A Faster, More Stable Method for Computing the $p$th Roots of Positive Definite Matrices

W. D. Hoskins
Department of Computer Science
University of Manitoba
Winnipeg, Manitoba, Canada

and

D. J. Walton
Department of Mathematical Sciences
Lakehead University
Thunder Bay, Ontario, Canada

Submitted by F. Robert

ABSTRACT

An accelerated, more stable generalization of Newton's method for finding matrix $p$th roots is developed in a form suitable for finding the positive definite $p$th root of a positive definite matrix. Numerical examples are given and compared with the corresponding Newton iterates.

1. INTRODUCTION

The process outlined in [1] for determining the positive definite square root of a positive definite matrix is an accelerated version of the process due to Beaver and Denman [2] and is clearly a generalization of the basic Newton process suitable for computing with economy the concurrent iterates of a set of values, all simultaneously converging to square roots. The ensuing analysis demonstrates how this latter process can be obtained and viewed as a specific case of the more general problem of determining economically the $p$th roots of each of a given set of positive values.

2. DEVELOPMENT

Consider $n \times n$ matrices $X$ and $Y$ with positive spectra satisfying the equations

$$X^{p-1}Y = I \quad (2.1)$$

LINEAR ALGEBRA AND ITS APPLICATIONS 28:139–163 (1979) 139
© Elsevier North Holland, Inc., 1979 0024-3795/79/040139 + 25 $01.75
140

W. D. HOSKINS AND D. J. WALTON

and

\[ X = AY, \] (2.2)

where \( A \) is a given \( m \times m \) matrix with positive spectrum whose \( p \)th root \((p > 0 \text{ and integral})\) must be determined, and \( I \) is the \( m \times m \) identity.

From Eqs. (2.1) and (2.2) it follows that provided the matrices \( X \) and \( Y \) are rational functions of the matrix \( A \), then

\[ A^{p-1}Y = I. \]

Hence

\[ Y^p = A^{1-p}, \] (2.3)

and of course

\[ X^p = A. \] (2.4)

The usual Newton iteration for solving Eqs. (2.3) and (2.4) gives

\[ X_{n+1} = \frac{1}{p} \left[ (p-1)X_n + AX_n^{1-p} \right], \] (2.5)

and

\[ Y_{n+1} = \frac{1}{p} \left[ (p-1)Y_n + A^{1-p}Y_n^{1-p} \right]. \] (2.6)

Obviously, since Eqs. (2.5) and (2.6) generate matrices which are all rational functions of the initial approximations \( X_0 \) and \( Y_0 \), convergence of the iterates is easily examined by transforming the iterates to diagonal form and then considering the convergence of the set of scalar equations by standard methods [3].

The relations (2.5) and (2.6) give sequences \( X_0, X_1, X_2, \ldots \) and \( Y_0, Y_1, Y_2, \ldots \) that converge to the matrices \( A^{1/p} \) and \( A^{(1-p)/p} \) respectively with quadratic convergence once the iterates are close enough to the limit values.

A better method is obviously obtained if Eqs. (2.5) and (2.6) are changed to

\[ X_{n+1} = \alpha_n X_n + \beta_n A X_n^{1-p} \] (2.7)

and

\[ Y_{n+1} = \alpha_n Y_n + \beta_n A^{1-p} Y_n^{1-p}, \] (2.8)
where the scalars $\alpha_n$ and $\beta_n$ are chosen so that (1) convergence is accelerated and (2) the limit of the sequence $(\alpha_0, \beta_0), (\alpha_1, \beta_1), \ldots$ is $((p-1)/p, 1/p)$. Thus if $X_k, Y_k$ lie outside the region whose convergence for the Newton iteration is quadratic, judicious choice of $\alpha_k, \beta_k$ can considerably change the speed of convergence.

Evaluating the two recurrence relations (2.7) and (2.8) would seem to have little merit if only the $p$th root is required. However, for matrices the sequence $X_0, X_1, X_2, \ldots$ does not appear to be as “well behaved” as for their scalar counterparts.

