Improved Computation of Bounds for Positive Roots of Polynomials

Masami Takata[1], Takuto Akiyama[2], Sho Araki[3], Kinji Kimura[4], and Yoshimasa Nakamura[5]
[1]Academic Group of Information and Computer Sciences, Nara Women's University,
Kita-Uoya-Nishi-Machi, Nara 630-8506, JAPAN
[2345]Graduate School of Informatics, Kyoto University,
Yoshidahonmachi, Sakyo-ku, Kyoto 606-8501, JAPAN
[1]takata@ics.nara-wu.ac.jp, [2]akiyama@amp.i.kyoto-u.ac.jp, [3]araki@amp.i.kyoto-u.ac.jp,
[4]kkimur@amp.i.kyoto-u.ac.jp, [5]ynaka@i.kyoto-u.ac.jp

Abstract— *A new lower bound for computing positive roots of polynomial equations is proposed. We discuss a two-stage algorithm for computing positive roots of polynomial equations. We employ the new bound to accelerate the continued fraction method based on Vincent's theorem. Finally, we conduct experiments to evaluate the effectiveness of the proposed lower bound.*

(This paper is submitted to PDPTA'13.)

Keywords: continued fraction method, Vincent's theorem, local-max bound, Newton's method, Laguerre's theorem

1. Introduction

The real roots of univariate polynomial equations are more useful than the imaginary roots for practical applications in various engineering fields. Thus, the objective of this study is the computation of all real roots of polynomial equations. For this purpose, we develop a real-root isolation algorithm. For polynomial equations without multiple roots, each root can be isolated into a numeric interval. Then, the accuracy of the isolated real roots can be easily enhanced by using a bisection method.

The continued fraction method for isolating the positive roots of univariate polynomial equations is based on Vincent's theorem [2], [10]. In this method, each positive root is isolated using Descartes' rule of signs [3], which focuses on the coefficients of the polynomial equations. The execution of Descartes' rule of signs requires origin shifts.

To accelerate the continued fraction method based on Vincent's theorem, the lower bound of the smallest positive root is required. In general, to obtain the lower bound of positive roots of a polynomial equation, we first substitute $1/x$ for x in the polynomial equation $f(x)$. Second, we compute the upper bound of the positive roots. Third, we obtain the lower bound by computing the inverse of the upper bound. The Cauchy bound [9] and the Kioustelidis bound [6] are known as upper bounds of the positive roots of polynomial equations. Akritas *et al.* introduced a generalized theorem including the Cauchy bound and the Kioustelidis bound [1]. Then, by specializing this generalized theorem, they proposed a new upper bound called the local-max bound, which is different from both the Cauchy bound and the Kioustelidis bound.

In this paper, we propose a new lower bound for accelerating the continued fraction method based on Vincent's theorem.

2. Positive Roots of Polynomials

To compute the positive roots of a polynomial equation

$$\begin{aligned} f(x) &= a_0x^n + a_1x^{n-1} + \cdots + a_{n-1}x + a_n = 0, \quad (1) \\ & x \in \mathbb{R}, a_i \in \mathbb{Z}, \end{aligned}$$

in the interval $x \in (0, \infty)$, we first isolate each root into a numeric interval. Second, we improve the accuracy of the real roots by using a bisection method. Here, the intervals are defined by,

$$x \in [a, b],\ x \in (a, b] \text{ or } x \in [a, b), a, b \in \mathbb{R}, a \le b, \qquad (2)$$

where $[,]$, $(,]$ or $[,)$ denote a closed interval, a left-open right-closed interval, and a left-closed right-open interval, respectively.

3. Continued Fraction Method based Vincent's Theorem for Isolating Positive Roots

3.1 Concept

In the continued fraction method based on Vincent's theorem, real roots in $(0, \infty)$ can be isolated using the Descartes' rule of signs.

Descartes' rule of signs is derived from the following theorem.

Theorem 1 (The Descartes' rule of signs): In a polynomial equation

$$\begin{aligned} f(x) &= a_0x^n + \cdots + a_{n-1}x + a_n = 0, \\ & x \in \mathbb{R}, \end{aligned}$$

with real coefficients,

W := the number of "changes of sign" in the list of coefficients$\{a_0, a_1, \ldots, a_n\}$ except $a_i = 0$,

N := the number of positive roots in $(0, \infty)$

are defined. Under these definitions, we have,

$$N = W - 2h.$$

Here, h is a non-negative integer.

