CS 107

A THREE-STAGE VARIABLE-SHIFT ITERATION FOR POLYNOMIAL ZEROS AND ITS RELATION TO GENERALIZED RAYLEIGH ITERATION

BY

M A JENKINS

AND

J. F. TRAUB

TECHNICAL REPORT NO. CS 107

AUGUST 26, 1968

COMPUTER SCIENCE DEPARTMENT

School of Humanities and Sciences

STANFORD UNIVERSITY

A THREE-STAGE VARIABLE-SHIFT ITERATION FOR POLYNOMIAL ZEROS AND ITS RELATION TO GENERALIZED RAYLEIGH ITERATION*

M. A. Jenkins
Stanford University
Stanford, California

and

J. F. Traub
Bell Telephone Laboratories, Incorporated
Murray Hill, New Jersey

* This work was supported in part by the National Science Foundation and the Office of Naval Research,

A THREE-STAGE VARIABLE-SHIFT ITERATION FOR POLYNOMIAL ZEROS AND ITS RELATION TO GENERALIZED RAYLEIGH ITERATION

by

M. A. Jenkins
Stanford University
Stanford, California

and

J. F. Traub
Bell Telephone Laboratories, Incorporated
Murray Hill, New Jersey

ABSTRACT

We introduce a new three-stage process for calculating the zeros of a polynomial with complex coefficients. The algorithm is similar in spirit to the two-stage algorithms studied by Traub in a series of papers, The algorithm is restriction free, that is, it converges for any distribution of zeros. A proof of global convergence is given.

Zeros are calculated in roughly increasing order of magnitude to avoid deflation Instability. Shifting is incorporated in a natural and stable way to break equimodularity and speed convergence. The three stages use no shift, a fixed shift, and a variable shift, respectively,

Tc obtain additional insight we recast the problem and algorithm into matrix form. The third stage is inverse iteration with the companion matrix, followed by generalized, Rayleigh iteration,

A program implementing the algorithm was written in a dialect of ALGOL 60 and run on Stanford University's IBM 360/67. The program has been extensively tested and testing is continuing. For polynomials with complex coefficients and of degrees ranging from 20 to 50, the time required to calculate all zeros averages $8n^2$ milliseconds.

Timing information and a numerical example are provided. A description of the implementation, an analysis of the effects of finite-precision arithmetic, an ALGOL 60 program, the results of extensive testing, and a second program which clusters the zeros and provides a posteriori error bounds will appear elsewhere.

TABLE OF CONTENTS

A THREE-STAGE VARIABLE-SHIFT ITERATION FOR POLYNOMIAL ZEROS AND ITS RELATION TO GENERALIZED RAYLEIGH ITERATION

by

M. A. Jenkins
Stanford University
Stanford, California

and

J. F. Traub
Bell Telephone Laboratories, Incorporated
Murray Hill, New Jersey

1. INTRODUCTION

We introduce a three-stage algorithm for calculating the zeros of" a polynomial P,

$$P(z) = \sum_{i=0}^{n} a_i z^{n-i}, \quad a_0 = 1, \; a_n \neq 0, \tag{1.1}$$

$$P(z) = \prod_{i=1}^{j} (z-\rho_i)^{m_i}.$$

The condition $a_0 = 1$ is for convenience only. The coefficients are in general complex. The algorithm involves iteration in the complex plane, Elsewhere we shall analyze the appropriate analogue for polynomials with real coefficients (and complex conjugate zeros) which uses only real arithmetic,,

The zeros are calculated one at a time and zeros of multiplicity m are found m times. The zeros are found in roughly increasing order of magnitude to avoid the instability arising from deflation with a large zero (Wilkinson [lo]).

The algorithm is similar in spirit to the two-stage algorithms proposed by Traub [6], [7],[8]. In [6] Traub gives a class of always convergent algorithms for calculating the largest zero. An instance of the class of algorithms given in[6] follows. Notation has been modified to agree with the notation of this paper.

Let

$$G^{(0)}(z) = P'(z),$$

(1.2)

$$G^{(\lambda+1)}(z) = zG^{(\lambda)}(z) - \alpha^{(\lambda)}P(z), \qquad A = 0,1,\ldots,L-1$$

where $\alpha^{(\lambda)}$ is the leading coefficient of $G^{(\lambda)}(z)$. Let z_0 be arbitrary and let

(1.3)
$$z_{i+1} = z_i - R^{(L)}(z_i)$$

where

$$R^{(L)}(z) = a^{(L)}P(z)/G^{(L)}(z)$$

If P(z) has a largest zero and if L is sufficiently large and fixed, then this iteration converges to the largest zero,

The appropriate modification of this algorithm for the case of a pair of complex conjugate zeros was announced in [7].

In [8] Traub gives the following algorithm for calculating the smallest zero. Let

(1.4)

$$H^{(0)}(z) = P'(z),$$

$$H^{(\lambda+1)}(z) = \frac{1}{z}\left[H^{(\lambda)}(z) - \frac{H^{(\lambda)}(0)}{P(0)} P(z)\right], \qquad A = 0,1,\ldots,L-1.$$

Let z_0 be arbitrary and let

(1.5)
$$z_{i+1} = z_i - V^{(L)}(z_i),$$

where

$$V^{(L)}(z) = \beta^{(L)} P(z)/H^{(L)}(z),$$

and $\beta^{(L)}$ is the leading coefficient of $H^{(L)}(z)$. If P(z) has a smallest zero and if L is sufficiently large and fixed, then the iteration converges to the smallest zero.

The implementation by Jenkins and Traub of a general polynomial solver based on two-stage algorithms is described in [3]. Separate procedures are used depending on whether there are one or two smallest zeros. If there are more than two distinct smallest zeros, a process of "double translation*' described in Section 6 of [3] is used to break up the equimodularity.