**TABLE 1**

<table>
<thead>
<tr>
<th>Iteration</th>
<th>(1,1) element</th>
<th>Newton</th>
<th>New Method</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>9.7</td>
<td>1.4</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>6.5</td>
<td>1.6</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>4.5</td>
<td>2.03</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>3.2</td>
<td>1.9994</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>2.5</td>
<td>2.0000003</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>2.1</td>
<td>2</td>
<td></td>
</tr>
</tbody>
</table>

Using the Newton iteration given by Eq. (2.5) to determine the cube root of

$$A = \begin{bmatrix} 14 & 14 & 6 \\ 14 & 20 & 14 \\ 6 & 14 & 14 \end{bmatrix}$$

gave on an IBM 370/168 in vsapl, using the built-in double precision matrix inversion operator $[+]$, a sequence of iterates where the element in the 1,1 position was as given by Table 1. The extra work required to evaluate the two sequences (2.7) and (2.8) would seem to be well justified, if only for the reason that stability can be improved. However, convergence is also considerably faster, and the two recurrence relations may be rewritten in a more efficient computational form by using Eq. (2.2), viz.

$$X_{n+1} = \alpha_n X_n + \beta_n X_n^{2-p} Y_n^{-1}, \quad (2.9)$$

$$Y_{n+1} = \alpha_n Y_n + \beta_n X_n^{1-p} \quad (2.10)$$

$$X_0 = A, \quad Y_0 = I.$$
where neither relation involves explicitly the matrix whose pth root is required. However, the two relations (2.9) and (2.10) are now coupled rather than being independent as they were formerly [Eqs. (2.7) and (2.8)], and consequently the initial estimates $X_0, Y_0$ must satisfy the equation (2.2). Furthermore, since the ensuing analysis requires commutativity of products, it is necessary that the matrices $X_0, Y_0$ be rational functions of the identity matrix and the matrix $A$.

3. DETERMINATION OF THE ACCELERATION PARAMETERS

The matrices $X_n$ and $Y_n$ satisfy the relation (2.2) exactly; however, they do not satisfy (except in the limit) the equation (2.1), and thus at each stage this equation can be used to determine $\alpha_n, \beta_n$, i.e.,

$$C_{n+1} = X_{n+1}^{p-1} Y_{n+1},$$

(3.1)

or using Eqs. (2.9) and (2.10),

$$C_{n+1} = (\alpha_n X_n + \beta_n X_n^{2-p} Y_n^{1-p})^{p-1} (\alpha_n Y_n + \beta_n X_n^{1-p}).$$

(3.2)

Now since $C_n, X_n, Y_n$ are rational functions of the same matrix, it follows that the $k$th eigenvalue of the matrix on the left hand side is precisely the function on the right hand side with $X_n$ and $Y_n$ replaced respectively by their $k$th eigenvalues.

Let $c(x, y)$ be the $k$th eigenvalue of $C_{n+1}$ and $x, y$ be the $k$th eigenvalues of $X_n, Y_n$ respectively. Then

$$c(x, y) = (\alpha_n x + \beta_n x^{2-p} y^{1-p})^{p-1} (\alpha_n y + \beta_n x^{1-p}),$$

or

$$c(x, y) = \frac{(\alpha_n x^{p-1} y + \beta_n)^p}{(x^{p-2} y)^{p-1} x^{p-1}}.$$
with finally

\[ c(x, y) = \frac{(a_n x^{p-1} y + \beta_n)^p}{(x^{p-1} y)^{p-1}}. \]

Now if we substitute

\[ \lambda = x^{p-1} y \text{ and } \lambda \in [a_n, b_n] \quad \text{(with } a_n < b_n), \]

then

\[ c(\lambda) = \frac{(a_n \lambda + \beta_n)^p}{\lambda^{p-1}}. \quad (3.3) \]

However, from Eqs. (2.1) and (3.1) it is apparent that if the sequences \( X_n, Y_n \) converge, then \( C_n+1 \) is an approximation to the identity matrix \( I \), and hence \( a_n \) and \( \beta_n \) can be chosen so that the spectrum of \( C_n+1 \) is as close to unity as possible. One method of determining \( a_n, \beta_n \) is by definition such that

\[
\min_{a_n, \beta_n} \max_{\lambda \in [a_n, b_n]} \left| \frac{(a_n \lambda + \beta_n)^p}{\lambda^{p-1}} - 1 \right| = \varepsilon_{n+1}, \quad (3.4)
\]

and then

\[ a_{n+1} = 1 - \varepsilon_{n+1}, \quad (3.5) \]

\[ b_{n+1} = 1 + \varepsilon_{n+1}, \]

with initially

\[ a_0 = \left\{ \frac{1}{\rho(A^{-1})} \right\}^{p-1}, \quad (3.6) \]

\[ b_0 = \left\{ \rho(A) \right\}^{p-1}, \]

where \( \rho(A) \) is the spectral radius of the matrix \( A \).
If this method of choosing $\alpha_n, \beta_n$ is used, then these parameters are determined by forcing $c(\lambda)$ to have as extreme values $1 - \epsilon_{n+1}, 1 + \epsilon_{n+1}$ and $1 - \epsilon_{n+1}$ at abscissae $a_n, \xi_n, b_n$ respectively, where $\xi_n$ is the only stationary point of $c(\lambda)$ with a positive abscissa and $a_n \leq \xi_n \leq b_n$ (see Fig. 1).

