,k,p) = \Omega(n,k,k+1), \tag{54}$$

then (52), (53) and (54) imply (50), and the theorem is proved. To prove (54), we first note that it can be written as

$$(n > M) \implies \forall p_0\ (\geq 2),\ \Omega(n,k,k+1) \leq \Omega(n,k,p_0). \tag{55}$$

We assume that n is an integer satisfying $(n > M)$ in the rest of the proof, and split the above inequality into the following two inequalities:

$$(2 \leq p_0 < k+1) \implies \Omega(n,k,p_0) \geq \Omega(n,k,k+1), \tag{56}$$

$$(k+1 \leq p_0) \implies \Omega(n,k,p_0) \geq \Omega(n,k,k+1). \tag{57}$$

First we prove (56). Theorem 3 tells us that, for sufficiently large number M, we have (recall that n satisfies $n > M$)

$$k + 1 \leq \tilde{p}(n, k). \tag{58}$$

Lemma 1 tells us that $\lambda'(p) = \frac{\partial \tilde{\Omega}}{\partial p}(n, k, p)$ $(2 \leq p)$ is monotonically increasing as a function of p, and this implies

$$a < b \iff \frac{\partial \tilde{\Omega}}{\partial p}(n, k, a) < \frac{\partial \tilde{\Omega}}{\partial p}(n, k, b). \tag{59}$$

Substituting $a = p_0$, $b = \tilde{p}(n, k)$ into (59), we obtain

$$p_0 < \tilde{p}(n, k) \iff \frac{\partial \tilde{\Omega}}{\partial p}(n, k, p_0) < \frac{\partial \tilde{\Omega}}{\partial p}(n, k, \tilde{p}(n, k)). \tag{60}$$

Note that from (34), we obtain

$$\frac{\partial \tilde{\Omega}}{\partial p}(n, k, \tilde{p}(n, k)) = \begin{cases} 0 & \text{when } \dfrac{\partial \tilde{\Omega}}{\partial p}(n, k, q) \geq 0, \\ \dfrac{\partial \tilde{\Omega}}{\partial p}(n, k, q) & \text{when } \dfrac{\partial \tilde{\Omega}}{\partial p}(n, k, q) < 0, \end{cases} \tag{61}$$

which implies

$$\frac{\partial \tilde{\Omega}}{\partial p}(n, k, \tilde{p}(n, k)) \leq 0. \tag{62}$$

This and (60) imply that

$$p_0 < \tilde{p}(n, k) \implies \frac{\partial \tilde{\Omega}}{\partial p}(n, k, p_0) < 0. \tag{63}$$

Thus, from (58) we obtain

$$p_0 < k + 1 \implies p_0 < \tilde{p}(n, k) \implies \frac{\partial \tilde{\Omega}}{\partial p}(n, k, p_0) < 0. \tag{64}$$

This shows that $\tilde{\Omega}(n, k, p)$ is monotonically decreasing as a function of p in the range $(2 \leq p < k + 1)$, which implies that

$$(2 \leq p_0 < k + 1) \implies \tilde{\Omega}(n, k, p_0) \geq \tilde{\Omega}(n, k, k + 1). \tag{65}$$

Note that from (45) and (46), we have

$$\tilde{\Omega}(n, k, k + 1) = \Omega(n, k, k + 1), \tag{66}$$

$$\Omega(n, k, p_0) \geq \tilde{\Omega}(n, k, p_0). \tag{67}$$

Thus, (65), (66) and (67) imply that

$$\begin{aligned}(2 \le p_0 < k+1) &\implies \tilde{\Omega}(n,k,p_0) \ge \tilde{\Omega}(n,k,k+1) \quad (\because (65))\\ &\implies \tilde{\Omega}(n,k,p_0) \ge \Omega(n,k,k+1) \quad (\because (66))\\ &\implies \Omega(n,k,p_0) \ge \Omega(n,k,k+1) \quad (\because (67)),\end{aligned}$$

which proves (56).

