The Comparison of Numerical Methods for Solving Polynomial Equations

By Aurél Galántai

Abstract. In this paper we compare the Turán process [5]–[6] with the Lehmer-Schur method [2]. We prove that the latter is better.

1. The Algorithms. We first describe the Turán process [5]–[6] which can be considered as an improvement of Graeffe's method. For the complex polynomial

\[ p_0(z) = \sum_{j=0}^{n} a_j z^j = 0 \quad (a_j \in \mathbb{C}, a_{00} a_{n0} \neq 0), \]

the method can be formulated as follows.

Let

\[ p_j(z) = p_{j-1}(\sqrt{z})p_{j-1}(-\sqrt{z}) = \sum_{k=0}^{n} a_{kj} z^k \quad (j = 1, 2, \ldots) \]

be the \( j \)th Graeffe transformation and let

\[ M[p_0(z), m_0] = \left[ \max_{1 \leq k \leq n} \left| \frac{\sigma_k}{n} \mu_{0/k} \right| \right]^{-1}, \]

where \( \mu_0 = 2^{-m_0}, \sigma_0 = 0, \)

\[ \sigma_k = \left[ k a_k m_0 - \sum_{j=1}^{k-1} a_j m_0 \sigma_{k-j} \right] / a_{0m_0} \quad (k = 1, \ldots, n) \]

and \( m_0 \geq 1 \) is fixed.

Let the constants \( \alpha_{m_0}, l \) be defined by the inequalities

\[ 0.5 < \alpha_{m_0} < 5^{-\mu_0}, \quad l > \pi \left[ \arccos \frac{2.5 + \alpha_{m_0}}{2 + 2\alpha_{m_0}} \right]^{-1} - 1, \quad m_0 \geq 2. \]

Then with the notations

\[ M^{(0)} = M[p_0(z), m_0], \quad S^{(0)} = 0, \]

the \( d \)th step of the algorithm is the following:

1. Algorithm (T). (i) Let

\[ S_{j}^{(d+1)} = S^{(d)} + 0.5(1 + \alpha_{m_0})M^{(d)} \exp \left( \frac{2\pi i}{l + 1} \right), \]

where \( j = 0, 1, \ldots, l \) and \( i = \sqrt{-1}. \)
(ii) If there exists an index $j$ such that $p_0(S_j^{(d+1)}) = 0$, then we get a root and the process terminates.

(iii) Let us compute the quantities

$$M_j^{(d+1)} = M[p_0(z + S_j^{(d+1)}), m_0] \quad (j = 0, 1, \ldots, l)$$

and let

$$M^{(d+1)} = \min_j M_j^{(d+1)} = M_j^{(d+1)}, \quad S^{(d+1)} = S_j^{(d+1)}.$$ 

Turán [5] proved that $S^{(d)}$ tends to a root of $p_0(z)$, and the convergence is linear. Turán [5] also proved that the number of iterations needed to achieve an arbitrary relative error $e (> 0)$ is independent of $p_0(z)$ and depends on degree $p_0(z)$ only.

Our purpose is to answer the remarks of the last section of [6]. For this reason we compare the Turán process with the Lehmer-Schur method which is often applied in practice ([2], [3], [4]). This algorithm can be described as follows.

Let

$$T[p_0(z)] = \sum_{j=0}^{n-1} \frac{(-1)^j}{j!} \frac{a_0 a_j - a_n a_{n-j}}{a_0} z^j$$

(1.6)

and

$$T'[p_0(z)] = T[T^{p-1}[p_0(z)]], \quad (j = 2, \ldots).$$

(1.7)

Let us compute the numbers $c_j = T'[p_0(0)]$, $(j = 1, \ldots, k)$, where

$$k = \min \{m \in \mathbb{N} | c_m = 0\}. \quad (1.8)$$

Here, $\mathbb{N}$ denotes the set of nonnegative integers. With the aid of the sequence $\{c_j\}_{j=1}^k$ we define the function $N[p_0(z)]$ as follows

$$N[p_0(z)] = \begin{cases} 1 & \text{if } \exists j \in \{1, \ldots, k-1\} \text{ such that } c_j < 0, \\ 0 & \text{if } c_j > 0 (j = 1, \ldots, k-1) \text{ and degree } T^{k-1}[p_0(z)] = 0, \\ -1 & \text{otherwise}. \end{cases}$$

(1.9)

Lehmer [2] proved that if $N[p_0(z)] = 1$ then the polynomial $p_0(z)$ has a root in $\{z \in \mathbb{C} | |z| < 1\}$, if $N[p_0(z)] = 0$ then $p_0(z)$ has no roots in this set. We shall return to the case $N[p_0(z)] = -1$.