In detail the process is as follows:

$$c'(\lambda) = \alpha_n p \frac{(\alpha_n \lambda + \beta_n)^{p-1}}{\lambda^{p-1}} - (p-1) \frac{(\alpha_n \lambda + \beta_n)^p}{\lambda^p},$$
and the stationary points are obtained from

\[ 0 = \frac{\left(\alpha_n \lambda + \beta_n\right)^{p-1}}{\lambda^p} \{ \alpha_n \lambda p - \left( p - 1 \right) \left( \alpha_n \lambda + \beta_n \right) \}, \]

i.e.,

\[ \lambda_1 = -\frac{\beta_n}{\alpha_n} \quad (3.7) \]

and

\[ \lambda_2 = \frac{\left( p - 1 \right) \beta_n}{\alpha_n} \quad (3.8) \]

Now

\[ c(\lambda_1) = 0 \]

for any combination of \( \alpha_n, \beta_n \); hence \( \lambda_1 \) is not useful for constraining selection of \( \alpha_n, \beta_n \). Also it can be observed from Eqs. (2.9) and (2.10) that from the standpoint of numerical stability the parameters \( \alpha_n, \beta_n \) should both be positive, so that \( \lambda_1 \) will be negative and therefore not belong to \([a_n, b_n]\).

The second root, \( \lambda_2 \), corresponds to a minimum value of the function \( c(\lambda) \), since

\[ c''(\lambda) = c'(\lambda) \left\{ \frac{\alpha_n \lambda - \left( p - 1 \right) \beta_n}{\alpha_n \lambda + \beta_n} \right\} + c(\lambda) \left\{ \frac{p - 1}{\lambda^2} - \frac{\alpha_n^2 p}{\left( \alpha_n \lambda + \beta_n \right)^2} \right\}, \]

and thus

\[ c''(\lambda_2) = c(\lambda_2) \frac{\alpha_n^2 \beta_n}{\beta_n^2} \left( \frac{1}{p - 1} - \frac{1}{p} \right), \]

so that

\[ c''(\lambda_2) > 0 \]

provided

\[ \alpha_n > 0, \]
\[ \beta_n > 0, \quad (3.9) \]
\[ b_n > \lambda_2 > a_n. \]
(See Fig. 1.) Now

\[ c(\lambda_2) = \left[ \frac{\alpha_n (p-1)\beta_n/\alpha_n + \beta_n}{(p-1)^{p-1}\beta_n^{p-1}/\alpha_n^{p-1}} \right]^p, \]

or

\[ c(\lambda_2) = \alpha_n^{p-1}\beta_n\frac{p^p}{(p-1)^{p-1}}, \]

and we require that

\[ c(\xi) = c(\lambda_2) = 1 - \varepsilon_n. \]

Thus

\[ \alpha_n^{p-1}\beta_n\frac{p^p}{(p-1)^{p-1}} = 1 - \varepsilon_n, \quad (3.10) \]

and of course

\[ c(\alpha_n) = 1 + \varepsilon_n \]

and

\[ c(\beta_n) = 1 + \varepsilon_n, \]

leading to

\[ \frac{(\alpha_n\beta_n + \beta_n)^p}{\alpha_n^{p-1}} = 1 + \varepsilon_n \quad (3.11) \]

and

\[ \frac{(\alpha_n\beta_n + \beta_n)^p}{\beta_n^{p-1}} = 1 + \varepsilon_n. \quad (3.12) \]
Equation (3.11) divided by (3.12) gives

\[ \alpha_n q_n + \beta_n = \left( \frac{a_n}{b_n} \right)^{(p-1)/p} (\alpha_n b_n + \beta_n), \]

and then

\[ \alpha_n = \left( \frac{\theta - 1}{a_n - \theta b_n} \right) \beta_n, \]

where \( \theta = (a_n / b_n)^{(p-1)/p} \), or in abbreviated form,

\[ \alpha_n = \gamma_n \beta_n, \quad (3.13) \]

where

\[ \gamma_n = \frac{a_n^{(p-1)/p} - b_n^{(p-1)/p}}{a_n b_n^{(p-1)/p} - a_n b_n^{(p-1)/p} b_n}. \quad (3.14) \]

Clearly \( \gamma_n > 0 \) for \( b_n > a_n \), and hence \( \alpha_n, \beta_n \) are either both positive or both negative.