Next we will prove (57). Suppose $(k+1) \le p$. We then have $t = \lceil \log_p(k+1) \rceil = 1$. Thus, we obtain

$$\begin{aligned}\Omega(n,k,p) &= \frac{\Psi(n,p)}{2}\left\{\frac{p^{2t}-p^2}{p^2-1} + \frac{p^t-p}{p-1} + (k+1)(k+2)\right\}\\ &= \frac{1}{2}(k+1)(k+2)\,\Psi(n,p). \qquad (68)\end{aligned}$$

From $2 < k+1 < p$, we have

$$\Psi(n,p) = n + p(\lceil \sqrt{n+1}\,\rceil) + p^2 - p - 3.$$

This and (68) imply that

$$\begin{aligned}\frac{\partial \Omega}{\partial p}(n,k,p) &= \frac{1}{2}(k+1)(k+2)\frac{\partial \Psi}{\partial p}(n,p)\\ &= \frac{1}{2}(k+1)(k+2)(2p + \lceil \sqrt{n+1}\,\rceil - 1) > 0 \quad (k+1 < p),\end{aligned}$$

which proves (57). Thus (56) and (57) are proved, which implies (55), hence (54). This completes the proof. □

As a corollary of the above theorem, we have the following:

COROLLARY 2. *For any integer p_0 satisfying $p_0 \ge 2$, n ($\in \mathbf{Z}$) and k ($\in \mathbf{Z}$) exist such that $p_0 = p(n,k)$.*

Proof. First, we will prove the corollary for the case $p_0 = 2$. Since function

$$\frac{\partial \tilde{\Omega}}{\partial p}(1,1,p) = \frac{8p^3 + 21p^2 + 18p + 2}{2(p+1)^2} \qquad (69)$$

has no real zeros except at $p = -0.129793$, $\tilde{\Omega}(1,1,p)$ $(p \ge 2)$ is a monotonically increasing function of p, which implies that

$$\tilde{\Omega}(1,1,3) \le \tilde{\Omega}(1,1,p) \quad (p \ge 3). \qquad (70)$$

Since we have

$$18 = \Omega(1,1,2) < \tilde{\Omega}(1,1,3) = \frac{195}{8} \qquad (71)$$

and (26), (70) implies that

$$\Omega(1,1,2) < \Omega(1,1,p) \quad (p \ge 3). \qquad (72)$$

This indicates that $2 = p(1,1)$, and the corollary is proved for the case $p_0 = 2$. When $p_0 > 2$, let n and k be large enough integers and $(p_0 - 1)$, respectively. □

Theorem 4 and Corollary 2 imply that the following properties of $\Omega(n,k,p)$.

(B1) When n is large enough, as a function of p, $\Omega(n,k,p)$ takes the minimum at $p = k+1$.

(B2) For any convergence degree p_0, n and k exist such that p_0 is optimal in terms of complexity.

From observations (A1) and (A2) in the previous subsection, $\tilde{\Omega}(n,k,p)$ is convex as a function of p for any n (≥ 3) and k (≥ 1), and the value of $\tilde{p}(n,k)$ tends to increase as n increases (compare Fig. 2, where $\Omega(1000,30,p)$ and $\tilde{\Omega}(1000,30,p)$ are plotted, with Fig. 1). Thus, as the asymptotic behavior of $\Omega(n,k,p)$, we have the following:

(B3) When n increases, $\Omega(n,k,p)$ tends to take a lower value with a higher value of p.