By using Theorem 1, the number of positive roots of the polynomial equation $f(x)$ is determined in the following conditional branch:

- In the case that $W = 0$, $f(x), x \in (0, \infty)$ does not have any positive roots.
- In the case that $W = 1$, $f(x)$ has only one positive root in the interval $x \in (0, \infty)$.
- In the case that $W \geq 2$, the number of positive roots of $f(x)$ cannot be determined.

In the case that $W = 1$, the isolated interval should be set to $(0, ub]$, where ub denotes the upper bound of positive roots for a polynomial equation $f(x)$. Computation methods for the upper bound of positive roots of a polynomial equation $f(x)$ are described in Section 3.2.

In the case that $W \geq 2$, we divide the interval $(0, \infty)$ in the two intervals. Then, Descartes' rule of signs is applied in each interval. In the continued fraction method based on Vincent's theorem, the interval $(0, \infty)$ is divided in $(0, 1)$ and $(1, \infty)$. The division is performed by the replacement

$$x \rightarrow x + 1,$$
$$x \rightarrow \frac{1}{x+1}.$$

By using the replacement $x \rightarrow x + 1$, the interval $(0, \infty)$ of the replaced polynomial equation corresponds to the interval $(1, \infty)$ of the original polynomial equation. Similarly, by using the replacement $x \rightarrow 1/(x+1)$, the interval $(0, \infty)$ of the replaced polynomial equation corresponds to the interval $(0, 1)$ of the original polynomial equation. The intervals $(1, \infty)$ and $(0, 1)$ do not include the case that $x = 1$. To solve for this case, after either replacement, we check whether the coefficient a_n, which is a constant term, vanishes. If $a_n = 0$ in the replaced polynomial equation, then 1 is a root of the original polynomial equation.

3.2 Computation for Upper Bound

Akritas et al. introduced the local-max pairing strategy (defined in Definition 1) in order to generate a suitable bound.

Definition 1 ("local-max"): For a polynomial equation $f(x)$ given by eq. (1), the coefficient $-a_k$ of the term $-a_k x^{n-k}$ in $f(x)$ is paired with the coefficient $a_m/2^l$ of the term $a_m x^{n-m}$, where a_m is the largest positive coefficient with $0 \leq m < k$ and t denotes the number of times the coefficient a_m has been used.

Algorithm 1 Implementation of the "local-max" bound.

```
cl ← {a_n, a_{n-1}, ···, a_1, a_0}
if n + 1 ≤ 1 then
  return ub_3 = 0
end if
j = n + 1
t = 1
for i = n to 1 step −1 do
  if cl(i) < 0 then
    tempub = (2^t(−cl(i)/cl(j)))^{1/(j−i)}
    if tempub > ub then
      ub = tempub
    end if
    t++
  else if cl(i) > cl(j) then
    j = i
    t = 1
  end if
end for
ub_3 = ub
```

The implementation of the local-max bound is described in Algorithm1, and the output is ub_3.

3.3 Acceleration using Lower Bound

The continued fraction method based on Vincent's theorem requires many replacement operations $x \rightarrow x + 1$ and $x \rightarrow 1/(x+1)$. In other words, the origin shift is realized by $x \rightarrow x + 1$. Thus, if the positive roots are much larger than 1, then the computation time increases, as we must repeat the replacement operation $x \rightarrow x+1$. To decrease the computation time, the lower bound of the smallest positive root of a polynomial equation should be used as a shift.

In general, to obtain the lower bound lb of an original polynomial equation, we first substitute $1/x$ for x in the original polynomial equation. Second, we compute the upper bound ub_3 of the positive roots. Third, we obtain the lower bound lb by computing the inverse of the upper bound as follows:

$$lb = \frac{1}{ub_3}. \quad (3)$$

If $lb > 1$, then the replacement $x \rightarrow x + lb$ is adopted, as the computation time for isolating the positive roots decrease. If $lb \leq 1$, then we do not adopt the lower bound lb, as the lower bound lb is not sufficiently large to reduce the computation time.