Now combining Eqs. (3.10), (3.11) and (3.12) yields, with Eq. (3.13),

\[ \beta_n^p = \frac{4}{(p-1)^{p-1} + \frac{(1 + \gamma_n a_n)^p}{a_n^{p-1}} + \frac{(1 + \gamma_n b_n)^p}{b_n^{p-1}}} \quad (3.15) \]

and

\[ \varepsilon_n = \frac{\beta_n^p}{4} \left\{ \frac{(1 + \gamma_n a_n)^p}{a_n^{p-1}} + \frac{(1 + \gamma_n b_n)^p}{b_n^{p-1}} - \frac{2 p^p \gamma_n^{p-1}}{(p-1)^{p-1}} \right\} \quad (3.16) \]

From Eq. (3.15) it is apparent that \( b_n > a_n \) implies \( \beta_n > 0 \) and \( \alpha_n > 0 \) \((\forall p)\). Furthermore, Eqs. (3.10), (3.11) and (3.12) now imply that

\[ 1 - \varepsilon_n > 0 \]
and

\[ 1 + \varepsilon_n \geq 0, \]

i.e.,

\[ |\varepsilon_n| < 1 \]

However, rearrangement of Eq. (3.14) shows that

\[
\gamma_n = \frac{1}{(a_n b_n)(p-1)/p} \left( \frac{a_n^{(p-1)/p} - b_n^{(p-1)/p}}{a_n^{1/p} - b_n^{1/p}} \right),
\]

or

\[ \gamma_n > 0 \quad \forall \text{ positive } a, b. \]

Hence

\[ (\alpha_n, \beta_n) > (0,0), \]

and from Eq. (3.10), since

\[
\varepsilon_n = 1 - \alpha_n^{p-1} \beta_n \frac{p^p}{(p-1)^{p-1}},
\]

we have

\[ \varepsilon_n < 1. \]

Finally, the assumption that \( \lambda_2 \) as given by Eq. (3.8) is such that

\[ a_n \leq \lambda_2 \leq b_n \]

requires that

\[ a_n \leq (p-1) \frac{\beta_n}{\alpha_n} \leq b_n. \]

From Eq. (3.13)

\[ a_n \leq \frac{p-1}{\gamma_n} \leq b_n, \]
and using (3.17),

\[ a_n \leq \frac{(p-1)(a_n b_n)^{(p-1)/p}(a_n^{1/p} - b_n^{1/p})}{a_n^{(p-1)/p} - b_n^{(p-1)/p}} \leq b_n. \]

This may be restated as the two inequalities

\[ a^{2p-1} - a^p b^p \geq (p-1)(a^p b^{p-1} - a^{p-1} b^p) \]

and

\[ (p-1)(a^p b^{p-1} - a^{p-1} b^p) \geq a^{p-1} b^p - b^{2p-1}, \]

where

\[ a = a_n^{1/p} \quad \text{and} \quad b = b_n^{1/p}. \]

Now regrouping terms yields

\[ a^{2p-1} - p a^p b^{p-1} + (p-1)a^{p-1} b^p \geq 0 \]

and

\[ b^{2p-1} - p a^{p-1} b^p + (p-1)a^p b^{p-1} > 0, \]

or since \( a, b > 0, \)

\[ a^p - p a b^{p-1} + (p-1)b^p \geq 0, \quad (3.20) \]

\[ b^p - p a^{p-1} b + (p-1)a^b > 0. \]

Let \( b = ra, \) where \( r \geq 1, \) so that the inequalities (3.20) become

\[ 1 - pr^{p-1} + (p-1)r^p \geq 0, \quad (3.21) \]

\[ r^p - pr + (p-1) > 0, \]

which are obviously both satisfied for \( r \geq 1. \)
4. ANALYSIS OF THE CONVERGENCE OF THE NEW ALGORITHM

The convergence of the algorithm can be examined by considering Eq. (3.16). This equation may be written as

\[
\varepsilon_n = \frac{\beta_n^p}{4} \left\{ f(a_n) - 2f(\lambda_2) + f(b_n) \right\},
\]

where

\[
f(x) = \frac{(1 + \gamma_n x)^p}{x^{p-1}}.
\]

The quantity \(\beta_n^p\) may be bounded from Eq. (3.15) once the following intermediate results are obtained.