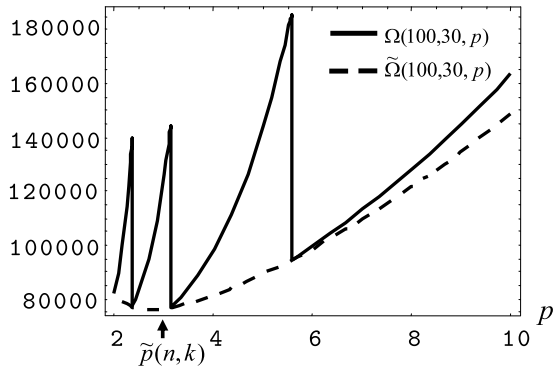

Fig. 1. $\Omega(100,30,p)$ and $\tilde{\Omega}(100,30,p)$

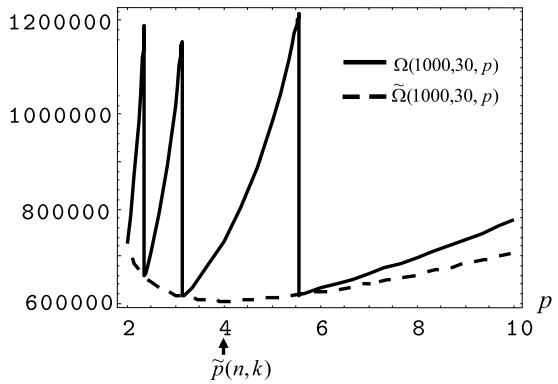

Fig. 2. $\Omega(1000,30,p)$ and $\tilde{\Omega}(1000,30,p)$

6.5.1. Characteristics of Algorithm 2

From observations (B1), (B2) and (B3), we expect the following characteristics of Algorithm 2 (below n and k denote the degree of $f(x, y)$ with respect to x and the truncation degree of power series roots, respectively):

(C1) When n is large enough compared with k, the optimal convergence degree is $(k + 1)$.

(C2) Any convergence degree p is optimal in terms of its complexity for some values of n and k.

(C3) When k is fixed and n becomes larger, a high degree convergence formula (i.e., one with a large p) is advantageous.

7. Numerical experiments

To confirm the validity of the observations in the previous section, we performed numerical experiments on a machine with a Pentium M 1.7 GHz processor and 512 MB of memory. All programs were implemented with *Mathematica* 5.0.

7.1. Power series arithmetic in *Mathematica*

To check the properties of power series arithmetic in *Mathematica*, we performed the following numerical experiments:

- Multiplication test: Compute $f_1(y) \times f_2(y)$, where $f_1(y)$, $f_2(y)$ are randomly generated power series whose degrees are k.
- Division test: Compute $f_1(y)/f_2(y)$, where $f_1(y)$, $f_2(y)$ are randomly generated power series whose degrees are k.

Each test was performed 100 times, and the average computation time was recorded. The results are shown in Table 1 and Table 2, where units are in milliseconds. The timing data in Table 1 and Table 2 can be approximated quite well by functions

$$h_1(k) = 0.00113194 \times \frac{(k+1)(k+2)}{2}, \quad h_2(k) = 0.00143496 \times \frac{(k+1)(k+2)}{2},$$

respectively (see Fig. 3 and Fig. 4, where the dots denote the results of the multiplication and division tests). The discussion and assumptions in Section 3 are thus confirmed as valid in *Mathematica*.

Table 1. Result of multiplication test (in milliseconds)

k	30	60	90	120	150	180	210	240	270	300
CPU Time	0.6	2.1	4.71	8.31	12.82	18.53	25.14	33.05	41.76	51.57

Table 2. Result of division test (in milliseconds)

k	30	60	90	120	150	180	210	240	270	300
CPU Time	0.8	2.8	6.01	10.72	16.42	23.64	32.04	41.76	52.87	65.3

