ELSEVIER

Journal of Computational and Applied Mathematics 126 (2000) 369–380

JOURNAL OF COMPUTATIONAL AND APPLIED MATHEMATICS

www.elsevier.nl/locate/cam

Computing a Hurwitz factorization of a polynomial

Luca Gemignani

Dipartimento di Matematica, Università di Pisa, Via Buonarroti 2, I-56125 Pisa, Italy

Received 31 March 1999; received in revised form 13 September 1999

Abstract

A polynomial is called a Hurwitz polynomial (sometimes, when the coefficients are real, a stable polynomial) if all its roots have real part strictly less than zero. In this paper we present a numerical method for computing the coefficients of the Hurwitz factor $f(z)$ of a polynomial $p(z)$. It is based on a polynomial description of the classical *LR* algorithm for solving the matrix eigenvalue problem. Similarly with the matrix iteration, it turns out that the proposed scheme has a global linear convergence and, moreover, the convergence rate can be improved by considering the technique of shifting. Our numerical experiments, performed with several test polynomials, indicate that the algorithm has good stability properties since the computed approximation errors are generally in accordance with the estimated condition numbers of the desired factors.

MSC: 65H05; 65F05

Keywords: *LR* matrix iteration; Polynomial factorization; Hurwitz polynomial; Stability

1. Introduction

Numerical techniques for splitting a polynomial $p(z)$ of degree n with respect to different contours C in the complex plane have both theoretical and practical interest. Despite of the vast literature concerning the case where C is the unit circle (see, for instance, [7,14] for general references on some classical methods based on Koenig's theorem and on the *qd* algorithm, and [9,13,12] for some recent improvements which yield numerical procedures with optimal sequential and parallel computational cost estimates), a few authors have considered the problem of factoring $p(z)$ with respect to the imaginary axis [2,3]. Such a decomposition has the form of $p(z)=f(z)g(z)$ where $g(z)$ and $f(z)$ have no roots with negative and nonnegative real parts, respectively. The polynomial $f(z)$, whose roots lie in the open plane $\Re(z)<0$, is called a Hurwitz polynomial or, if its coefficients are real, a stable polynomial. Stable polynomials play an essential role in dynamic stability and control theory [1,11].

E-mail address: gemignan@dm.unipi.it (L. Gemignani)

0377-0427/00/$ - see front matter
PII: S0377-0427(99)00365-9

Therefore, it is important to devise numerical tools for recognizing whether a given polynomial is a Hurwitz polynomial and for approximating its stable factor whenever the answer is negative.

The theory of generalized circles [6] easily yields a link between the two above-mentioned factorization problems. Indeed, the image of the straight line $\Re(z) = 0$ under the bilinear (Moebius) transformation $t_\alpha : z \to (z - \alpha)/(z + \alpha)$, $\alpha \in \mathbb{R}$, $\alpha \neq 0$, is the unit circle. Therefore, at least in principle, a factorization of $p(z)$ with respect to the imaginary axis can be obtained from one of the $\hat{p}(z) = p(t_\alpha(z))(z + \alpha)^n$ with respect to the unit circle. However, this approach suffers from several computational disadvantages. Firstly, the coefficients generated by the pre-processing phase are available only within certain errors and, by virtue of the well-known conditioning theorems, this can result in large final perturbations. Secondly, a numerical algorithm for recognizing stable polynomials can use profitably some necessary conditions that are very easy to check [6].

The aim of the present paper is to describe a numerical iterative scheme for evaluating the coefficients of the stable factor of degree k of a polynomial $p(z)$. The stability of a single step is not guaranteed, but since it is a method of iterative refinement using the original data, i.e., the coefficients of $p(z)$, for computing the correction, the overall stability of the method is quite good. The well-known *LR* algorithm for solving the eigenvalue problem for matrices of Hessenberg form gives the theoretical basis on which our approach rests. The polynomial scheme shares the convergence property of the matrix iteration but, unlike this, it can be arranged to produce a sequential algorithm which is very efficient in the typical case where $n - k \ll n$.

Let A_1 be an $n \times n$ lower Hessenberg matrix with unit superdiagonal entries having $p(z)$ as its characteristic polynomial. The stationary *LR* iteration applied to the starting matrix A_1 defines a sequence of similar matrices by means of the following rule:

$$\begin{aligned} &q(A_s) = L_s R_s, \\ &A_{s+1} = L_s^{-1} A_s L_s, \quad s \geqslant 1. \end{aligned} \tag{1}$$

Usually the shift function $q(z)$ is a polynomial whose job is to transform the spectrum of A_1 in order to accelerate the convergence. However, other possibilities can be considered and, for our purposes, the case when $q(z)$ is a rational function is much more interesting. In fact, the choice $q(z) = t_\alpha(z)$ influences the convergence in such a way that the eigenvalues λ_i of A_1 are treated not with respect to their order in magnitude, but in magnitude of $t_\alpha(\lambda_i)$. This means that, under very mild assumptions, the iterates A_s approach a block triangular form revealing the separation of the eigenvalues in the right half plane from the eigenvalues in the left half plane.

The matrix algorithm (1) can be completely translated into a polynomial setting. The resulting iterative scheme performs the shift with rational $q(z)$ implicitly and it generates the sequences of the characteristic polynomials of the last $n - k$ leading principal submatrices of A_s at the cost of $O(n(n - k))$ arithmetic operations per step. For the sequence of polynomials of degree k a global linear convergence to the stable factor $f(z)$ can be shown that is governed by the ratio of $\min\{|t_\alpha(\lambda_i)|: \lambda_i \in \Re(z) \geqslant 0\}$ and of $\max\{|t_\alpha(\lambda_i)|: \lambda_i \in \Re(z) < 0\}$. Once the convergence has taken place, an improvement of the convergence rate can be achieved by complementing the basic procedure with some suitable shift strategy (cf. [4,5]).