**Theorem 1.** If \(b_n = r_n a_n\) and \(r = r_n^{1/p}\), then

\[
\gamma_n = \frac{1}{r^{p-1} a_n} \left( \frac{r^{p-1} - 1}{r - 1} \right) > \frac{p - 1}{b_n}.
\]

**Proof.** From Eq. (3.14), by substitution,

\[
\gamma_n = \frac{1}{r^{p-1} a_n} \left( \frac{r^{p-1} - 1}{r - 1} \right).
\]

But \(r > 1\) and \(r a_n = b_n\); hence

\[
\gamma_n > \frac{p - 1}{b_n}.
\]

**Theorem 2.**

\[
\frac{1 + \gamma_n a_n^p}{a_n^{p-1}} > \frac{p^p}{b_n^{p-1}}.
\]
Proof. Using Theorem 1,

\[ \gamma_n = \frac{1}{r^{p-1} a_n \left( \frac{r^{p-1} - 1}{r - 1} \right)}. \]

Thus

\[ 1 + \gamma_n a_n = \frac{r^p - r^p 1 + r^p 1 - 1}{r^{p-1} (r - 1)}, \]

i.e.,

\[ \left( 1 + \gamma_n a_n \right)^p = \frac{1}{(a_n r^p)^{p-1}} \left( \frac{r^p - 1}{r - 1} \right)^p, \]

so that

\[ \frac{(1 + \gamma_n a_n)^p}{a_n^{p-1}} = \frac{1}{b_n^{p-1}} \left( \frac{r^p - 1}{r - 1} \right)^p > \frac{p^p}{b_n^{p-1}}, \]

since \( r > 1 \).

\[ \hfill \blacksquare \]

Theorem 3.

\[ \frac{(1 + \gamma_n b_n)^p}{b_n^{p-1}} > \frac{p^p}{b_n^{p-1}}. \]

Proof. From Theorem 1,

\[ \gamma_n > \frac{p - 1}{b_n}. \]

Therefore

\[ 1 + \gamma_n b_n > p. \]
so that
\[
\frac{(1 + \gamma_n b_n)^p}{b_n^{p-1}} > \frac{p^p}{b_n^{p-1}}.
\]

Substituting the appropriate bounds into Eq. (3.15) gives
\[
\beta_n^p < \frac{4}{4p^p / b_n^{p-1}},
\]
i.e.,
\[
\beta_n^p < \frac{b_n^{p-1}}{p^p} < \frac{1}{2},
\]
(4.3)
since \( p > 2 \), and since \( b_n < 2 \) for \( n > 0 \). Referring now to Eq. (4.1), it can be seen that use of the inequality (4.3) enables this equation to be rewritten as the inequality
\[
\varepsilon_n < \frac{1}{2} \left\{ f(a_n) - 2f(\lambda_2) + f(b_n) \right\}.
\]
(4.4)
Now expressing \( f(a_n) \) and \( f(b_n) \) using Taylor series yields
\[
f(a_n) = f(\lambda_2) + (a_n - \lambda_2)f'(\lambda_2) + \frac{(a_n - \lambda_2)^2}{2!} f''(\lambda_2) + \cdots
\]
and
\[
f(b_n) = f(\lambda_2) + (b_n - \lambda_2)f'(\lambda_2) + \frac{(b_n - \lambda_2)^2}{2!} f''(\lambda_2) + \cdots,
\]
so substitution in (4.4) gives
\[
\varepsilon_n < \frac{1}{2} \left\{ f'(\lambda_2)[a_n + b_n - 2\lambda_2] + \frac{1}{2} \left[ (a_n - \lambda_2)^2 + (b_n - \lambda_2)^2 \right] f''(\lambda_2) + \cdots \right\}.
\]
(4.5)
However from Eq. (3.8) and Eq. (4.2)
\[
f'(\lambda_2) = \frac{\gamma_n p(1 + \gamma_n \lambda_2)^p - 1}{\lambda_2^{p-1}} - (p - 1) \frac{(1 + \gamma_n \lambda_2)^p}{\lambda_2^p},
\]
or

\[ f\left( \frac{p-1}{\gamma_n} \right) = \frac{\gamma_n p^p}{(p-1)^{p-1}} - (p-1)p^{p-1} \gamma_n = 0. \]

Thus the relation (4.5) can be written

\[ \varepsilon_n < \frac{1}{8} \left[ \frac{1}{2} \left( (a_n - \lambda_2)^2 + (b_n - \lambda_2)^2 \right) f''(\lambda_2) + \cdots \right], \]

and since \( b_n - a_n = 2\varepsilon_{n-1} \) and \( a_n < \lambda_2 < b_n \),

\[ \varepsilon_n < \frac{1}{4} \varepsilon_{n-1}^2 f''(\lambda_2) + \frac{1}{6} \varepsilon_{n-1}^3 f'''(\lambda_2) + \cdots. \quad (4.6) \]

Thus it can be seen that asymptotically the new method is of second order. Convergence in general can be established by proving that

\[ \frac{1 + \varepsilon_k}{1 - \varepsilon_k} < \frac{1 + \varepsilon_{k-1}}{1 - \varepsilon_{k-1}}, \]

in the following way.