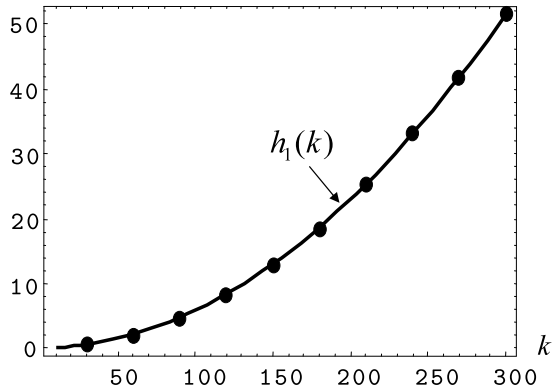

Fig. 3. Result of Multiplication test

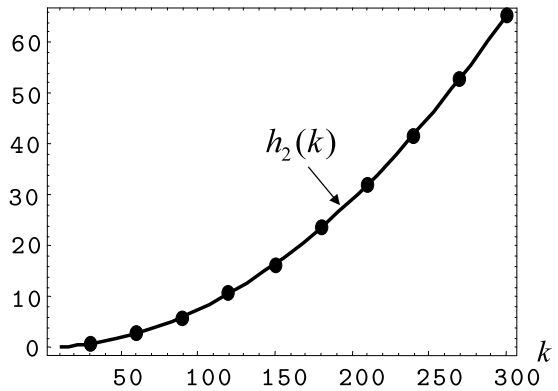

Fig. 4. Result of Divison test

7.2. Problem formulation

The two experiments outlined below were performed, where k, n and p denote the truncation degree of the power series roots, the degree of $f(x, y)$ with respect to x and the degree of convergence, respectively.

- Experiment A: $n = 100$, $k = 15$, $p = 2, \ldots, 11$.
- Experiment B: $n = 300$, $k = 15$, $p = 2, \ldots, 18$.

In each experiment, power series roots of polynomials in the form of

$$f(x, y) = \sum_{i=0}^{n} \sum_{j=0}^{15} c_{i,j} x^i y^j, \tag{73}$$

where $c_{i,j}$ are randomly generated integers satisfying $-10 \leq c_{i,j} \leq 10$, were computed using Algorithm 2, and the average computation time (out of 10 trials) was recorded.

7.3. Results of Experiment A

The results are shown in Table 3 and plotted in Fig. 5, where the dots (representing the data from Experiment A) are connected with straight lines. For the reference, $\Omega(100, 15, p)$ $(p = 2, \ldots, 11)$ is also plotted in Fig. 6 with dots connected with straight lines. We also plot $\tilde{\Omega}(100, 15, p)$ in Fig. 6 with dashed lines. From Fig. 5 and Fig. 6, we see that the tendency of the results of Experiment A closely match those of the complexity analysis (i.e., $\Omega(100, 15, p)$) except at $p = 2$. Both results indicate that when $(n, k) = (100, 15)$, Algorithm 2 is the most efficient at $p = 4$ (we have $\Omega(100, 15, 2) = 22755 > \Omega(100, 15, 4) = 22338$).

Table 3. Result of Experiment A (in seconds)

p	2	3	4	5	6	7	8	9	10	11
CPU Time	1.52	1.10	0.749	0.763	0.780	0.795	0.812	0.830	0.856	0.8864

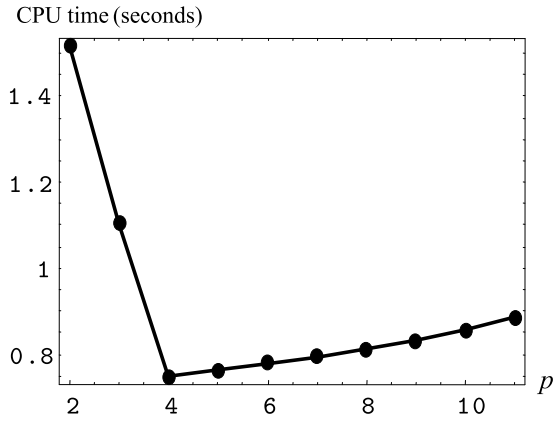

Fig. 5. Result of Experiment A

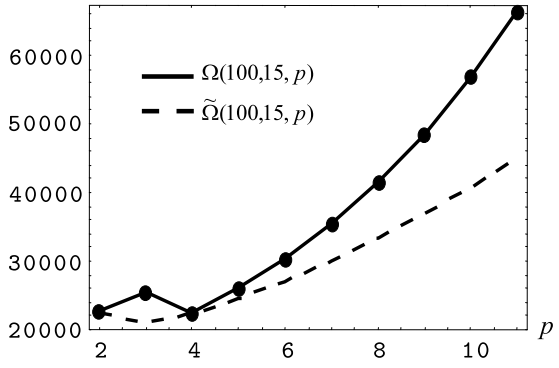

Fig. 6. $\Omega(100, 15, p)$ and $\tilde{\Omega}(100, 15, p)$

7.4. Results of Experiment B

The results of Experiment B are shown in Table 4 and Fig. 7, from which we see that Algorithm 2 is the most efficient at $p = 16$ $(= k + 1)$ in this case. For the reference, we plot $\Omega(300, 15, p)$ $(p = 2, \ldots, 18)$ in Fig. 8 with dots connected by straight lines. We also plot $\tilde{\Omega}(300, 15, p)$ in Fig. 8 with dashed lines.