The paper is organized in the following way. Section 2 deals with the theoretical derivation of our method whereas, in Section 3, we discuss implementation issues and report the results of some numerical experiments.

2. Splitting a polynomial with respect to the imaginary axis

Let $p(z)$ be a monic complex polynomial of degree n,

$$p(z) = \sum_{i=0}^{n-1} a_i z^i + z^n = \prod_{i=1}^{n} (z - \lambda_i). \tag{2}$$

For any given positive real number α, one may consider the bilinear transformation

$$t_\alpha(z) = \frac{z - \alpha}{z + \alpha}. \tag{3}$$

Whenever $t_\alpha(z)$ is defined on the set Λ of the roots of $p(z)$, it can be easily observed that

$$\{\lambda_i \in \Lambda\colon |t_\alpha(\lambda_i)| < 1\} = \{\lambda_i \in \Lambda\colon \lambda_i \in \Re(z) > 0\},$$

$$\{\lambda_i \in \Lambda\colon |t_\alpha(\lambda_i)| = 1\} = \{\lambda_i \in \Lambda\colon \lambda_i \in \Re(z) = 0\}$$

and

$$\{\lambda_i \in \Lambda\colon |t_\alpha(\lambda_i)| > 1\} = \{\lambda_i \in \Lambda\colon \lambda_i \in \Re(z) < 0\}.$$

Thus, without loss of generality, we assume that the roots of $p(z)$ are numbered in such a way that

$$|t_\alpha(\lambda_1)| \geqslant \cdots \geqslant |t_\alpha(\lambda_k)| > 1 \geqslant |t_\alpha(\lambda_{k+1})| \geqslant \cdots \geqslant |t_\alpha(\lambda_n)| \tag{4}$$

and, then, we define the Hurwitz factor $f(z)$ of $p(z)$ by

$$f(z) = \prod_{i=1}^{k} (z - \lambda_i). \tag{5}$$

In this section we develop a numerical iterative scheme for computing approximations to the coefficients of $f(z)$. It is based on the theory of multishift LR iterations for the solution of the matrix eigenvalue problem [16,17].

Let us consider an $n \times n$ lower Hessenberg matrix A_1 with unit superdiagonal entries having $p(z)$ as its characteristic polynomial. A stationary multishift LR iteration for the computation of the eigenvalues of A_1, i.e., the roots of $p(z)$, is defined by the following rule:

$$\begin{aligned} &q(A_s) = L_s R_s, \\ &A_{s+1} = L_s^{-1} A_s L_s, \quad s \geqslant 1, \end{aligned} \tag{6}$$

where $q(z)$ is called the shift function and we tacitly assume that the matrix $q(A_s)$ can be factored at any step in the LR way, that is, as the product of a unit lower triangular matrix L_s by an upper triangular matrix R_s. It can be shown that the rate of convergence of the matrix iteration (6) depends on certain ratios associated with partitions in the spectrum of the shifted matrix $q(A_1)$. If $q(z)$ is a linear or a quadratic factor then (6) reduces to a some variant of the classical LR algorithm of Rutishauser. However, under the further assumption that $q(z)$ is defined on the spectrum of A_1, other different choices have been also investigated in the literature [16].

For our purposes, a natural choice is $q(z) = t_\alpha(z)$. In this way, separation of the eigenvalues in the right-half-plane from the eigenvalues in the left-half-plane takes place. Moreover, one step of

the matrix iteration (6) can be split into one standard LR step

$$\begin{aligned} &A_s - \alpha I = \hat{L}_s \hat{R}_s, \\ &A_{s+1/2} = \alpha I + \hat{R}_s \hat{L}_s, \quad s \geqslant 1, \end{aligned} \tag{7}$$

followed by an other standard RL step

$$\begin{aligned} &A_{s+1/2} + \alpha I = \hat{R}_{s+1/2} \hat{L}_{s+1/2}, \\ &A_{s+1} = \hat{L}_{s+1/2} \hat{R}_{s+1/2} - \alpha I, \quad s \geqslant 1. \end{aligned} \tag{8}$$

Since A_1 is a lower Hessenberg matrix with unit superdiagonal entries, then $\hat{R}_1$ is an upper bidiagonal matrix with unit superdiagonal entries and hence $A_{3/2}$ is of lower Hessenberg form with unit superdiagonal entries. The same arguments apply to (8) and, therefore, we may conclude that all the matrices A_s and $A_{s+1/2}$ are lower Hessenberg matrix with unit superdiagonal entries.

Now, let us introduce the polynomial vector

$$[\psi_0^{(s)}(z), \psi_1^{(s)}(z), \ldots, \psi_{n-1}^{(s)}(z)]^{\mathrm{T}},$$

where $\psi_0^{(s)}(z) = 1$ and $\psi_i^{(s)}(z)$ is the characteristic polynomial of the $i \times i$ leading principal submatrix of A_s. The following equation [8]:

$$z \begin{bmatrix} \psi_0^{(s)}(z) \\ \psi_1^{(s)}(z) \\ \vdots \\ \psi_{n-1}^{(s)}(z) \end{bmatrix} = A_s \begin{bmatrix} \psi_0^{(s)}(z) \\ \psi_1^{(s)}(z) \\ \vdots \\ \psi_{n-1}^{(s)}(z) \end{bmatrix} + \begin{bmatrix} 0 \\ \vdots \\ 0 \\ p(z) \end{bmatrix} \tag{9}$$

is the key property in order to express (7) and (8) in a polynomial setting.