Let us introduce the notations

$$\alpha^{(d)}_j = \begin{cases} 0.5 \gamma_0^{(d)} R^{(d-1)} & (j = 0), \\ 0.4 \gamma_0^{(d)} R^{(d-1)} & (j = 1, \ldots, 8), \end{cases}$$

(1.10) 

and

$$\beta^{(d)}_j = \begin{cases} z^{(d-1)} & (j = 0), \\ z^{(d-1)} + \frac{0.75 R^{(d-1)}}{\cos \frac{\pi}{8}} \exp \left( \frac{2 \pi i (j-1)}{8} \right) & (j = 1, \ldots, 8), \end{cases}$$

(1.11)


where the sequences \( \{R^{(d)}\} \), \( \{z^{(d)}\} \) and \( \{\gamma^{(d)}\} \) are defined by the \( d \)th step of the Lehmer-Schur method \((d = 1, \ldots)\). Let \( p_0(z) = p_0(\psi) \) \((\psi > 0)\) and

\[
(1.11) \quad z^{(0)} = 0; \quad R^{(0)} = 1 + \max_i \left| \frac{d_{j0}}{a_{n0}} \right|.
\]

Then the \( d \)th step can be written as follows.

2. **Algorithm (L).** (i) If there exists an index \( j \) such that \( p_0(\beta^{(d)}_j) = 0 \), then we get a root and the process terminates.

(ii) We choose the index \( j \in \{0, 1, \ldots, 8\} \) such that

\[
N[p_0(\alpha^{(d)}_j z + \beta^{(d)}_j)] = 1
\]

and let

\[
z^{(d)} = \beta^{(d)}_j, \quad R^{(d)} = \alpha^{(d)}_j.
\]

The numbers \( \gamma^{(d)} \in [1, 1 + \delta] \), \((\delta \leq 0.5)\) are chosen such that \( N[p_0(\alpha^{(d)}_j z + \beta^{(d)}_j)] \geq 0 \) will be satisfied (except in unusual circumstances \( \gamma^{(d)}_j = 1 \) can be chosen). Lehmer [2] proved that process converges linearly. The number of iteration steps needed to achieve an arbitrary absolute error \( \epsilon > 0 \) depends on \( p_0(z) \).

2. **The Limitations of the Algorithms.** Denote by \( Z \) the set of integers and let \( P_n \) be the set of complex polynomials of degree \( n \).

A numerical method \( M \) (iterative process) for solving \( p_0(z) = 0 \) where \( p_0(z) \in P_n \) can be identified with the sequence \( \{b_k\} \subset C \) which rises from the computation. This sequence depends on \( p_0(z) \) and will be denoted by \( \{M_{p_0}\} = \{b_k\} \). There exists a subsequence \( \{b_{k_j}\} \) of \( \{b_k\} \) such that

\[
(2.1) \quad z^* = \lim_{j \to \infty} b_{k_j} \quad \text{and} \quad p_0(z^*) = 0.
\]

A digital computer can perform elementary (complex) operations only over the finite set

\[
(2.2) \quad S[0, K] \cap C_{\delta},
\]

where \( S[0, K] = \{z \in C \mid |z| \leq K\} \) and

\[
(2.3) \quad C_{\delta} = \{z \in C \mid |z - k\delta + j\delta i: k, j \in Z\} \quad (\delta > 0).
\]

If there exists an element \( b_{k_0} \) in the sequence \( \{b_k\} \) such that \( |b_{k_0}| > K \), then the algorithm \( M \) cannot continue to run because of overflow.

In order to study the overflow we introduce the class of polynomials

\[
(2.4) \quad P_{M}(a, K, K^*) = \{p_0(z) \in P(a, K^*) \mid \{M_{p_0}\} \subset S[0, K], \quad |\{M_{p_0}\}| = \infty\},
\]

where

\[
(2.5) \quad P(a, K^*) = \{p_0(z) \in P_n \mid 0 < |z_j| \leq a \quad (j = 1, \ldots, n), \quad \|p_0(z)\| \leq K^*\}
\]

and
Here |{M_p_j}| denotes the cardinality of \{b_k\}, and \(z_j\) is the \(j\)th zero of \(p_0(z)\).