Although the optimal p (p such that the cost function takes the minimum) for $\Omega(300, 15, p)$, i.e., $p = 4$, is different from that of Experiment B (where it is $p = 16$), the tendency of the result from Experiment B closely matches that of

Table 4. Result of Experiment B (in seconds)

p	2	3	4	5	6	7	8	9	10	11
CPU Time	4.83	3.56	2.43	2.48	2.53	2.58	2.63	2.69	2.74	2.80

p	12	13	14	15	16	17	18			
CPU Time	2.86	2.92	2.98	3.04	1.56	1.58	1.62			

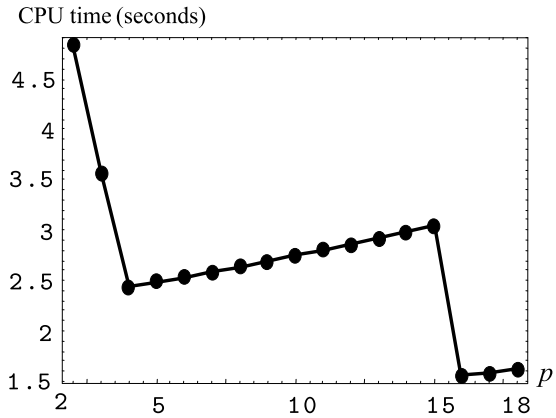

Fig. 7. Result of Experiment B

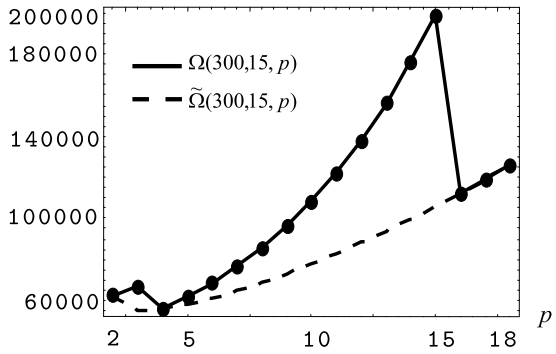

Fig. 8. $\Omega(300, 15, p)$ and $\tilde{\Omega}(300, 15, p)$

$\Omega(300, 15, p)$ (the cost function takes local minima at $p = 4$ and $p = 16$ in both of Fig. 7 and Fig. 8).

From Theorem 4 and characteristic (C1), when n is large enough, the optimal p should be $(k+1)$. In fact, when $n = 10^5$, function $\Omega(n, 15, p)$ takes the minimum at $p = 16$ (see Fig. 9), and $n = 300$ is just not large enough.

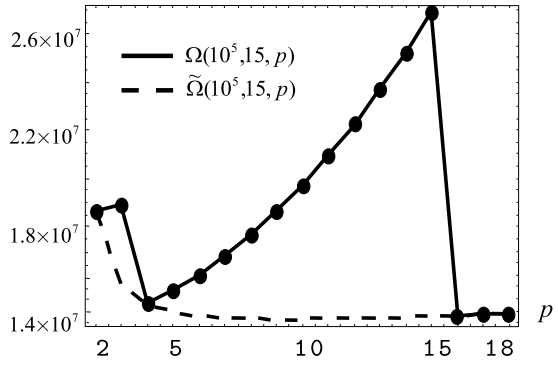

Fig. 9. $\Omega(10^5, 15, p)$ and $\tilde{\Omega}(10^5, 15, p)$

7.5. Observations

From Experiments A and B, we see the following:

(i) The results of Experiments A and B closely match those of complexity analysis $\Omega(n, k, p)$ in their tendencies, and have the same local minima.