From Eqs. (3.10) and (3.11),

\[ 1 + \varepsilon_k = \frac{(\alpha_k a_k + \beta_k)^p}{a_k^{p-1}}, \]

\[ 1 - \varepsilon_k = \frac{\alpha_k^{p-1} \beta_k p^p}{(p-1)^{p-1}}, \]

so that

\[ \frac{1 + \varepsilon_k}{1 - \varepsilon_k} = \frac{(1 + \gamma_k a_k)^p (p-1)^{p-1}}{a_k^{p-1} \gamma_k^{p-1} p^p}, \quad (4.7) \]

on using Eq. (3.13). Now from Theorem 2

\[ (1 + \gamma_k a_k)^p = \left( \frac{r^p - 1}{r^p - r^{p-1}} \right)^p \]
and

\[ \gamma_k a_k = \frac{r^{p-1} - 1}{r^p - r^{p-1}}; \]

hence Eq. (4.7) may be rewritten as

\[ \frac{1 + \epsilon_k}{1 - \epsilon_k} = \left( \frac{r^p - 1}{r^p - r^{p-1}} \right)^{p-1} \left( \frac{r^p - r^{p-1}}{r^{p-1} - 1} \right)^{p-1}, \]

or

\[ \frac{1 + \epsilon_k}{1 - \epsilon_k} = \left( \frac{r^p - 1}{r^{p-1} - 1} \right)^{-p-1} \left( \frac{p-1}{p} \right)^{p-1} \frac{1}{p} \frac{r^p - 1}{r^p - r^{p-1}}. \]

Finally this leads to

\[ \frac{1 + \epsilon_k}{1 - \epsilon_k} = \frac{r^n}{r} \left( \frac{r^n - 1}{r^p - r} \right)^{p-1} \left( \frac{p-1}{p} \right)^{p-1} \frac{1}{p} \left( \frac{1 - 1/r^n}{1 - 1/r} \right), \quad (4.8) \]

where it can be noted that

\[ \frac{1 - 1/r^n}{1 - 1/r} < p \quad (4.9) \]

and

\[ \left( \frac{r^p - 1}{r^p - r} \right)^{p-1} < \left( \frac{p}{p-1} \right)^{p-1} \quad (4.10) \]

on using the inequality (3.21).

Substituting the inequalities (4.9) and (4.10) into (4.2) produces

\[ \frac{1 + \epsilon_k}{1 - \epsilon_k} < \frac{1}{r} \left( \frac{1 + \epsilon_{k-1}}{1 - \epsilon_{k-1}} \right), \]

which implies

\[ \epsilon_k < \epsilon_{k-1}. \quad (4.11) \]

Equations (4.11) and (4.6) therefore prove that the algorithm converges and is asymptotically of second order.
5. LIMITING VALUES FOR THE ACCELERATION PARAMETERS

These limits can be obtained quite simply from the results of Sec. 3 as follows. From Eq. (3.13)
\[ \alpha_n = \beta_n \gamma_n, \]
and thus from Eq. (3.18)
\[ \varepsilon_n = 1 - \gamma_n^{p-1} \beta_n^p \frac{p^p}{(p-1)^{p-1}}. \]  

(5.1)

Now from the inequality (3.19), since \( a_n \to 1, b_n \to 1 \), we have that
\[ \lim_{n \to \infty} \gamma_n = p - 1. \]  

(5.2)

Thus, since from Sec. 4
\[ \lim_{n \to \infty} \varepsilon_n = 0, \]

it follows from the Eq. (5.1) that
\[ \lim_{n \to \infty} \left( 1 - (p-1)^{p-1} \beta_n^p \frac{p^p}{(p-1)^{p-1}} \right) = 0, \]
so that
\[ \lim_{n \to \infty} \beta_n = \frac{1}{p}, \]  

(5.3)

and hence from (3.13) and (5.2),
\[ \lim_{n \to \infty} \alpha_n = \frac{p-1}{p}. \]  

(5.4)

6. THE PROPOSED ALGORITHM

The algorithm developed and described in Sec. 3 is specifically given by the following steps, where the given matrix is \( A \) and its \( p \)th root is required
to a precision $\varepsilon$.