Algorithm 2 shows a continued fraction method based on Vincent's theorem with origin shift using the local-max

Algorithm 2 Continued fraction method based on Vincent's theorem with the local-max shift strategy.

```
R ← φ
S ← {poly}
if 0 is a solution of poly then
  R ← R ∪ [0, 0]
  poly ← poly/x
end if
while S ≠ φ do
  poly ← dequeue(S)
  W ← Descartes(poly)
  if W = 1 then
    ub_3 ← Algorithm1 with poly
    R ← R ∪ Inverse Möbius trans ((0, ub_3])
  else if W ≥ 2 then
    poly2 ← Trans(poly, x → 1/x)
    ub_3 ← Algorithm1 with poly2
    lb ← 1/ub_3
    if lb > 1 then
      poly ← Trans(poly, x → x + lb)
      if 0 is a solution of poly then
        R ← R ∪ Inverse Möbius trans ([lb, lb])
        poly ← poly/x
      end if
    end if
    poly3 ← Trans(poly, x → x + 1)
    if 0 is a solution of poly3 then
      R ← R ∪ Inverse Möbius trans ([1, 1])
      poly3 ← poly3/x
    end if
    poly4 ← Trans(poly, x → 1/x + 1)
    S ← S ∪ {poly3, poly4}
  end if
end while
```

bound. The computation time for the Algorithm 2 is less than that for the continued fraction method based on Vincent's theorem without the origin shift. The replacements $x \to x+1$ and $x \to 1/(x+1)$ are called Möbius transformations. After the intervals for isolating the positive roots of a polynomial equation are determined, each interval should be replaced by the interval processed by all inverse transformations of Möbius transformations.

4. New Lower Bound

The acceleration of the continued fraction method based on Vincent's theorem employs the origin shift, which adopts the lower bound lb of the smallest positive root of a given polynomial equation. Thus, if the lower bound tends to the smallest positive root, then the computation time of the continued fraction method decreases.

In this paper, we propose a new lower bound generated by Newton's method. Note that in some polynomial equations, a bound generated by Newton's method is not suitable as the lower bound. Hence, by using Laguerre's theorem [7], it must be checked whether a bound generated by Newton's method is a suitable lower bound.

Newton's method is defined by the following recurrence formula:

$$x_{m+1} = x_m - \frac{f(x_m)}{f'(x_m)}. \tag{4}$$

Here, $f'(x)$ denotes the first derivative of $f(x)$. If Newton's method is adopted at the origin, then a candidate for the lower bound r is computed as follows:

$$r = 0 - \frac{f(0)}{f'(0)} = -\frac{a_n}{a_{n-1}}. \tag{5}$$

The cost for computing r is $O(1)$.

We can check whether a candidate for the lower bound r is suitable by using the Laguerre theorem. The improved algorithm of the continued fraction method based on Vincent's theorem with the shift strategy, including both the local-max bound and the new lower bound generated by Newton's method, is shown in Algorithm 3.

5. Experiment

In this section, we conduct experiments to evaluate the effectiveness of the proposed lower bound.

Here, Algorithm 2 and Algorithm 3 are compared.

As test polynomial equations, we use $f(x)$ with integer coefficients:

$$\begin{aligned} f(x) &= \prod_{i=0}^{r}(x - x_i) \times \\ &\quad \prod_{j=0}^{s}(x - \alpha_j + i\beta_j)(x - \alpha_j - i\beta_j), \qquad (6) \\ &\quad x_i, \alpha_j, \beta_j \in \mathbb{R}. \end{aligned}$$

Here, parameters x_i, α_j, and β_j are randomly set as follows:

$$-10000 \le x_i, \alpha_j, \beta_j \le 10000, \tag{7}$$

Parameters s and r are set to $s = 490, r = 20$. Then, we generate 100 test polynomial equations.

In the continued fraction method based on Vincent's theorem, the multiple-precision arithmetic library GMP [4] is needed to compute all coefficients in replaced polynomial equations.

Figure 1 shows the plots of the computation time in all test polynomial equations. In Figure 1, the computation time for Algorithm 3 is less than that for Algorithm 2, and the difference among the computation time in Algorithm 3 is small.

Table 1 shows the computation time for the 100 random polynomial equations. The maximum computation time for Algorithm 3 is 1.48 times faster than that for Algorithm 2. The average computation time for Algorithm 3 is 1.09 times

Algorithm 3 Improvement of the continued fraction method based on Vincent's theorem with the shift strategy including both the local-max bound and the new lower bound generated by Newton's method.