The two-stage algorithm implemented in [3] has the following desirable characteristics:

1. The mathematical algorithm is restriction-free, that is, it converges for any distribution of zeros.
2. Zeros are calculated in roughly increasing order of modulus; this avoids the instability which occurs when the polynomial is deflated with a large zero.
3. The final stage is an iterative process and thus has the desirable stability features of iterative processes.
4. Few critical decisions have to be made by the program which implements the algorithm.
5. The algorithm is fast except for polynomials with many nearly equimodular zeros.

The three-stage algorithm introduced in this paper enjoys the first three characteristics, improvedfourth and fifth characteristics and a new characteristic.

4′ The number of critical decisions is further reduced,

5′ The algorithm is fast for all distribution of zeros,

6. Shifting is incorporated in the algorithm itself in a natural and stable way, Shifting breaks equi-modularity and speeds convergence.

We summarize the contents -of this paper,, The main properties of fixed and variable-shift H polynomials are given in Sections 2 and 3 and the mathematical algorithm is stated in Section 4. Global convergence for an arbitrary distribution of zeros is proven in Section 5 and the quadratic character of the convergence is established in Section 6.

In Section 7 we recast the problem and algorithm in matrix form and prove that Stage Three may be viewed as an efficient process for carrying out inverse powering using a companion matrix with shifted eigenvalues and generalized Rayleigh iteration. Although we are dealing with the case of a non-Hermitean matrix with nonlinear elementary divisors, the process does not suffer from the customary (Ostrowski [5]) slow convergence.

In Section 8 we prove that the third stage is precisely equivalent to Newton-Raphson iteration applied to a sequence of rational functions converging to a linear

polynomial. It is a Newton-Raphson iteration even though no differentiation is performed.

Our focus in this paper is on the mathematical algorithm and its properties., Timing information and a numerical example are provided. A description of the implementation, an analysis of the effects of finite-precision arithmetic, an ALGOL 60 program, the results of extensive testing, and a second program which clusters the zeros and provides a posteriori error bounds will appear elsewhere.- In Section 9 we do discuss a number of important points pertaining to stability and decisions to be made by the implementing program, In the final section we give a small numerical example.

2. FIXED-SHIFT H POLYNOMIALS

We introduce fixed-shift H polynomials and prove a number of their properties. Let $H^{(0)}(z)$ be a polynomial of degree at most n - 1. Let s be a complex number with $P(s) \neq 0$. Define the sequence

(2.1)
$$H^{(\lambda+1)}(z) = \frac{1}{z-s}\left[H^{(\lambda)}(z) - \frac{H^{(\lambda)}(s)}{P(s)}P(z)\right], \qquad \lambda = 0,1,\ldots .$$

The $H^{(\lambda)}(z)$ are polynomials of degree at most n - 1.

Define

(2.2)
$$P_i(z) = \frac{P(z)}{z-\rho_i} .$$

The properties of $H^{(\lambda)}(z)$ follow from the following lemma which is easily proven by induction.

LEMMA. Assume

$$H^{(0)}(z) = \sum_{i=1}^{j} c_i^{(0)} P_i(z).$$

Then for all λ,

(2.3)
$$H^{(\lambda)}(z) = \sum_{i=1}^{j} c_i^{(0)} (\rho_i - s)^{-\lambda} P_i(z).$$

Note that the assumption about $H^{(0)}(z)$ is equivalent to assuming that the partial fraction expansion of $H^{(0)}(z)/P(z)$ has only linear terms. We will ensure this by taking $H^{(0)}(z) = P'(z)$. A matrix formulation of this is given in Section 7.

We define

$$\overline{H}^{(\lambda)}(z) = \frac{H^{(\lambda)}(z)}{\sum_{i=1}^{j} c_i^{(0)}(\rho_i - s)^{-\lambda}}.$$

Thus $\overline{H}^{(\lambda)}(z)$ is $H^{(\lambda)}(z)$ divided by its leading coefficient. Our interest in H polynomials is due to the following theorem -which follows from (2.3).

THEOREM. Assume $c_1^{(0)} \neq 0$. Let s be such that $|\rho_1 - s| < |\rho_i - s|$, $i = 2,\ldots,j$. Then for all finite z,

$$\lim_{\lambda\to\infty} \overline{H}^{(\lambda)}(z) = P_1(z). \tag{2.4}$$

Observe that (2.4) may- be written as

$$\lim_{\lambda\to\infty} z - \frac{P(z)}{\overline{H}^{(\lambda)}(z)} = \rho_1. \tag{2.5}$$

(The zero labelled ρ_1 depends on the choice of s.) The rate of convergence depends on $\max[|\rho_1 - s|/|\rho_i - s|]$. This suggests that s be changed to be the best available approximation to ρ_1. This leads to the idea of 'variable-shift H polynomials.

3. VARIABLE-SHIFT H POLYNOMIALS

Let $H^{(0)}(z)$ be a polynomial of degree at most n - 1. Let s_0 be a complex number with $P(s_0) \neq 0$. Define the sequence

$$H^{(\lambda+1)}(z) = \frac{1}{z-s_\lambda}\left[H^{(\lambda)}(z) - \frac{H^{(\lambda)}(s_\lambda)}{P(s_\lambda)}P(z)\right],$$

(3.1)

$$s_{\lambda+1} = s_\lambda - \frac{P(s_\lambda)}{H^{(\lambda+1)}(s_\lambda)},$$

$\lambda = 0,1,\ldots$. If $P(s_\lambda) = 0$, terminate the calculation. The $H^{(\lambda)}(z)$ are polynomials of degree at most n - 1. There should be no confusion from using the same symbol for the sequences generated by (2.1) and (3.1). The following lemma is easily verified.

LEMMA. Let

$$H^{(0)}(z) = \sum_{i=1}^{j} c_i^{(0)} P_i(z).$$

Then for all λ,

$$H^{(\lambda)}(z) = \sum_{i=1}^{j} c_i^{(\lambda)} P_i(z),$$

(3.2)

$$c_i^{(\lambda)} = c_i^{(0)} \prod_{t=0}^{A-1'} (\rho_i - s_t)^{-1}.$$

We defer the investigation of the convergence of the variable-shift process to Section 5.