The set \(P_M(a, K, K^*)\) represents the class of all polynomials which can be solved by \(M\) in a bounded set.

The following statements are valid.

**Theorem 2.1.** The set \(P_M(a, K, K^*)\) defined by Algorithm 1 is empty for every \(a, K, K^* > 0\).

**Proof.** If the roots of \(p_0(z)\) are arranged so that

\[
|z_1| \geq |z_2| \geq \ldots \geq |z_n|,
\]

then the estimate

\[
5^{-\mu_0} \leq \frac{|z_n|}{M[p_0(z), m_0]} \leq 1
\]

is valid (see [5] - [6]). For this reason the convergence of Algorithm 1 is identical with

\[
|z_n^{(d)}| \leq cq^d \quad (c > 0, 0 < q < 1),
\]

where \(z_n^{(d)}\) is the zero of \(p_0(z + S^{(d)})\), \((d = 0, 1, \ldots)\) of minimal absolute value.

Using the inequality (2.8), we have

\[
\frac{n}{5e^r} \left(\frac{1}{q}\right)^{d'/\mu_0} \leq \frac{n}{5|z_n^{(d)}|^{1/\mu_0}} \leq |\sigma_{k(d)}^{(d')}| \quad (d \geq d')
\]

where \(k(d) \in \{1, \ldots, n\}\) is the index of the maximal element in (1.3) and

\[
\frac{n}{5e^r} \left(\frac{1}{q}\right)^{d'/\mu_0} > 1.
\]

Since \(|\sigma_{k(d)}^{(d')}| = O(w^d)\), where \(w = (1/q)^{1/\mu_0}\), therefore for a large index \(d_0\)

\[
|\sigma_{k(d)}^{(d')}| > K \quad (d \geq d_0)
\]

is satisfied. Thus the theorem is proved.

**Theorem 2.2.** If \(K \geq K^*2^{n+1}(1 + a^n2^n)^{n+1} + 1\), then

\[
P_L(a, K, K^*) = P(a, K^*)
\]

is satisfied for Algorithm 2.

**Proof.** It is easy to see that the quantities recurring in the algorithm satisfy the inequalities

\[
|p_0(\beta_j^{(d)})| \leq \left\{ \begin{array}{ll}
\|p_0\|2^{n+1} & (a < 0.5), \\
\|p_0\|(1 + 2^n a^n)^{n+1} & (a \geq 0.5),
\end{array} \right.
\]

\[
\|T^j[p_0(z)]\| \leq \frac{1}{4}(2\|p_0(z)\|)^{2^j} \quad (j = 1, \ldots, n)
\]
SOLVING POLYNOMIAL EQUATIONS

and

\[ \| p_0(x^{(d)}z + \beta^{(d)}) \| \leq \| p_0(z) \| (2 + 2^{n+1}a^n)^n \quad (j = 0, 1, \ldots, 8), \]

for \( d = 0, 1, \ldots \). With the notation

\[ \delta = \| p_0(z) \| (2 + 2^{n+1}a^n)^n, \]

and by using (2.15)–(2.16), we have

\[ \| T_k[p_0(x^{(d)}z + \beta^{(d)})] \| \leq \frac{\delta}{2^k} \quad (k = 1, \ldots, n). \]

Since \( K \) is greater than the right side of (2.14) and (2.16), using \( \psi > 2\delta \) we can get \( \bar{\delta} < 0.5 \) which proves the theorem.

The difference between Algorithms 1 and 2 is caused by the fact that Algorithm 1 is based on the inequality (2.8) while Algorithm 2 is based on the characteristic function \( N[p_0(z)] \) which is invariant for the mapping \( p_0(z) \rightarrow p_0(z)/\psi, (\psi > 0) \).

We remark that Algorithm 1 modified by the mappings

\[ p_0(z) \rightarrow p_0(z)/\psi, \quad p_0(z) \rightarrow p_0(z/\psi) \quad (0 < \psi \leq K) \]

also has a \( P_\gamma(a, K, K^*) \) empty for every \( a, K, K^* > 0 \).