$R \leftarrow \phi$
$S \leftarrow \{poly\}$
if 0 is a solution of $poly$ **then**
  $R \leftarrow R \cup [0,0]$
  $poly \leftarrow poly/x$
end if
while $S \neq \phi$ **do**
  $poly \leftarrow dequeue(S)$
  $W \leftarrow Descartes(poly)$
  if $W = 1$ **then**
    $ub \leftarrow$ Algorithm 1 with $poly$
    $R \leftarrow R \cup$ Inverse Möbius trans $((0, ub])$
  else if $W \geq 2$ **then**
    $poly2 \leftarrow Trans(poly, x \rightarrow 1/x)$
    $ub_3 \leftarrow$ Algorithm 1 with $poly2$
    $r \leftarrow NewtonLowerbound(poly)$ which is checked by using the Laguerre theorem
    $lb \leftarrow \max(1/ub_3, r)$
    if $lb > 1$ **then**
      $poly \leftarrow Trans(poly, x \rightarrow x + lb)$
      if 0 is a solution of $poly$ **then**
        $R \leftarrow R \cup$ Inverse Möbius trans $([lb, lb])$
        $poly \leftarrow poly/x$
      end if
    end if
    $poly3 \leftarrow Trans(poly, x \rightarrow x + 1)$
    if 0 is a solution of $poly3$ **then**
      $R \leftarrow R \cup$ Inverse Möbius trans $([1, 1])$
      $poly3 \leftarrow poly3/x$
    end if
    $poly4 \leftarrow Trans(poly, x \rightarrow 1/x + 1)$
    $S \leftarrow S \cup \{poly3, poly4\}$
  end if
end while

faster than that for Algorithm 2. The standard deviations in Algorithm 2 and Algorithm 3 are not considerably large.

From Figure 1 and Table 1, the computation time for Algorithm 3 is less than that for Algorithm 2. This is because some lower bounds generated from Newton's method are more suitable than the local-max bound. Consequently, the continued fraction method based on Vincent's theorem is improved by using the proposed lower bound.

Hence, the improved continued fraction method with the local-max bound and the proposed lower bound generated by Newton's method is efficient.

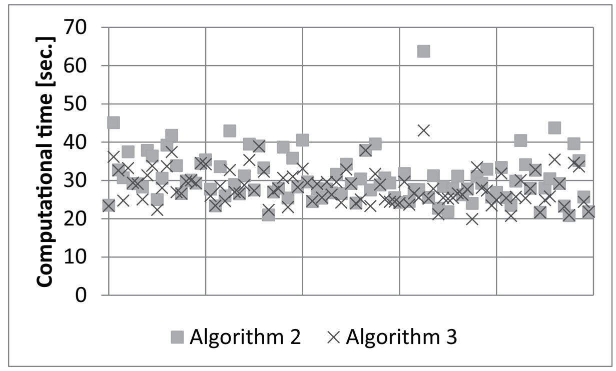

Fig. 1: Computation time in all test polynomial equations. [sec.]

Table 1: Computation time.

	average [sec.]	deviation	max.[sec.]	min.[sec.]
Algorithm 2	30.45	6.45	68.74	20.76
Algorithm 3	28.05	4.46	43.04	19.89

6. Conclusions

In this paper, we proposed a new lower bound for accelerating the continued fraction method based on Vincent's theorem.

In the future, the proposed lower bound should be evaluated using different types of test polynomials from (7).

References

[1] A. Akritas, A. Strzeboński, P. Vigklas, "Implementations of a New Theorem for Computing Bounds for Positive Roots of Polynomials," Computing, 78, pp. 355–367, 2006.
[2] A. Akritas, A. Strzeboński, P. Vigklas, "Improving the performance of the continued fractions method using new bounds of positive roots". Nonlinear Analysis: Modelling and Control, 13, pp. 265–279, 2008.
[3] G. Collins, A. Akritas, "Polynomial Real Root Isolation Using Descartes' Rule of Signs", SYMSAC '76, Proceedings of the third ACM symposium on Symbolic and algebraic computation, Yorktown Heights, NY, USA, ACM, pp. 272-275, 1976.
[4] (2013) The GNU MP Bignum Library. [Online]. Available: http://gmplib.org/
[5] P. Henrich, "Applied and computational complex analysis", Wiley Classics Library Edition, Volume I, 1988.
[6] B. Kioustelidis, "Bounds for positive roots of polynomials,", J. Comput. Appl. Math., 16(2), pp. 241–244, 1986.
[7] E. Laguerre, "On the Theory of Numeric Equations", Journal de Mathématique pures et appliquées, 3^e série, t. IX, 1883. Translated by Stewart A. Levin, available from http://sepwww.stanford.edu/oldsep/stew/laguerre.pdf
[8] R.E. Moore, "Interval Analysis". Prentice Hall, Englewood Cliffs, N.J., 1966
[9] N. Obreschkoff, "Verteilung und Berechnung der Nullstellen reeller Polynome", Berlin: VEB Deutscher Verlag der Wissenschaften 1963.
[10] A.J.H. Vincent, "Sur la resolution des équations numériques, " J. Math. Pures Appl. 1, pp. 341–372, 1836.