4. THE ALGORITHM

We motivate the three-stage algorithm described below. In Stage One, we calculate a sequence of fixed-shift H polynomials with s = 0. This is the no-shift process, The purpose is to make the smaller zeros stand out. (See Section 9.) If there is a smallest zero, we obtain convergence according to the theorem of Section 2 and the fixed-shift calculation of Stage Two is not necessary.

However, rather than testing for convergence of the no-shift H sequence, we terminate Stage One after a small number of steps and enter Stage Two where we calculate a sequence of fixed-shift H polynomials using a complex number s -whose madulus is less than the smallest zero and whose amplitude is randomly chosen. (See Section 9). There are only a finite number of points on the circle $|z| = |s|$ which are equidistant from two or more zeros. According to (2.5) the sequence

$$t_\lambda = s - \frac{P(s)}{\overline{H}^{(\lambda)}(s)}$$

will converge to the zero closest to s, provided there is such a zero. As soon as $\{t_\lambda\}$ passes a. convergence test (see Section 9), we are ready to enter Stage Three. Let the test be passed when A = L. Then $s - P(s)/\overline{H}^{(L)}(s)$ should be close to ρ_1 and this is the starting value of the shift for Stage Three. These shifts should converge very rapidly to ρ_1 (see Sections 5 and 6).

The algorithm is used to calculate a zero of P. After each zero is found, the polynomial is deflated and then the algorithm is applied to the deflated polynomial,, Hence P represents either the original. polynomial or a polynomial obtained by deflation.

Stage One. No-Shift Process .

$$H^{(0)}(z) = P(z),$$

(4.1)

$$H^{(\lambda+1)}(z) = \frac{1}{z}\left[H^{(\lambda)}(z) - \frac{H^{(\lambda)}(0)}{P(0)} P(z)\right], \quad \lambda = 0,1,\ldots,M-1.$$

Stage Two. Fixed-Shift Process.

Take β to be a positive number such that $\beta \leq \min|\rho_i|$ and let s be such that $|s| = \beta$ and such that

(4.2) $$|s-\rho_1| < |s-\rho_i|, \quad i = 2,\ldots,j.$$

Let

(4.3)

$$H^{(\lambda+1)}(z) = \frac{1}{z-s}\left[H^{(\lambda)}(z) \quad \frac{H^{(\lambda)}(s)}{P(s)} P(z)\right], \quad \lambda = M,M+1,\ldots,L-1.$$

Stage Three. Variable-Shift Process.

Take

$$s_L = s - \frac{P(s)}{\overline{H}^{(L)}(s)} ,$$

and let

$$H^{(\lambda+1)}(z) = \frac{1}{z-s_\lambda}\left[H^{(\lambda)}(z) - \frac{H^{(\lambda)}(s_\lambda)}{P(s_\lambda)} P(z)\right],$$

(4.4)

$$s_{\lambda+1} = s_\lambda - \frac{P(s_\lambda)}{\overline{H}^{(\lambda+1)}(s_\lambda)} , \quad \lambda = L, L+1, \ldots .$$

There are a number of iterative processes used in the algorithm. In each of the three stages there is an iteration producing a sequence of polynomials. Regarding the vector of coefficients as basic, we refer to these iterations as vector iterations. In Stage Three we compute a sequence of shifts, We refer to this as a scalar iteration.

5. PROOF OF GLOBAL CONVERGENCE

We investigate the convergence of the three-stage algorithm. We begin by investigating the convergence of the variable-shift process defined by (4.4).

LEMMA: Assume

i. $|s_L-\rho_1| < \frac{1}{2}R$, where $R = \min_i |\rho_1-\rho_i|$,

ii. $c_1^{(L)} \neq 0$,

iii. $D_L \leq \sum_{i=2}^{j} \frac{|c_i^{(L)}|}{|c_1^{(L)}|} \leq \frac{1}{3}$.

Then $s_\lambda \to \rho_1$.

PROOF. We defer to the end of this proof the demonstration that the iteration is always defined. We show first that if the iteration is defined, it converges. We know by (3.2) that

$$H^{(\lambda)}(z) = \sum_{i=1}^{j} c_i^{(\lambda)} P_i(z),$$

$$c_i^{(A)} = c_i^{(L)} \prod_{t=L}^{h-1} (\rho_i-s_t)^{-1}, \quad \lambda \geq L .$$

Some algebraic manipulation leads to

$$(5.2)\qquad \frac{s_{\lambda+1}-\rho_1}{s_\lambda-\rho_1} = \frac{\sum_{i=2}^{j}\left[r_i^{(\lambda)}\right]^2 d_i^{(\lambda)} - \sum_{i=2}^{j} r_i^{(\lambda)} d_i^{(\lambda)}}{1 + \sum_{i=2}^{j}\left[r_i^{(\lambda)}\right]^2 d_i^{(\lambda)}},$$

where

$$r_i^{(\lambda)} = \frac{s_\lambda-\rho_1}{s_\lambda-\rho_i}, \qquad d_i^{(\lambda)} = \frac{c_i^{(\lambda)}}{c_1^{(\lambda)}}.$$

Let

$$\frac{|s_{\lambda+1}-\rho_1|}{|s_\lambda-\rho_1|} = T_\lambda.$$

We prove convergence by showing there exists a τ_L such that for all $\lambda \geq L$, $T_\lambda \leq \tau_L < 1$. The proof is by induction.

Observe that

$$|r_i^{(L)}| = \frac{|s_L-\rho_1|}{|s_L-\rho_i|} < 1.$$

Hence

$$T_L \le \frac{\sum_{2}^{j} |d_i^{(L)}| + \sum_{2}^{j} |d_i^{(L)}|}{1 - \sum_{2}^{j} |d_i^{(L)}|} = \frac{2D_L}{1-D_L} .$$

By hypothesis $D_L < \frac{1}{3}$. Let

$$\tau_L = \frac{2D_L}{1-D_L} . \tag{5.3}$$

Then $T_L \le \tau_L < 1$.