(ii) Although Theorem 4 tells us that the optimal p is $(k + 1)$ when n is large enough, the required scale of n might be quite large.

(iii) Compared with $\Omega(n, k, p)$ (the result of complexity analysis), the results of Experiments A and B have the tendencies that are more advantageous for a high degree convergence formula. For example, when $(n, k) = (300, 15)$, the optimal p is $p = 16$ in the experiments, while it is $p = 4$ for $\Omega(n, k, p)$.

One of the reasons for (iii) above is that the experiments were performed in *Mathematica*. This is an interpreter-type programming language whose running speed is significantly lower than that of a compiler-type language. The influence of the number of program steps can therefore not be ignored (we only count the number of multiplications/divisions in complexity analysis), which makes a high degree convergent formula more advantageous due to the lower number of program steps.

8. Conclusions

We proposed an algorithm to compute the power series root of a given bivariate polynomial $f(x, y)$. The algorithm, whose convergence degree is arbitrary, is a combination of the polynomial evaluation algorithm in [9] and the arbitrary degree convergent formula in [1]. We performed complexity analysis on the algorithm, counting the number of numerical multiplications/divisions required to compute a power series root, and presented the number as a function of n (the degree of

given polynomial $f(x, y)$ with respect to x), k (the truncation degree of the power series root) and p (the degree of convergence) in an explicit form. We analyzed the function $\Omega(n, k, p)$ and derived some of the properties of Algorithm 2 in terms of its complexity, which indicates the following:

- When n (the degree of given polynomial $f(x, y)$ with respect to x) is large enough, the optimal convergence degree in terms of its complexity is $(k + 1)$, where k is the truncation degree of the power series roots.
- Any convergence degree p is optimal in terms of its complexity for some degree n of $f(x, y)$ with respect to x and truncation degree k of the power series roots.
- When n (the degree of given polynomial $f(x, y)$ with respect to x) is large enough, the high degree convergent algorithm outlined in this paper is advantageous over the symbolic Newton's method.

We performed two numerical experiments that confirmed the validity of the above claims. As with the power series roots in this paper, we can define power series eigenvalues as power series expansions of the eigenvalues of a matrix with polynomial entries (see [4] for details). Reference [5] proposes algorithms to compute these power series eigenvalues with arbitrary degree p ($\in \mathbf{Z}$) of convergence. In future work, we would like to analyze the algorithm and examine its characteristics.

References

[1] T. Kitamoto, Hensel construction with an arbitrary degree of convergence. Japan J. Indust. Appl. Math., **13** (1996), 203–215.

[2] T. Kitamoto, On efficient computation of approximate roots (in Japanese). Trans. IEICE, **J85-A** (2002), 189–196.

[3] T. Kitamoto, On extension of symbolic Newton's method to the formula with high degree of convergence (in Japanese). Trans. IEICE, **J84-A** (2001), 983–988.

[4] T. Kitamoto, Approximate Eigenvalues, Eigenvectors and Inverse of a Matrix with Polynomial Entries. Japan J. Indust. Appl. Math., **11** (1994), 75–85.

[5] T. Kitamoto, On Computation of Approximate Eigenvalues and Eigenvectors. Trans. IEICE, **E85-A** (2002), 664–675.

[6] T. Kitamoto, Accurate Computation of a High Degree Coefficient of a Power Series Root. Trans. IEICE, **E88-A** (2005), 718–727.

[7] H.T. Kung and J.F. Traub, All algebraic functions can be computed fast. J. ACM, **25** (1978), 245–260.

[8] J.D. Lipson, Newton's method: A great algebraic algorithm. ACM Symposium on Symbolic and Algebraic Computations Proceedings, 1976, 260–270.

[9] M. Shaw and J.F. Traub, On the number of multiplications for the evaluation of a polynomial and some of its derivatives. J. ACM, **21** (1974), 161–167.