Writing

$$z \begin{bmatrix} \psi_0^{(s+1/2)}(z) \\ \psi_1^{(s+1/2)}(z) \\ \vdots \\ \psi_{n-1}^{(s+1/2)}(z) \end{bmatrix} = \hat{L}_s^{-1} A_s \hat{L}_s \begin{bmatrix} \psi_0^{(s+1/2)}(z) \\ \psi_1^{(s+1/2)}(z) \\ \vdots \\ \psi_{n-1}^{(s+1/2)}(z) \end{bmatrix} + \begin{bmatrix} 0 \\ \vdots \\ 0 \\ p(z) \end{bmatrix}, \tag{10}$$

we find that

$$\begin{bmatrix} \psi_0^{(s+1/2)}(z) \\ \psi_1^{(s+1/2)}(z) \\ \vdots \\ \psi_{n-1}^{(s+1/2)}(z) \end{bmatrix} = \hat{L}_s^{-1} \begin{bmatrix} \psi_0^{(s)}(z) \\ \psi_1^{(s)}(z) \\ \vdots \\ \psi_{n-1}^{(s)}(z) \end{bmatrix}.$$

By replacing $A_{s+1/2}$ in (10) with $\alpha I + \hat{R}_s \hat{L}_s$, we obtain that

$$(z - \alpha) \begin{bmatrix} \psi_0^{(s+1/2)}(z) \\ \psi_1^{(s+1/2)}(z) \\ \vdots \\ \psi_{n-1}^{(s+1/2)}(z) \end{bmatrix} = \hat{R}_s \begin{bmatrix} \psi_0^{(s)}(z) \\ \psi_1^{(s)}(z) \\ \vdots \\ \psi_{n-1}^{(s)}(z) \end{bmatrix} + \begin{bmatrix} 0 \\ \vdots \\ 0 \\ p(z) \end{bmatrix}.$$

Similarly, from (8) we find that

$$(z+\alpha)\begin{bmatrix}\psi_0^{(s+1/2)}(z)\\ \psi_1^{(s+1/2)}(z)\\ \vdots\\ \psi_{n-1}^{(s+1/2)}(z)\end{bmatrix}=\hat{R}_{s+1/2}\begin{bmatrix}\psi_0^{(s+1)}(z)\\ \psi_1^{(s+1)}(z)\\ \vdots\\ \psi_{n-1}^{(s+1)}(z)\end{bmatrix}+\begin{bmatrix}0\\ \vdots\\ 0\\ p(z)\end{bmatrix}.$$

By combining these two relations we finally arrive at the following result:

$$(z-\alpha)\left\{\hat{R}_{s+1/2}\begin{bmatrix}\psi_0^{(s+1)}(z)\\ \psi_1^{(s+1)}(z)\\ \vdots\\ \psi_{n-1}^{(s+1)}(z)\end{bmatrix}+\begin{bmatrix}0\\ \vdots\\ 0\\ p(z)\end{bmatrix}\right\}=(z+\alpha)\left\{\hat{R}_s\begin{bmatrix}\psi_0^{(s)}(z)\\ \psi_1^{(s)}(z)\\ \vdots\\ \psi_{n-1}^{(s)}(z)\end{bmatrix}+\begin{bmatrix}0\\ \vdots\\ 0\\ p(z)\end{bmatrix}\right\}. \tag{11}$$

Equivalently, by recalling that both $\hat{R}_s$ and $\hat{R}_{s+1/2}$ are upper bidiagonal matrices with unit superdiagonal entries, we can write

$$\begin{aligned}&(z-\alpha)(\hat{r}_1^{(s+1/2)}\psi_0^{(s+1)}(z)+\psi_1^{(s+1)}(z))=(z+\alpha)(\hat{r}_1^{(s)}\psi_0^{(s)}(z)+\psi_1^{(s)}(z)),\\ &\ldots,\\ &\ldots,\\ &(z-\alpha)(\hat{r}_n^{(s+1/2)}\psi_{n-1}^{(s+1)}(z)+\psi_n^{(s+1)}(z))=(z+\alpha)(\hat{r}_n^{(s)}\psi_{n-1}^{(s)}(z)+\psi_n^{(s)}(z)),\end{aligned} \tag{12}$$

where $\hat{r}_i^{(s)}$, $\hat{r}_i^{(s+1/2)}$ denotes the diagonal entries of $\hat{R}_s$ and $\hat{R}_{s+1/2}$, respectively, and, for the sake of simplicity, we set $\psi_n^{(s)}(z)=p(z)$ for any s. The computation of the unknowns of (12) can be performed as follows. First, we observe that α is a root of the polynomials placed on the left-hand side of (12) from which there follows that

$$\hat{r}_i^{(s)}=-\frac{\psi_i^{(s)}(\alpha)}{\psi_{i-1}^{(s)}(\alpha)},\quad 1\leqslant i\leqslant n. \tag{13}$$

Then, $\hat{r}_{n+1-i}^{(s+1/2)}$ and the coefficients of $\psi_{n-i}^{(s+1)}(z)$ are recursively determined for $i=1,2,\ldots,n$ by means of a direct inspection of the coefficients of the auxiliary polynomials

$$\chi_i^{(s)}(z)=(z+\alpha)\left\{\frac{\hat{r}_i^{(s)}\psi_{i-1}^{(s)}(z)+\psi_i^{(s)}(z)}{z-\alpha}\right\}. \tag{14}$$

In view of these findings, we propose the following iterative scheme — called Factor Iteration — for approximating the coefficients of the stable factor $f(z)$ of degree $k\geqslant 1$ of a monic polynomial $p(z)$ of degree n.