Assume now that $T_L, T_{L+1}, \ldots, T_{\lambda-1} \le \tau_L < 1$. Hence fort = $L, L+1, \ldots, \lambda$,

$$|s_t - \rho_1| \le |s_L - \rho_1| < \tfrac{1}{2}R,$$

$$|s_t - \rho_i| \ge |\rho_1 - \rho_i| - |s_t - \rho_1| > \tfrac{1}{2}R .$$

Thus

$$|r_i^{(t)}| < 1, \quad t = L, L+1, \ldots, \lambda. \tag{5.4}$$

Observe that

$$d_i^{(\lambda)} = r_i^{(\lambda-1)} d_i^{(\lambda-1)} .$$

Hence

$$\sum_{2}^{j} |d_i^{(\lambda)}| \le D_L . \tag{5.5}$$

From (5.2), (5.3), (5.4),(5.5) it follows-that $T_\lambda \leq \tau_L < 1$ for $\lambda \geq L$ which completes the proof of convergence.

We now show that the sequence $\{s_\lambda\}$ is always well defined for $\lambda \geq L$.

$$\bar{H}^{(\lambda+1)}(s_\lambda) = \frac{\sum_{i=1}^{j} c_i^{(\lambda)}(\rho_i - s_\lambda)^{-1} P_i(s_\lambda)}{\sum_{i=1}^{j} c_i^{(\lambda)}(\rho_1 - s_\lambda)^{-1}}$$

$$= P_1(s_\lambda)\left[\frac{1 + \sum_{i=2}^{j} d_i^{(\lambda)}\left[r_i^{(\lambda)}\right]^2}{1 + \sum_{i=2}^{j} d_i^{(\lambda)} r_i^{(\lambda)}}\right].$$

$P_1(s_\lambda) \neq 0$ by hypothesis i and the contraction argument. Since, as we have seen,

$$\left|\sum_{i=2}^{j} d_i^{(\lambda)}\left[r_i^{(\lambda)}\right]^2\right| < \frac{1}{3},$$

$\bar{H}^{(\lambda+1)}(s_\lambda) \neq 0$ and the iteration is well defined. This completes the proof of the lemma.

We now investigate the convergence of the three-stage algorithm defined in Section 4. The major result of the paper is given by the following

THEOREM. For all L sufficiently large and fixed, $s_\lambda \to \rho_1$.

PROOF. Since $H^{(0)}(z) = P'(z)$,

$$H^{(0)}(z) = \sum_{i=1}^{j} m_i P_i(z) ,$$

and it follows from (2.3), with s = 0, that

$$H^{(M)}(z) = \sum_{i=1}^{j} m_i \rho_i^{-M} P_i(z).$$

Then

$$H^{(L)}(z) = \sum_{i=1}^{j} m_i \rho_i^{-M} (\rho_i - s)^{-(L-M)} P_i(z)$$

$$= \sum_{i=1}^{j} c_i^{(L)} P_i(z).$$

We have $c_1^{(L)} \neq 0$. Furthermore

$$\sum_{i=2}^{j} \frac{c_i^{(L)}}{c_1^{(L)}} = \sum_{i=2}^{j} \frac{m_i}{m_1} \left(\frac{\rho_1}{\rho_i}\right)^M \left(\frac{\rho_1 - s}{\rho_i - s}\right)^{L-M} .$$

Recall that $| \rho_1 - s | < | \rho_i - s |$.

Fix M. Then by choosing L sufficiently large we can make

$$D_L = \sum_{i=2}^{j} \frac{|c_i^{(L)}|}{|c_1^{(L)}|}$$

as small as desired. Choose L so that

(5.6) $$D_L < \tfrac{1}{3}$$

and

(5.7) $$|s-\rho_1| \frac{2D_L}{1-D_L} < \tfrac{1}{2}R \quad .$$

-The condition of (5.7) ensures that $|s_L-\rho_1| < \frac{1}{2}R$. All the hypotheses of the lemma now hold and the conclusion follows.

6. RATE OF CONVERGENCE

Let

$$C(h) = \frac{|s_{L+\lambda+1}-\rho_1|}{|s_{L+\lambda}-\rho_1|^2} . \tag{6.1}$$

In the last section we proved the existence of a number τ_L such that for $A \geq 0$,

$$\frac{|s_{L+\lambda+1}-\rho_1|}{|s_{L+\lambda}-\rho_1|} = T_\lambda \leq \tau_L < 1, \tag{6.2}$$

where $\tau_L = 2D_L/(1-D_L)$. We defined $R = \min_i|\rho_1-\rho_i|$. The rate of convergence of our algorithm is governed by the following

THEOREM. Let the hypotheses of the lemma of the previous section hold. Then

$$C(\lambda) \leq \frac{2}{R}\,\tau_L^{\lambda(\lambda-1)/2}. \tag{6.3}$$

PROOF. From (5.2),

$$\frac{s_{L+\lambda+1}-\rho_1}{(s_{L+\lambda}-\rho_1)^2} = \frac{\sum_{i=2}^{j} r_i^{(L+\lambda)} d_1^{(L+\lambda)} A_L \, s_{L+\lambda}-\rho_i - - - \sum_{i=2}^{j} \frac{d_i^{(L+\lambda)}}{s_{L+\lambda}-\rho_i}}{1 + \sum_{i=2}^{j} \left[r_i^{(L+\lambda)}\right]^2 d_i^{(L+\lambda)}} . \tag{6.4}$$

One may verify that for all λ and $i > 1$,

$$|r_i^{(L+\lambda)}| \leq \tau_L^{\lambda}, \quad \frac{1}{|s_{\lambda+L}-\rho_i|} \leq \frac{2}{R},$$

and

$$\sum_{i=2}^{j} |d_i^{(L+\lambda)}| \leq \tfrac{1}{3}\tau_L^{\lambda(\lambda-1)/2}.$$

Substituting these bounds into (6.4) establishes the theorem.