(1) 

\[
X_0 \leftarrow A, \quad Y_0 \leftarrow I;
\]

\[
a_0 \leftarrow \left\{ 1 / \rho(A^{-1}) \right\}^{p-1}, \quad b_0 \leftarrow \left\{ \rho(A) \right\}^{p-1};
\]

\[n \leftarrow 0.\]

(2) While $b_n - a_n > \varepsilon$ repeat the following steps:

(2.1) Compute $\gamma_n$ using Eq. (3.14).
(2.2) Compute $\beta_n$ using Eq. (3.15).
(2.3) Compute $\alpha_n$ using Eq. (3.13).
(2.4) Compute $\varepsilon_n$ using Eq. (3.16).
(2.5) Compute $X_{n+1}$ and $Y_{n+1}$ using Eqs. (2.9) and (2.10).
(2.6) $a_{n+1} \leftarrow 1 - \varepsilon_n$; $b_{n+1} \leftarrow 1 + \varepsilon_n$; $n \leftarrow n + 1$.

(3) $A^{1/p}$ is approximated by $X_n$; $A^{(1-p)/p}$ is approximated by $Y_n$.

In place of $\rho(A^{-1})$ and $\rho(A)$ the overestimates $\|A^{-1}\| \|A\|$ respectively may be used.

7. NUMERICAL EXAMPLES

Both Newton's method and the new algorithm were used on the following examples for illustration:

(a) $p = 2$,

\[
A = \begin{bmatrix} 5 & 4 & 1 \\ 4 & 6 & 4 \\ 1 & 4 & 5 \end{bmatrix},
\]

(b) $p = 3$,

\[
A = \begin{bmatrix} 14 & 14 & 6 \\ 14 & 20 & 14 \\ 6 & 14 & 14 \end{bmatrix},
\]

(c) $p = 5$,

\[
A = \begin{bmatrix} 132 & 164 & 100 \\ 164 & 232 & 164 \\ 100 & 164 & 132 \end{bmatrix}
\]

For these examples it is known that the $p$th root of $A$ is given by

\[
C = \begin{bmatrix} 2 & 1 & 0 \\ 1 & 2 & 1 \\ 0 & 1 & 2 \end{bmatrix}.
\]
The Euclidean norm,
\[ \epsilon_k = \| C \cdot X_k \|_E, \]
is used as a measure of convergence after \( k \) iterations. The results for both Newton's method and the new algorithm are listed in Table 2.

**TABLE 2**
Comparison of Convergence of Algorithms for the \( p \)th Root of \( A \)

<table>
<thead>
<tr>
<th>Example</th>
<th>Iterations</th>
<th>( \epsilon_k )</th>
</tr>
</thead>
<tbody>
<tr>
<td>( p = 2 )</td>
<td>1</td>
<td>2.95</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>0.67</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>0.055</td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>0.00043</td>
</tr>
<tr>
<td></td>
<td>5</td>
<td>2.8 \times 10^{-8}</td>
</tr>
<tr>
<td></td>
<td>6</td>
<td>3.8 \times 10^{-14}</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Example</th>
<th>Iterations</th>
<th>( \epsilon_k )</th>
</tr>
</thead>
<tbody>
<tr>
<td>( p = 3 )</td>
<td>1</td>
<td>23.4</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>14.4</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>8.4</td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>4.5</td>
</tr>
<tr>
<td></td>
<td>5</td>
<td>2.1</td>
</tr>
<tr>
<td></td>
<td>6</td>
<td>0.71</td>
</tr>
<tr>
<td></td>
<td>7</td>
<td>0.11</td>
</tr>
<tr>
<td></td>
<td>8</td>
<td>0.003</td>
</tr>
<tr>
<td></td>
<td>9</td>
<td>4 \times 10^{-6}</td>
</tr>
<tr>
<td></td>
<td>10</td>
<td>1.6 \times 10^{-9}</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Example</th>
<th>Iterations</th>
<th>( \epsilon_k )</th>
</tr>
</thead>
<tbody>
<tr>
<td>( p = 5 )</td>
<td>1</td>
<td>711.7</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>568.9</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>454.6</td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>363.2</td>
</tr>
<tr>
<td></td>
<td>5</td>
<td>290.1</td>
</tr>
<tr>
<td></td>
<td>6</td>
<td>231.6</td>
</tr>
<tr>
<td></td>
<td>7</td>
<td>184.8</td>
</tr>
<tr>
<td></td>
<td>8</td>
<td>147.4</td>
</tr>
<tr>
<td></td>
<td>9</td>
<td>117.5</td>
</tr>
<tr>
<td></td>
<td>10</td>
<td>93.5</td>
</tr>
</tbody>
</table>
Fig. 2. Graphical illustration of the convergence of the function $c(x,y)$ for example (a).
Fig. 3. Graphical illustration of the convergence of the function $c(x,y)$ for example (b).
Fig. 4. Graphical illustration of the convergence of the function $c(x,y)$ for example (e).
8. CONCLUSION