Factor Iteration

Let α be a positive real number; moreover, for $1\leqslant i\leqslant n-k$, let $\psi_{n-i}^{(0)}(z)$ be a monic polynomial of degree $n-i$;

for $s=1,2,\ldots,$

1. compute $\hat{r}_{n-i+1}^{(s)}$, $1\leqslant i\leqslant n-k$, by means of (13);
2. evaluate the coefficients of the polynomials $\chi_{n-i}^{(s)}(z)$, $1\leqslant i\leqslant n-k$, defined by (14);

3. determine the coefficients of the monic polynomials $\psi_{n-i}^{(s+1)}(z)$, $1 \leqslant i \leqslant n-k$, by recursively using

$$\hat{r}_{n-i+1}^{(s+1/2)} \psi_{n-i}^{(s+1)}(z) = \chi_{n-i}^{(s)}(z) - \psi_{n-i+1}^{(s+1)}(z), \quad 1 \leqslant i \leqslant n-k,$$

where we set $\psi_n^{(s+1)}(z) = p(z)$.

end

The arithmetic cost of one iteration is $O(n(n-k))$. In particular, in the case where $n-k \ll n$, the arithmetic cost is almost linear with respect to the degree n of the input polynomial.

In [5] the author studied some numerical methods for the approximate factorization of complex polynomials that are based on *LR* matrix iterations with polynomial shifts. The convergence theory of these methods can be also applied to the Factor Iteration yielding the following result. We recall that the l-norm, $l \in \{1, 2, \infty\}$, of a polynomial $p(z)$ coincides with the l-norm of its coefficient vector.

Theorem 1. *Let $p(z)$ be a monic polynomial of degree n. For a fixed positive real number α, assume that its zeros λ_i, $1 \leqslant i \leqslant n$, are numbered so that* (4) *holds. Then, for almost any choice of the polynomials $\psi_{n-j}^{(0)}(z)$, $1 \leqslant j \leqslant n-k$, the Factor Iteration does not break down for any $s \in \mathbb{N}$. Moreover, we have that*

$$\left\| \psi_k^{(s+1)}(z) - \prod_{i=1}^{k} (z - \lambda_i) \right\|_\infty = O(\varepsilon^{s+1}),$$

where ε is any number satisfying

$$|t_\alpha(\lambda_{k+1})/t_\alpha(\lambda_k)| < \varepsilon < 1.$$

This theorem states the convergence of the polynomial sequence $\{\psi_k^{(s)}(z)\}$ to the stable factor $f(z)$ of $p(z)$. Analogous results can also be proved for the other sequences $\{\psi_{n-i}^{(s)}(z)\}$, $1 \leqslant i \leqslant k-1$, generated by the Factor Iteration whenever a separation $|t_\alpha(\lambda_{n-i})| > |t_\alpha(\lambda_{n-i+1})|$ occurs. In this way, at least virtually, our method may be applied in order to extract the roots of $p(z)$ with respect to their ordering in magnitude of $t_\alpha(\lambda_i)$.

It is clear that the convergence rate depends on the choice of the parameter α. A good selection needs some preliminary information about the root distribution of $p(z)$ in the complex plane, that is, about the separation between the wanted and unwanted spectrum. To establish a more precise claim, we assume that the roots $\lambda_{k+1}, \ldots, \lambda_n$ in the right-half-plane are on or inside the rectangular region

$$R_+ = \{z \in \mathbf{C}: 0 < x_1 \leqslant \Re(z) \leqslant x_2,\ 0 \leqslant |\Im(z)| < y_1,\ y_1 \geqslant x_2\}, \tag{15}$$

whereas the remaining roots $\lambda_1, \ldots, \lambda_k$ in the left-half-plane belong to the symmetric region

$$R_- = \{z \in \mathbf{C}: -x_2 \leqslant \Re(z) \leqslant -x_1,\ 0 \leqslant |\Im(z)| < y_1,\ y_1 \geqslant x_2\}. \tag{16}$$

We have performed numerical experiments in order to evaluate

$$\frac{\max\{|t_\alpha(\lambda_i)|: \lambda_i \in R_+\}}{\min\{|t_\alpha(\lambda_i)|: \lambda_i \in R_-\}}$$

for many different choices of α and of the distribution of the zeros of $p(z)$. They indicate that the smallness of the quantity

$$\cos\theta = \frac{x_1}{|x_1 + \mathrm{i}y_1|}$$

gives a rather precise measure of how difficult our problem is.

Indeed, some theoretical results can also be obtained. For a fixed value $\bar{\alpha} > 0$ of α, let us denote by $z(\bar{\alpha}) = a(\bar{\alpha}) + \mathrm{i}b(\bar{\alpha})$ a point of R_+ where the function $|t_{\bar{\alpha}}(z)|$ attains to its maximum. By calculus, we easily find that $b(\bar{\alpha}) = y_1$ and, moreover,

$$\rho(\bar{\alpha}) = \frac{\max\{|t_{\bar{\alpha}}(z)|\colon z \in R_+\}}{\min\{|t_{\bar{\alpha}}(z)|\colon z \in R_-\}} = \frac{|t_{\bar{\alpha}}(a(\bar{\alpha}) + \mathrm{i}y_1)|}{|t_{\bar{\alpha}}(-a(\bar{\alpha}) + \mathrm{i}y_1)|}.$$