Thus the process is second order with an error constant C(A) which approaches zero. This may be contrasted with the conventional Newton-Raphson iteration in which there is no control over the error constant.

COROLLARY. Let the hypotheses of the lemma of the previous section hold. Then for $\lambda > 0$,

$$|s_{L+\lambda}-\rho_1| \leq \tfrac{1}{2}R\tau_L^{\eta},$$

$$\eta = \tfrac{1}{2}[3\cdot 2^{\lambda} - (\lambda^2+\lambda+2)].$$

PROOF. For $\lambda = 1$ this follows from (6.2). For $\lambda > 1$ it follows upon substituting (6.3) into (6.1).

7. VARIABLE-SHIFT ITERATION IS GENERALIZED RAYLEIGH ITERATION

We now give a matrix interpretation of our algorithm. We show that in a matrix formulation the vector iteration of the third stage is inverse powering with a matrix whose eigenvalues have been shifted, while the scalar iteration is generalized Rayleigh iteration.

Let

$$A = \begin{pmatrix} 0 & 0 & \cdots & 0 & -a_n \\ 1 & 0 & \cdots & 0 & -a_{n-1} \\ 0 & 1 & \cdots & 0 & -a_{n-2} \\ \vdots & \vdots & \cdots & & \vdots \\ \vdots & \vdots & \cdots & \vdots & \vdots \\ \vdots & \vdots & \cdots & \vdots & \vdots \\ \vdots & \vdots & \cdots & 1 & -a_1 \end{pmatrix}$$

be the companion matrix of P. Let

$$H^{(\lambda)}(z) = \sum_{i=0}^{n-1} h_i^{(\lambda)} z^{n-1-i}$$

and

$$[h^{(\lambda)}]^T = (h_{n-1}^{(\lambda)}, \ldots, h_0^{(\lambda)}).$$

Let

$$P_i(z) = \frac{P(z)}{z-\rho_i} = \sum_{j=0}^{n-1} p_{ij} z^{n-1-j} .$$

Define

$$\underline{p}_i^T = (p_{in-1}, \ldots, p_{i0}),$$

$$\underline{q}_i^T = (1, \ldots, \rho_i^{n-i}).$$

One may easily verify that for the eigenvalue ρ_i, the right and left eigenvectors are $\underline{p}_i$ and $\underline{q}_i^T$, respectively.

One may verify that the initial condition

$$H^{(0)}(z) = P'(z)$$

is equivalent to

$$\underline{h}^{(0)} = \sum_{i=1}^{j} m_i \underline{p}_i ,$$

the fixed-shift recurrence

$$(7.1) \qquad H^{(\lambda+1)}(z) = \frac{1}{z-s}\left[H^{(\lambda)}(z) - \frac{H^{(\lambda)}(s)}{P(s)} P(z)\right]$$

is equivalent to

$$\underline{h}^{(\lambda+1)} = (A-sI)^{-1}\underline{h}^{(\lambda)} \quad , \tag{7.2}$$

while the variable-shift recurrence is equivalent to

$$h^{(\lambda+1)} = (A-s_{\lambda}I)^{-1}\underline{h}^{(\lambda)} \quad . \tag{7.3}$$

Equations (7.2) and (7.3) exhibit the processes as inverse powering with a matrix whose eigenvalues have been shifted, This is also called inverse iteration (Wielandt [9]). We show that

$$s_{\lambda+1} = s_{\lambda} - \frac{P(s_{\lambda})}{H^{(\lambda+1)}(s_{\lambda})} \tag{7.4}$$

is equivalent to

$$s_{\lambda+1} = \frac{[\underline{s}^{(\lambda)}]^{T}A\underline{h}^{(\lambda+1)}}{[\underline{s}^{(\lambda)}]^{T}\underline{h}^{(\lambda+1)}} \tag{7.5}$$

where

$$[\underline{s}^{(\lambda)}]^{T} = (1, s_{\lambda}, \ldots, s_{\lambda}^{n-1}).$$

Now (7.4) may be written as

$$s_{\lambda+1} = \frac{s_{A}H^{(\lambda+1)}(s_{A}) - h_{0}^{(\lambda+1)}P(s_{\lambda})}{H^{(\lambda+1)}(s_{\lambda})} \tag{7.6}$$

Observe that

$$H^{(\lambda+1)}(s_\lambda) = [s^{(\lambda)}]^T h^{(\lambda+1)} . \tag{7.7}$$

From (7.1),

$$h_0^{(\lambda+1)} = - \frac{H^{(\lambda)}(s_\lambda)}{P(s_\lambda)} .$$

Hence.

$$\begin{aligned} s_\lambda H^{(\lambda+1)}(s_\lambda) - h_0^{(\lambda+1)} P(s_\lambda) &= s_\lambda H^{(\lambda+1)}(s_\lambda) + H^{(\lambda)}(s_\lambda) \\ &= [s^{(\lambda)}]^T (A\underline{h}^{(\lambda+1)} - \underline{h}^{(\lambda)}) + [s^{(\lambda)}]^T h^{(\lambda)} \\ &= [\underline{s}^{(\lambda)}]^T A h^{(\lambda+1)} . \end{aligned}$$

Substituting this result together with ('7.7) into (7.6) com- , pletes the proof.