The usual assumptions [5] that the inversion of an $n \times n$ matrix requires $n^3$ operations and the multiplication of two $n \times n$ matrices also requires $n^3$ operations are made where an operation is understood to be one scalar multiplication together with one scalar addition [4]. The new algorithm, using Eqs. (2.9) and (2.10), then requires

$$\left[2 + B_{p-2} + \log_2(p-2)\right]n^3, \quad p > 2,$$

$$2n^3, \quad p = 2,$$

operations per iteration, where

$$B_{p-2} = \text{number of ones in binary representation of } p - 2,$$

and the number of operations is determined as follows:

1. $n^3$ operations to find $X_n^{-1}$,
2. $[-1 + B_{p-2} + \log_2(p-2)]n^3$ operations to find $X_n^{2-p}$ [6],
3. $n^3$ operations to find $w = x_{n-2}^{-1}$ by solving

$$W = X_n^{2-p} Y_n^{-1}$$

4. $n^3$ operations to find $X_n^{1-p}$.

It can be observed that $X_n^{2-p}$ is the identity matrix for $p = 2$, in which case only $2n^3$ operations per iteration are required. If $\sigma$ iterations are needed, then the total number of iterations required is given by

$$s = \begin{cases} \sigma \left[2 + B_{p-2} + \log_2(p-2)\right]n^3, & p > 2, \\
2n^3\sigma, & p = 2. \end{cases}$$

Typical values for $\sigma, s$ corresponding to $p = 2, 3, 5$
for seven digits of accuracy are
\[ \sigma = 4, 6, 9 \]
and
\[ s = 8n^3, 18n^3, 45n^3. \]

Newton’s method [Eq. (2.7)] requires
\[ [1 + B_{p-1} + \log_3(p-1)]n^3, \quad p > 1. \]
operations per iteration, which can be determined as follows:

1. \( n^3 \) operations to find \( X_n^{-1} \),
2. \( [-1 + B_{p-1} + \log_3(p-2)]n^3 \) operations to find \( X_n^{1-p} \),
3. \( n^3 \) operations to find \( AX_n^{1-p} \).

If \( \sigma \) iterations are needed, then the total number of iterations required is given by
\[ s = \sigma [1 + B_{p-1} + \log_3(p-1)]n^3. \]

Typical values for \( \sigma, s \) corresponding to
\[ p = 2, 3, 5 \]
for seven digits of accuracy are
\[ \sigma = 5, 10, 35 \]
and
\[ s = 10n^3, 30n^3, 140n^3. \]

The storage required for the new algorithm is \( 6n^2 \) words, if it is assumed that an \( n \times n \) matrix occupies \( n^2 \) words of storage. The storage requirement is determined as follows:

- \( n^2 \) words for \( Y_n \),
- \( n^2 \) words for \( X_n \),
- \( n^2 \) words for \( X_n^{-1} \),
n^2 \text{ words for work storage to find } X_n^{2-p},
\begin{align*}
&n^2 \text{ words for } X_n^{2-p}, \\
&n^2 \text{ words for } X_n^{2-p} Y_n^{-1}, \text{ which can also be used for storing } X_n^{1-p}.
\end{align*}

The storage required for Newton's method is \( 5n^2 \) words, determined as follows:
\begin{align*}
&n^2 \text{ words for } A, \\
&n^2 \text{ words for } X_n, \\
&n^2 \text{ words for } X_n^{-1}, \\
&2n^2 \text{ words to find } X_n^{1-p}.
\end{align*}

Thus since the new algorithm requires fewer steps to achieve the same accuracy, the only apparent extra penalty involved is the extra storage needed for the matrix \( Y_n \). The convergence of the eigenvalues of the matrix \( C \) for examples (a), (b), (c) is illustrated graphically in Figs. 2–4.

REFERENCES


*Received 1 June 1978; revised 19 October 1978*