The two points $a(\bar{\alpha})+\mathrm{i}y_1$ and $-a(\bar{\alpha})+\mathrm{i}y_1$ are symmetric with respect to the imaginary axis which is mapped into the unit circle by $t_{\bar{\alpha}}(z)$. In this way, using the fact that Moebius transformations preserve the symmetry of a pair of points with respect to generalized circles (see [6, Theorem 5.4d]), we are able to prove that $t_{\bar{\alpha}}(a(\bar{\alpha}) + \mathrm{i}y_1)$ and $t_{\bar{\alpha}}(-a(\bar{\alpha}) + \mathrm{i}y_1)$ are symmetric with respect to the unit circle. This means that these two points lie in the same ray emanating from the origin and that the product of their distances from the origin equals 1. Thus, we have that

$$|t_{\bar{\alpha}}(-a(\bar{\alpha}) + \mathrm{i}y_1)| - |t_{\bar{\alpha}}(a(\bar{\alpha}) + \mathrm{i}y_1)| = |t_{\bar{\alpha}}(a(\bar{\alpha}) + \mathrm{i}y_1) - t_{\bar{\alpha}}(-a(\bar{\alpha}) + \mathrm{i}y_1)|.$$

In order to obtain a lower bound on the distance between the images of the two considered points, we observe that

$$|t_{\bar{\alpha}}(a(\bar{\alpha}) + \mathrm{i}y_1) - t_{\bar{\alpha}}(-a(\bar{\alpha}) + \mathrm{i}y_1)| \geqslant \frac{4\bar{\alpha}a(\bar{\alpha})}{(|a(\bar{\alpha}) + \mathrm{i}y_1| + \bar{\alpha})^2},$$

from which it follows that

$$|t_{\bar{\alpha}}(a(\bar{\alpha}) + \mathrm{i}y_1) - t_{\bar{\alpha}}(-a(\bar{\alpha}) + \mathrm{i}y_1)| \geqslant \frac{2\bar{\alpha}a(\bar{\alpha})}{a(\bar{\alpha})^2 + y_1^2 + \bar{\alpha}^2} \geqslant \frac{2\bar{\alpha}x_1}{x_1^2 + y_1^2 + \bar{\alpha}^2} = \delta(\bar{\alpha}). \tag{17}$$

In particular, the choice $\bar{\alpha} = \sqrt{x_1^2 + y_1^2}$ is the one maximizing $\delta(\alpha)$ and it produces

$$\delta(\sqrt{x_1^2 + y_1^2}) = \cos\theta.$$

Finally, since the real function $\mathbb{R}^+ \times \mathbb{R}^+ \ni (x, y) \to x/(x + y)$ is increasing with respect to the variable x and decreasing with respect to y, we obtain that

$$\rho(\bar{\alpha}) \leqslant \frac{1}{1 + \delta(\bar{\alpha})}.$$

Summing up, the following result holds.

Theorem 2. *Let R_+ and R_- be the two subsets of $\mathbb{C}$ defined by* (15) *and* (16), *respectively. Assume that the n zeros λ_i of a complex polynomial $p(z)$ are such that $k \geqslant 1$ roots belong to R_- whereas the remaining $n - k \geqslant 1$ roots lie in R_+. Then, for any given $\alpha > 0$ we have that*

$$\frac{\max\{|t_\alpha(\lambda_i)|\colon \lambda_i \in R_+\}}{\min\{|t_\alpha(\lambda_i)|\colon \lambda_i \in R_-\}} \leqslant \frac{\max\{|t_\alpha(z)|\colon z \in R_+\}}{\min\{|t_\alpha(z)|\colon z \in R_-\}} \leqslant \frac{1}{1 + \delta(\alpha)},$$

where $\delta(\alpha)$ *is given by* (17). *Furthermore, if we set* $\alpha = \sqrt{x_1^2 + y_1^2}$ and $\cos\theta = x_1/|x_1 + \mathrm{i}y_1|$, *then we find that*

$$\frac{\max\{|t_\alpha(\lambda_i)|:\ \lambda_i \in R_+\}}{\min\{|t_\alpha(\lambda_i)|:\ \lambda_i \in R_-\}} \leqslant \frac{1}{1+\cos\theta}.$$

In the next section we will discuss an effective implementation of the Factor Iteration and we will present the results of our computational experience.

3. Computational results

We implemented different versions of the Factor Iteration using *Mathematica*^{TM}. In its basic form our code gets in input the coefficients of an n-degree real monic polynomial $p(z)$ together with an integer k that represents the degree of the stable factor $f(z)$ of $p(z)$. Then it generates $n-k$ real monic polynomials $\psi_{n-h}^{(0)}(z)$, $1 \leqslant h \leqslant n-k$, where $\psi_{n-h}^{(0)}(z)$ has degree $n-h$, and it starts with the computation of the Factor Iteration.

There are two possible ways of terminating the iteration for a factor. The first one uses a well-known property about the sign of the coefficients of $f(z)$ [6]. Observe that we can write

$$f(z) = \prod_i (z+\gamma_i) \prod_j (z+\beta_j)(z+\bar{\beta}_j),$$

where $-\gamma_i < 0$ are the real roots of $f(z)$ and $-\beta_j$, $-\bar{\beta}_j$ are the complex roots with $\Re(\beta_j) > 0$. Carrying on the multiplication we realize that all the coefficients of $f(z)$ are of the same sign of its leading coefficient, that is, they are positive. Hence, at the sth step our algorithm checks first if all the coefficients of the computed approximation $\psi_k^{(s)}(z)$ are positive. If this condition is satisfied then we also check if

$$\frac{\|\psi_k^{(s)}(z) - \psi_k^{(s-1)}(z)\|_2}{\|\psi_k^{(s)}(z)\|_2} \leqslant \eta,$$

where, at present, ad hoc choices are made for the threshold value η. If yes, then the algorithm is halted and $\psi_k^{(s)}(z)$ is the computed approximation for $f(z)$. On the contrary, when the number of iterations exceeds a fixed value *itmax*, the program stops by reporting failure.