We summarize this result in a

THEOREM. The variable-shift recurrence

$$H^{(\lambda+1)}(z) = \frac{1}{z-s_\lambda} \left[H^{(\lambda)}(z) - \frac{H^{(\lambda)}(s_\lambda)}{P(s_\lambda)} P(z) \right] ,$$

$$s_{\lambda+1} = s_\lambda - \frac{P(s_\lambda)}{\overline{H}^{(\lambda+1)}(s_\lambda)}$$

is equivalent to

$$\underline{h}^{(\lambda+1)} = (A-s_\lambda I)^{-1}\underline{h}^{(\lambda)} ,$$

$$s_{\lambda+1} = \frac{[\underline{s}^{(\lambda)}]^T A\underline{h}^{(\lambda+1)}}{[\underline{s}^{(\lambda)}]^T\underline{h}^{(\lambda+1)}} .$$

Observe that $\underline{h}^{(\lambda)} \to \underline{p}_1$ and $[\underline{s}^{(\lambda)}]^T \to \underline{q}_1^T$. Hence (7.5) is a generalized Rayleigh iteration (Wilkinson [11, p.179], Ostrowski [5]) appropriate for non-Hermitean matrices.

However we are in a very favorable position as compared with the usual situation when inverse iteration and generalized Rayleigh iteration are applied.

1. No calculation has to be performed to determine the left eigenvector. It is simply $[\underline{s}^{(\lambda)}]^T$.
2. Multiple eigenvalues of a companion matrix imply nonlinear elementary divisors. In general this leads to (Ostrowski [5]) linear convergence. In our case multiple zeros do not affect the rate of convergence (it is still quadratic) and require no special attention.
3. The inverse iteration is carried out explicitly.
4. The initial vector $h^{(0)}$ and all succeeding vectors $\underline{h}^{(\lambda)}$ lie in the subspace spanned by the eigenvectors

of A. Furthermore the $\underline{h}^{(\lambda)}$ cannot be deficient in the eigenvector corresponding'to the eigenvalue being calculated.

5. Our process is globally convergent to one of the smallest eigenvalues of A and hence deflation is carried out under favorable conditions.

Bauer and Samelson [2] have suggested an iteration which is related to Stage Three of our algorithm. Performance of an analogous process on a sequence of polynomials of decreasing degree leads to Bauer's [1] Treppeniteration.

8. NEWTON-RAPHSON ITERATION

We show that the formula

$$s_{\lambda+1} = s_\lambda - \frac{P(s_\lambda)}{\overline{H}^{(\lambda+1)}(s_A)}$$

is precisely a Newton-Raphson iteration performed on a certain rational function. The word precisely in the previous sentence is to emphasize that the iteration is not merely of Newton-Raphson type. We prove the following

THEOREM. The formula

$$s_{\lambda+1} = s_\lambda - \frac{P(s_\lambda)}{\overline{H}^{(\lambda+1)}(s_{\lambda+1})}$$

is identical with

$$s_{\lambda+1} = s_\lambda - \frac{W^{(\lambda)}(s_\lambda)}{[W^{(\lambda)}(s_\lambda)]'},$$

where

$$W^{(\lambda)}(z) = \frac{P(z)}{H^{(\lambda)}(z)}.$$

PROOF. Let

$$V^{(\lambda)}(z) = \frac{H^{(\lambda)}(z)}{P(z)} .$$

Hence

$$H^{(\lambda+1)}(z) = \frac{1}{z-s_\lambda}\left[H^{(\lambda)}(z) \quad \frac{H^{(\lambda)}(s_\lambda)}{P(s_\lambda)} P(z)\right]$$

may be written as

$$V^{(\lambda+1)}(z) = \frac{V^{(\lambda)}(z) - V^{(\lambda)}(s_\lambda)}{z - s_\lambda}$$

and

$$V^{(\lambda+1)}(s_\lambda) = [V^{(\lambda)}(s_\lambda)]' .$$

Furthermore

$$h_0^{(\lambda+1)} = V^{(\lambda)}(s_\lambda)$$

Hence

$$s_{\lambda+1} = s_\lambda - \frac{P(s_\lambda)}{\overline{H}^{(\lambda+1)}(s_\lambda)} = s_\lambda - \frac{P(s_\lambda)h_0^{(\lambda+1)}}{H^{(\lambda+1)}(s_\lambda)}$$

$$= s_\lambda + \frac{V^{(\lambda)}(s_\lambda)}{[V^{(\lambda)}(s_\lambda)1'}$$

$$= s_\lambda - \frac{W^{(\lambda)}(s_\lambda)}{[W^{(\lambda)}(s_\lambda)]'} .$$

This permits us to regard variable shift iteration in the following manner. We are performing Newton-Raphson iteration on a sequence of rational functions $P(z)/H^{(\lambda)}(z)$. For A sufficiently large, $P(z)/H^{(\lambda)}(z)$ is as close as desired to a linear polynomial whose zero i's ρ_1. This shows why the process is so powerful.

Note that no differentiation is performed in our calculation of the sequence $\{s_\lambda\}$. The division by $z - s_\lambda$ has the effect of differentiation.

9. IMPLEMENTATION OF THE ALGORITHM

The program implementing the algorithm, a discussion of how the program makes its decisions, stability of the algorithm in finite precision arithmetic, the results of extensive testing, and a program which clusters the zeros and provides a posteriori error bounds will appear elsewhere. Here we confine ourselves to a few observations.

The termination of Stage One, that is, the choice of M, is not crucial. Indeed Stage- One is not necessary from theoretical considerations. The function of Stage One is to accentuate the smaller zeros. Numerical experimentation indicates that this makes the decision to terminate Stage Two -more reliable. In the implementation, M is set at 5, a number arrived at by numerical experience.

The following three major decisions have to be made by the program:

1. Selection of the shift s.
2. Termination of Stage Two; that is, the choice of I_j.
3. Termination of Stage Three.

We indicate how these three decisions are made.