3. The Study of Cost Functions. In the previous section it was proved that Algorithm 1 is unapplicable. Since an approximate solution with a given error \( e > 0 \) can be computed in the bounded set \( S[0, \bar{K}] \), where \( \bar{K} \) depends on \( p_0(z) \), \( e \), and the method \( M \), further analysis of the algorithms is necessary.

The cost function of the \( j \)th algorithm \( (j = 1, 2) \) is defined by the number of additions and multiplications per step and denoted by \( K_{m}^j \) and \( K_{a}^j \).

Assuming that the computing time of the \( k \)th root can be characterized by three additions and three multiplications (which is a rough underestimate), the cost function of Algorithm 1 is

\[ K_m^1 = (l + 1)(m_0 + 4)\frac{n^2}{2} + (l + 1)(m_0 + 8)\frac{n}{4} + O(1), \]

\[ K_a^1 = (l + 1)(m_0 + 4)\frac{n^2}{4} + (2l + 3)n + O(1). \]

For the cost function of Algorithm 2 the inequalities

\[ K_m^2 \leq 27n^2 - 18n, \]

\[ K_a^2 \leq 9n^2 + 36n, \]

hold.

If we identify the bounds (3.3)–(3.4) with the cost of one step, then the speed of Algorithm 2 is

\[ |z^{(d)} - z^*| \leq c_2(2/5)^d \quad (d = 0, 1, \ldots). \]

The speed of Algorithm 1 is

\[ |S^{(d)} - z^*| \leq c_1 [q(a, m_0, m_0, l)]^d \quad (d = 0, 1, \ldots), \]
where

\[ q(\alpha_{m_0}, m_0, l) = \left[ 1 + 0.25(1 + \alpha_{m_0})^2 - (1 + \alpha_{m_0})\cos \frac{\pi}{l + 1} \right]^{1/2} \alpha_{m_0}^{-1}. \]

If \( \delta = (m_0 + 4)(l + 1)/54 > 1 \) and \( n \geq n' \), then

\[ K_m^1 \geq \delta K_m^2 \quad \text{and} \quad K_a^1 < \delta K_a^2. \]

**Theorem 3.1.** If \( l \geq l' \), then

\[ q(\alpha_{m_0}, m_0, l) > (2/5)^\delta. \]

**Proof.** For a large \( l' \)

\[ q(\alpha_{m_0}, m_0, l)^2 \geq \frac{1 - (\cos \pi/(l + 1))^2}{\alpha_{m_0}^2} > \frac{9\alpha_{m_0}^{-2}}{(l + 1)^2} \quad (l \geq l') \]

and

\[ (5/2)^\delta > l + 1. \]

From this fact the theorem immediately follows.

If \( l \geq l' \), then the cost of \( d \) steps of Algorithm 1 gives \([\delta d]\) steps using the Lehmer-Schur method. By Theorem 3.1 we have

\[ c^* [q(\alpha_{m_0}, m_0, l)]^d > (2/5)^{\delta d} \quad (c^* > 0, d \geq d_0), \]

which proves that the Lehmer-Schur process is faster than the Turán process. For the parameters \( m_0 = 4, \alpha_4 = 0.9, l = 11 \), (see [5] – [6]) the relation (3.12) is also satisfied. This can be verified easily by (3.10) and (3.11).

In the paper [6] there is a reference to the infinite precision integer arithmetics [1] for the sake of application of Algorithm 1. It is known [1] that the computing time of the multiplication is at most

\[ l(x)^{1+\tau} \quad (1 \geq \tau > 0) \]

units of time \((l(x)\) denotes the length of \( x \) in the binary system). Since Algorithm 1 has to use numbers of length at least \( 2^{m_0-2}l(x) \) where \( l(x) \) is needed by Algorithm 2, for the cost functions in the measure of computing time,

\[ K_m^1(t) \geq (\delta 2^{m_0-2})^{1+\tau} K_m^2(t) \]

is satisfied. As a simple corollary, in (3.12) we can write \( \delta 2^{m_0-2} \) instead of \( \delta \). This fact increases the relative convergence speed of the Lehmer-Schur process.

Department for Numerical Mathematics and Computing
Eötvös Loránd University
Budapest, Hungary