Our code has been tested by performing numerical experiments on a computer using the equivalent of 15 decimal digits floating point arithmetic. It is clear that a good selection of the value of η is crucial and it should take into account some information about the conditioning of the problem. Regarding at this important numerical issue, let $p(z;\varepsilon) = f(z;\varepsilon)g(z;\varepsilon)$ be the factorization with respect to the imaginary axis of the n-degree perturbed polynomial $p(z;\varepsilon)$ where $p(z;0) = p(z)$ and $p(z)$ has no roots along the imaginary axis. By means of a linearization process, the new factors $f(z;\varepsilon)$ and $g(z;\varepsilon)$ can be thought of as determined by one step of the Newton iteration with initial guesses $f(z;0) = f(z)$ and $g(z;0) = g(z)$. In this case the Jacobian matrix is the resultant matrix of $f(z)$ and $g(z)$ and, therefore, its condition number gives an estimate of the conditioning of splitting $p(z)$. In our numerical tests we computed the spectral condition number of a matrix by means of the *Mathematica*^{TM} function *SingularValues*.

Table 1

degree $= j$	s_j	cond_j
31	0.35	36
29	0.87	2.4E + 04
28	0.64	5.1E + 05
24	0.78	1.2E + 07

A lot of numerical examples were run to investigate the stability properties of our method. We considered real polynomials $p(z)$ of degree 32 generated as the product of a real polynomial $q(z)$ of degree 16, with random coefficients between -0.5 and 0.5, by $q(-z)$. We chose $\alpha=1$, $\psi_{31}^{(0)}(z)=l_{31}\,p'(z)$ and $\psi_j^{(0)}(z)=l_j(\psi_{j+1}^{(0)}(z))'$, $16\leqslant j\leqslant 30$, where the coefficients l_j were determined in such a way that all the polynomials are monic. Next we computed the separation ratios $s_j = |t_1(\lambda_{j+1})/t_1(\lambda_j)|$, $1\leqslant j\leqslant 31$, where λ_j were the numerical approximations of the zeros of $p(z)$ obtained by means of the *Mathematica*^{TM} function *NSolve*. For each of the indices j such that $16\leqslant j\leqslant 31$ and $s_j < 1$, we also estimated the spectral condition cond_j of factoring $p(z)$ as $p(z)=\prod_{i=1}^{j}(z-\lambda_j)\prod_{h=j+1}^{n}(z-\lambda_h)$.

The resulting implementation was used to study the convergence of the sequences $\{\psi_j^{(s)}(z)\}$ towards the desired factor $f_j(z)=\prod_{i=1}^{j}(z-\lambda_j)$. In particular, in order to illustrate the behaviour of these approximations, we set the value of η equal to the machine precision so that the algorithm always terminated reporting failure. Below we report the results of a typical execution. Table 1 shows the estimated values of the condition number and of the separation ratio of the first 4 factors $f_j(z)$.

The plot of the logarithm to base 10 of the approximation errors

$$\varepsilon_j^{(s)} = \|\psi_j^{(s)}(z)-\psi_j^{(s-1)}(z)\|_2/\|\psi_j^{(s)}(z)\|_2, \quad j=24,28,29,31,$$

evaluated at the steps $s=4i$, $1\leqslant i\leqslant 16$, is described in Fig. 1.

This illustration suggests two important observations that are confirmed by our numerical experience. Firstly, the errors of the computed approximations to the desired factors are generally in accordance with the estimates on the conditioning. Secondly, despite of the recursive construction of the approximations, a rapid convergence can be observed at a certain level j independently of the behaviour of the approximations at the previous levels whenever $s_j \ll s_h$ for $h > j$. This property motivates the interest in developing numerical methods for the approximation of a factor.

The second set of numerical tests was concerned with the description of the root-locus of certain real polynomials formed from the odd and the even parts of Hurwitz polynomials. These polynomials are generated by many recursive algorithms for signal processing [10]. Specifically, we considered Hurwitz polynomials $p(z)$ of degree 32 obtained as follows:

$$p(z)=\prod_{i=1}^{16}(z^2+2za_i+a_i^2+b_i^2), \tag{18}$$

where a_i and b_i are drawn from the uniform distribution in $[0,1]$.

The component parts of $p(z)$ are given by

$$q_{\text{even}}(z)=0.5(p(z)+p(-z)), \quad q_{\text{odd}}(z)=0.5(p(z)-p(-z)).$$

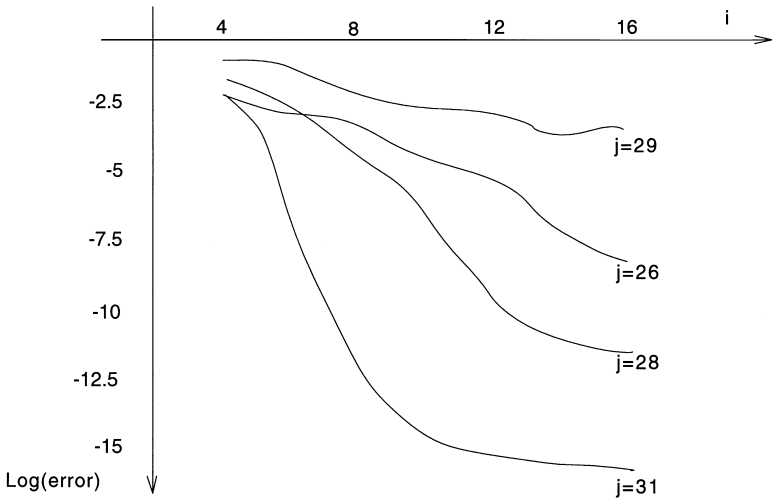

Fig. 1.