1. Selection of s.

This parameter is chosen so that $|s| = \beta$, $\beta < \min|\rho_i|$, $i = 1,2,\ldots,j$ and so that

$$|s-\rho_1| < |s-\rho_i|, \quad i = 2,\ldots,j. \tag{9.1}$$

A lower bound on the moduli of the zeros due to Cauchy (Marden [4, p.98, ex. 1]) is given by the unique positive zero, β, of the polynomial

$$z^n + |a_1| z^{n-1} + \ldots + |a_{n-1}| z - |a_n| .$$

This number is easily calculated by Newton-Raphson iteration. The value of s is then chosen by using random numbers from a uniform distribution to pick a point-on the circle of radius β. It is highly probable that the s so chosen will be closest to just one of the zeros of P and hence the condition (9.1) is satisfied. If the condition is not satisfied, the test described below may not be passed in which case a new value of s is chosen. Observe that s need not be closest to the smallest zero of P. It is easy to show that it will be closest to a zero whose modulus is at most three times the modulus of the smallest zero. Hence we guarantee that we will never perform a deflation using a zero which is large compared to other zeros of P. Thus we avoid a situation (Wilkinson [10] which could lead to serious instability.

2. Termination of Stage Two.

We do not attempt to carry out Stage Two far enough to assure that the conditions of the Theorem of Section 5 are

satisfied. (These are only sufficient conditions). Instead we test for the convergence of the sequence $s - P(s)\,/\overline{H}^{(A)}(s)$. Experience has shown that it is efficient to terminate Stage Two after only a very weak test for convergence has been passed. Let $t_\lambda = s - P(s)\,/\overline{H}^{(\lambda)}(s)$. If $t_\lambda, t_{\lambda+1}, t_{\lambda+2}$ are defined and

$$|t_{\lambda+1}-t_\lambda| \leq \tfrac{1}{2}|t_\lambda|, \quad |t_{\lambda+2}-t_{\lambda+1}| \leq \tfrac{1}{2}|t_{\lambda+1}|$$

then we terminate Stage Two.

If the test is not passed by the time λ reaches a certain value (which itself is varied depending on how many shifts have been tried), a new value of s is generated with modulus β and random amplitude.

Since we are using such a weak convergence test for the sequence, we must allow for the fact that there will be cases when Stage Three is started prematurely with an s_L and $H^{(L)}$ which do not lead to convergence. This causes no difficulty as we observe below. We emphasize that the generation of a sequence of values of s is only a contingency. In practice it is very rare that the first s chosen doesn't serve, and the largest number of trial shifts required in the course of extensive testing has been three.

Termination of Stage Three.

As in [3] we terminate Stage Three when the computed value of the polynomial at s_λ is less than or equal to a bound on the roundoff error in evaluating $P(s_\lambda)$. Numerical experience indicates this is a good stopping criterion for polynomial zero-finding as it stops the iteration just as the limiting accuracy has been reached. Since we expect rapid convergence of the third stage we place a limit of ten on the number of iterations. If convergence is not aehieved when this limit is reached we return to Stage Two with the value of s used the last time we were in Stage Two unless the limiting value of A has been reached.

We turn to some other matters. We describe how the process for computing the H polynomials in either Stage Two or Three is actually carried out, We describe the Stage Three process, the Stage Two process being entirely analogous.

Rather, than computing the H polynomials by

$$(9.2) \qquad H^{(\lambda+1)}(z) = \frac{1}{z-s_\lambda}\left[H^{(\lambda)}(z) - \frac{H^{(\lambda)}(s_\lambda)}{P(s_\lambda)}\,P(z)\right],$$

the "scaled recurrence"

$$\overline{H}^{(0)}(z) \frac{1}{n} \quad P \quad (z),$$

(9.3)

$$\overline{H}^{(\lambda+1)}(z) = \frac{1}{z-s_\lambda}\left[P(z) - \frac{P(s_\lambda)}{\overline{H}^{(\lambda)}(s_\lambda)}\,\overline{H}^{(\lambda)}(z)\right]$$

is used. This generates a sequence of monic polynomials and avoids the overflow and underflow problems' which would occur if (9.2) were used. The use of (9.3) is equivalent to a method of scaling used by Traub [6, Section 9].

The computation of $\overline{H}^{(\lambda+1)}(z)$ by (9.3) requires $4n$ (in general complex) multiplications and additions. This may be reduced to $3n$ by the following observation.

Let

$$P(z) = Q_P^{(\lambda)}(z)(z-s_\lambda) + P(s_\lambda),$$

$$\overline{H}^{(\lambda)}(z) = Q_H^{(\lambda)}(z)(z-s_\lambda) + \overline{H}^{(\lambda)}(s_\lambda).$$

-Then

$$\overline{H}^{(\lambda+1)}(z) = Q_P^{(\lambda)}(z) - \frac{P(s_\lambda)}{\overline{H}^{(\lambda)}(s_\lambda)} Q_H^{(\lambda)}(z). \tag{9.4}$$

If $P(s_\lambda)$ and $\overline{H}^{(\lambda)}(s_\lambda)$ are calculated by the usual Horner recurrence, then $Q_P^{(\lambda)}(z)$, $Q_H^{(\lambda)}(z)$ are generated as a byproduct. In Stage Two s is fixed, P(s) and $Q_P^{(\lambda)}(z)$ are formed just once, and only 2n (in general complex) multiplications and additions are required per step.

A discussion of stability will appear elsewhere. Here we limit ourselves to a few observations. The process of calculating $\overline{H}^{(\lambda+1)}(z)$ by (9.3) is precisely the deflation of the polynomial

$$P(z) - \frac{P(s_\lambda)}{\overline{H}^{(\lambda)}(s_\lambda)} \overline{H}^{(\lambda)}(z) \tag{9.5}$$

which has a zero at s_λ and whose remaining zeros are near the zeros of $P(z)/(z-\rho_1)$. Because of the way that s and the s_λ are chosen, these numbers are among the smallest zeros in modulus of the polynomial given by (9.5), which is a desirable situation.

We use (9.4) rather than (9.3), in carrying out the process. Observe that in (9.4) $\overline{H}^{(\lambda)}(z)$ is calculated as the sum of a polynomial which is near $P(z)/(z-\rho_1)$ and a polynomial with small coefficients which may be viewed as a correction -term. This is a favorable situation for the control of roundoff.

The variable-shift stage may be viewed as a non-stationary iteration. It has the usual desirable stability properties common to iterative processes.