If we combine these parts by forming the family of new polynomials

$$\hat{p}(z; w_1; w_2) = q_{\text{even}}(z) + (w_1 + w_2 z) q_{\text{odd}}(z),$$

where w_1 and w_2 are real parameters, then it can be shown that the zero distribution of $\hat{p}(z; w_1; w_2)$ in the complex plane is completely defined by the sign of w_1 and of $w_2 + (lc_{\text{even}}/lc_{\text{odd}})$, where lc_{even} and lc_{odd} denote the leading coefficient of $q_{\text{even}}(z)$ and of $q_{\text{odd}}(z)$, respectively. In particular, $\hat{p}(z; w_1; w_2)$ has 31 zeros in the left-half-plane and 1 zero in the right-half-plane whenever w_1 is positive and $w_2 < -(lc_{\text{even}}/lc_{\text{odd}})$. Moreover, the positive zero, denoted by $\lambda_n(w_1; w_2)$, tends to the infinity as w_2 approaches $-(lc_{\text{even}}/lc_{\text{odd}})$.

Our numerical experiments returned a plot of the function $\lambda_n(t; -(lc_{\text{even}}/lc_{\text{odd}})(2-t))$ with the variable t varying from 0 to 1. To do this, we defined a sequence of equispaced nodes $t_i = i/50$, $1 \leqslant i \leqslant 49$, in the open interval $(0, 1)$ and then we determined numerical approximations $\hat{\lambda}_n(t_i; -(lc_{\text{even}}/lc_{\text{odd}})(2-t_i))$ of the zeros $\lambda_n(t_i; -(lc_{\text{even}}/lc_{\text{odd}})(2-t_i))$ by means of our program. If $i = 1$, we set $\alpha = 1$ and $\psi_{31}^{(0)}(z) = l_{31} p'(z)$. Otherwise, if $i > 1$, we performed the selection $\alpha = \hat{\lambda}_n(t_{i-1}; -(lc_{\text{even}}/lc_{\text{odd}})(2-t_{i-1}))$ and

$$\psi_{31}^{(0)}(z) = \text{PolynomialQuotient}[p(z), z - \hat{\lambda}_n(t_{i-1}; -(lc_{\text{even}}/lc_{\text{odd}})(2-t_{i-1}))],$$

where PolynomialQuotient$[p(z), q(z)]$ gives the result of dividing $p(z)$ by $q(z)$ with any remainder dropped.

Concerning the stop criterion, a little modification of the basic procedure was implemented. At each step s, first we computed a suitable approximation $r_i^{(s)}$ of $\lambda_n(t_i; -(lc_{\text{even}}/lc_{\text{odd}})(2-t_i))$ according to the following rule (cf. (9)):

$$r_i^{(s)} = \text{lc}(z\psi_{31}^{(s)}(z) - p(z)),$$

where lc$(p(z))$ denotes the leading coefficient of $p(z)$. Then we calculated the value $\delta_i^{(s)} = 32|p(r_i^{(s)})/p'(r_i^{(s)})|$. A classical result [6] states that there is a zero r_i of $p(z; t_i; -(lc_{\text{even}}/lc_{\text{odd}})(2-t_i))$ such that

$$|r_i - r_i^{(s)}| \leqslant \delta_i^{(s)}.$$

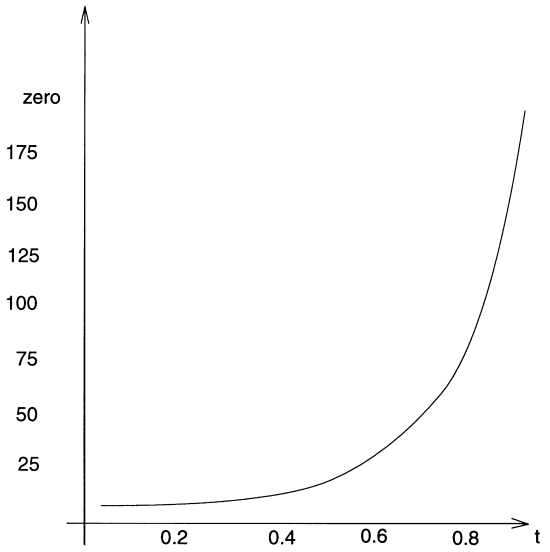

Fig. 2.

Thus, the computation for the ith zero was stopped if the condition $\delta_i^{(s)} \leqslant \| p(z; t_i; -(lc_{\text{even}}/lc_{\text{odd}})(2 - t_i))\|_2 \varepsilon$, where ε denotes here the machine precision, was satisfied. In this case, $r_i^{(s)} = \hat{\lambda}_n(t_i; -(lc_{\text{even}}/lc_{\text{odd}})(2 - t_i))$ was the numerical approximation of $\lambda_n(t_i; -(lc_{\text{even}}/lc_{\text{odd}})(2 - t_i))$ to be used in the following steps.

In order to accelerate the convergence we also complemented the basic procedure with a suitable shift strategy. If we found that

$$|r_i^{(s)} - r_i^{(s-1)}| \leqslant \sqrt{\| p(z; t_i; -(lc_{\text{even}}/lc_{\text{odd}})(2 - t_i))\|_2 \varepsilon},$$

then the value of α was changed to $r_i^{(s)}$ into the subsequent iterations for the ith zero.