10. NUMERICAL RESULTS

Extensive numerical experimentation, performed on an IBM 360/67, leads to the timing results given below, Additional testing is planned.

For polynomials with real coefficients and of degrees ranging from 20 to 50, the time required to calculate all zeros averages $4n^2$ milliseconds. Thus a 20th degree polynomial takes 1.6 seconds, -a 50th degree polynomial takes 10 seconds.

The time for all the real polynomials of degree 20 or greater which were tested ranges from $3n^2$ to $7n^2$ milliseconds. . The polynomials used in the testing range from polynomials with randomly chosen zeros to polynomials with multiple zeros and clusters of near equimodular zeros. The fact that the time required is insensitive to the distribution of zeros is most encouraging.

The algorithm reported in this paper was not tailored, for polynomials with real coefficients. Elsewhere we shall report on an algorithm designed for real coefficients. The real algorithm cuts the time by a factor of roughly two.

For polynomials with complex coefficients and of degrees ranging from 20 to 50, the time required to calculate all the zeros averages $8n^2$ milliseconds.

These figures were obtained from an ALGOL 60 implementation of the algorithm. A FORTRAN implementation would be faster.

For illustration we exhibit a low degree numerical example. The purpose of this example is not intended to prove anything about the efficacy of the algorithm and its implementation, This has been done through extensive testfng which will be reported elsewhere,

The example given below has a zero of multiplicity two as well as three almost equimodular zeros, two of which form a near-multiple pair.

$$P(z) = z^5 - (13.999+5i)z^4 + (74.99+55.998i)z^3$$

$$- (159.959+260.982i)z^2 + (1.95+463.934i)z$$

$$+ (150-199.95i),$$

$$P(z) = (z-1-i)^2(z-4+3i)(z-4-3i)(z-3.999-3i).$$

In calculating each of the zeros below, five no-shift steps were taken (M = 5). In Table 1 we give the value of s used in Stage Two, the number of Stage Two steps (L-M), the value of s_L used to start Stage Three and the iterates s_{L+j} used in Stage Three.8 The program was written in a dialect of ALGOL 60 and run on Stanford University's IBM 360/67.

Observe that the well-conditioned zero at 4 -3iis calculated accurately even though the polynomial has already been deflated with three ill-conditioned zeros.

TABLE 1. A Numerical Example

Zero (1) $s = -.37087 + .17907i$, $L - M = 2$, $s_L = .99996 + 1.0001i$

j	s_{L+j}
3	
1	1.00000000000426 + .9999999999782011i

Zero (2) $s = -.66572 - .093001i$, $L - M = 2$, $s_L = .99975 + 1.0005i$

j	s_{L+j}
3	
1	.9999999999974143 + 1.000000000001921
2	.9999999999995742 + 1.000000000021801

Zero (3) $s = 1.1968 + .92016i$, $L - M = 4$, $s_L = 4.6023 + 3.0859i$

j	s_{L+j}
1	3.99420181191240 + 3.006238036392071
2	3.99946028022349 + 2.999953671867681
3	3.99939735288312 + 2.999886588791281
4	3.99900069469391 + 3.000383503135241
5	3.99892674982940 + 3.000001012817341
6	3.99899965220352 + 2.999999183018531
7	3.99899999997589 + 2.999999999985161

Zero (4) s = .26797 - 2.38811, L - M = 4, S_L = 3.8372 - 2.6754i

j	S_{L+j}
1	3.99861695624235 - 3.00234341883751
2	4.00000000757741 - 3.00000000435632i
3	4.00000000000000 - 3.00000000000000i

Zero (5) 4.00000000002411 + 3.00000000001484i

REFERENCES

1. Bauer, F. L., Beiträge zur Entwicklung numerischen Verfahren für programmgeseuerte Rechenanlagen. II. Direkte Faktorisierung eines Polynoms. Bayer. Akad. Wiss. Math. Nat. Kl. S. B. (1956), 163-203.
2. Bauer, F. L., and Samelson, K., Polynomkerne und Iterationsverfahren. Math. Zeitschr 67, (1957), 93 -98.
3. Jenkins, M. A. and Traub, J. F., An Algorithm for an Automat@ General Polynomial Solver. To appear in Proceedings of a Symposium on Constructive Aspects of the Fundamental Theorem of Algebra, Rüschlikon, Switzerland, 1967. Also available as Rept CS 71, Computer Science Department, Stanford University, July 21, 1967.
4. Marden, M., The Geometry of the Zeros of a Polynomial in a Complex Variable, Amer. Math. Soc., Providence, Rhode Island, 1949.
5. Ostrowski, A. M., On the Convergence of the Rayleigh Quotient Iteration for the Computation of the Characteristic Roots and Vectors, IV Generalized Rayleigh Quotient for Nonlinear Elementary Divisors. Arch. Rat. Mech. Anal. 3 (1959), 341-347.
6. Traub, J. F., A Class of Globally Convergent Iteration Functions for the Solution of Polynomial Equations. Math. Comp. 20 (1966), 113-138.

7. _____, Proof of Global Convergence of an Iterative Method for Calculating Complex Zeros of a Polynomial, Notices Amer. Math. Soc. 13 (1966), 117.

8. _____, The Calculation of Zeros of Polynomials and Analytic Functions. Proceedings of Symposia in Applied Mathematics Volume 19, Mathematical Aspects of Computer Science, pp. 138-152. Amer. Math. Soc., Providence, Rhode Island, 1967. Also available as Rept CS 36, Computer Science Department, Stanford University, April 12, 1966.

9. Wielandt, H., Bestimmung höherer Eigenwerte durch gebrochene Iteration. Ber. der Aerodynamischen Versuchanstalt Göttingen, B44/J/37, 1944.

10. Wilkinson, J. H., Rounding Errors in Algebraic Processes, Chapter 2, Prentice-Hall, Englewood Cliffs, New Jersey, 1963.

11. _____, The Algebraic Eigenvalue Problem. Clarendon Press, Oxford, 1965.