Fig. 2 illustrates a characteristic path of the positive zero obtained in one numerical experiment. As it is theoretically proved, the trajectory goes to the infinity when t_i approaches to 1.

In all the numerical experiments we have carried out the program never reported failure. The number of iterations needed to approximate the positive zero of $p(z; t_i; -(lc_{\text{even}}/lc_{\text{odd}})(2-t_i))$ usually increases together with the value of i. This is especially true as the point t_i is near to 1 because of the very rapid deterioration of the separation ratio. For instance, in the numerical test of Fig. 2 we found that the average number of iterations was 11 for the computation of the first 35 zeros whereas it was 63 for the computation of the 14 remaining zeros.

Finally, we planned a third set of numerical experiments in order to verify if our algorithm can be used as a stability-test procedure. In particular, we considered the problem of recognizing whether a given real polynomial is Hurwitz. It can be shown [15] that if $p(z)$ is a Hurwitz polynomial then $\| p(z)\|_1 \geqslant 2^{32}\sqrt{a_0}$, where a_0 denotes the known coefficient of $p(z)$. This fact ensures that many polynomials can not be Hurwitz even though they have positive coefficients. Moreover, since the 1-norm of $p(z)$ increases exponentially with the degree the same holds for the conditioning of splitting $p(z)$ notwithstanding that one factor has a small degree.

To support these statements numerically, we generated 100 Hurwitz polynomials $p(z)$ of degree 32 by the rule (18). The condition number of determining a quadratic factor of $p(z)$ with $\alpha = 0.1$ was 10^{14} in the average. It is interesting to point out that the roots of each polynomial were usually computed at high precision by the *NSolve* function. This means that the condition number of the factorization problem is not related to the condition numbers of the zeros and, in addition, different arrangements of the zeros lead to different factorizations whose condition numbers can vary greatly.

In conclusion, we have proposed a numerical algorithm for the approximate Hurwitz factorization of a polynomial $p(z)$. Our numerical experiments indicates that it can be used to determine efficiently the coefficients of the two factors $f(z)$ and $g(z)$ whenever one factor has a small degree. If, otherwise, $p(z)$ has a quite balanced zero distribution with respect to the imaginary axis, then the conditioning of the Hurwitz factorization problem generally increases in such a way that the problem cannot be solved in a satisfactory way by finite precision arithmetic computations. We also arrive at similar conclusions when we apply our algorithm to Hurwitz polynomials as a stability-test procedure. To our experience, in these latter two cases using some numerical algorithm for the computation of all the roots of $p(z)$ with respect to a different ordering, like the one that is introduced by a random shift of the origin, usually provides a better approach.

References

[1] S. Barnett, Polynomials and Linear Control Systems, Marcel Dekker, New York, 1983.
[2] F.L. Bauer, Ein Direktes Iterationsverfharen zur Hurwitz Zerlegung eines Polynoms, Archiv Elektrischen Ubertragung 9 (1955) 285–290.
[3] J.P. Cardinal, On two iterative methods for approximating the roots of a polynomial, in: J. Renegar, M. Shub, S. Smale (Eds.), Proceedings of Workshop: Mathematics of Numerical Analysis: Real Numbers Algorithms, Park City, Ut, 1995, Lectures in Applied Mathematics, American Mathematical Society, Providence, RI.
[4] L. Gemignani, Polynomial root computation by means of the LR algorithm, BIT 37 (1997) 333–345.
[5] L. Gemignani, Computing a factor of a polynomial by means of multishift LR algorithms, SIAM J. Matrix Anal. Appl. 19 (1998) 161–181.
[6] P. Henrici, Applied and Computational Complex Analysis, Vol. 1, Wiley, New York, NY, 1974.
[7] A.S. Householder, The Numerical Treatment of a Single Nonlinear Equation, International Series in Pure and Applied Mathematics, McGraw-Hill, New York, 1970.
[8] J. Kautsky, G.H. Golub, On the calculation of Jacobi matrices, Linear Algebra Appl. 52/53 (1983) 439–455.
[9] P. Kirrinnis, Partial fraction decomposition in $\mathbf{C}(z)$ and simultaneous Newton iteration for factorization in $\mathbf{C}[z]$, J. Complexity, in preparation.
[10] A. Lepschy, U. Viaro, Derivation of recursive stability-test procedure, Circuits Systems Signal Process 13 (1994) 615–623.
[11] M. Marden, Geometry of Polynomials, American Mathematical Society, Providence, RI, 1966.
[12] V.Y. Pan, Optimal and nearly optimal algorithms for approximating polynomial zeros, Comput. Math. Appl. 31 (1996) 97–138.
[13] A. Schoenhage, The fundamental theorem of algebra in terms of computational complexity, manuscript, 1982.
[14] G.W. Stewart, On a companion operator for analytic functions, Numer. Math. 18 (1971) 26–43.
[15] V. Trevisan, Recognition of Hurwitz polynomials, SIGSAM Bull. 24 (1990) 26–32.
[16] D.S. Watkins, QR-like algorithms — an overview of convergence theory and practice, in: J. Renegar, M. Shub, S. Smale (Eds.), Proceedings of Workshop: Mathematics of Numerical Analysis: Real Numbers Algorithms, Park City, Utah, 1995, Lectures in Applied Mathematics, American Mathematical Society, Providence, RI.
[17] D.S. Watkins, L. Elsner, Convergence of algorithms of decomposition type for the eigenvalue problem, Linear Algebra Appl. 143 (1991) 19–47.