CALCOLO 40, 273 – 283 (2003)
DOI: 10.1007/s10092-003-0079-9

CALCOLO
© Springer-Verlag 2003

A note on computing the matrix square root

B. Iannazzo

Dipartimento di Matematica "L. Tonelli", Università di Pisa, Pisa, Italy
e-mail: iannazzo@mail.dm.unipi.it

Received: June 2003 / Revised version: October 2003 – © Springer-Verlag 2003

Abstract. We revisit recent algorithms for computing matrix square roots and relate them to Newton iteration. New iterations are derived and a stability analysis is performed. A suitable scaling of an algorithm based on cyclic reduction is introduced, which removes the slow convergence of this iteration encountered in certain cases.

1 Introduction

Consider the matrix equation

$$X^2 - A = 0, \tag{1}$$

where $A \in \mathbb{C}^{n\times n}$. A solution X of (1) is called a matrix square root of A. When A has no non-positive real eigenvalues, Eq. (1) has a unique solution X, denoted by $A^{1/2}$ and called the *principal square root,* such that $\sigma(A^{1/2}) \subset \mathbb{C}^+$ [5]. Here $\sigma(M)$ is the set of eigenvalues of the matrix M and $\mathbb{C}^+$ is the set of complex numbers with positive real parts. In this note we assume that A has no non-positive real eigenvalues and we are interested in the computation of the principal square root.

The classical Newton iteration

$$\begin{cases} X_k H_k + H_k X_k = A - X_k^2, \\ X_{k+1} = X_k + H_k, \end{cases} \qquad k = 0, 1, \ldots \tag{2}$$

for suitable initial values X_0, provides a sequence $\{X_k\}$ which converges to $A^{1/2}$. This method has good properties of convergence and of stability, but it

This work was supported by MIUR, grant number 2002014121.

is too expensive in terms of arithmetic operations. In fact, at each iteration the matrix H_k which solves the Sylvester equation $X_k H_k + H_k X_k = A - X_k^2$ must be computed. The large cost of this computation makes Newton iteration useful only as an iterative refinement.

If we set $X_0 = A$, then X_k commutes with H_k [4] and so (2) can be written in a simplified form as

$$X_{k+1} = \frac{A X_k^{-1} + X_k}{2}. \tag{3}$$

Unfortunately this iteration, which is computationally simpler than (2), becomes unstable in most cases as pointed out by Higham [4].

Similar iterations have been designed differently, in particular, the Denman and Beavers (DB) method [3], based on the matrix sign function iteration, and the Meini iteration [8], based on the cyclic reduction algorithm (CR). The former generates the sequence $\{Y_k\}$, which converges to $A^{1/2}$, defined by

$$\begin{cases} Y_0 = A, \quad Z_0 = I, \\ Y_{k+1} = \dfrac{1}{2}\left(Y_k + Z_k^{-1}\right) \qquad k = 0, 1, \ldots \\ Z_{k+1} = \dfrac{1}{2}\left(Z_k + Y_k^{-1}\right) \end{cases} \tag{4}$$

and is based on the fact that

$$\operatorname{sign}\left(\begin{bmatrix} 0 & A \\ I & 0 \end{bmatrix}\right) = \begin{bmatrix} 0 & A^{1/2} \\ A^{-1/2} & 0 \end{bmatrix}.$$

The latter generates the sequence $\{Z_k\}$, which converges to $\frac{1}{4}A^{1/2}$, defined by

$$\begin{cases} Z_0 = 2(I + A), \quad Y_0 = I - A, \\ Y_{k+1} = -Y_k Z_k^{-1} Y_k, \\ Z_{k+1} = Z_k - 2 Y_k Z_k^{-1} Y_k, \quad k = 0, 1, \ldots \end{cases} \tag{5}$$

and is based on the fact that the matrix

$$H_0 = \frac{1}{4} A^{-1/2}$$

is the block constant coefficient of the inverse of the Laurent matrix polynomial

$$R(z) = (I - A)z^{-1} + 2(I - A) + (I - A)z.$$

The same iteration can be obtained by applying the algorithm developed in [1] to solve Riccati equations; in fact Eq. (1) is a simple case of a continuous-time algebraic Riccati equation.

Both iterations (4) and (5) have good stability properties, but the latter is more stable and, from the numerical experiments reported in [8], in most cases it appears to be the fastest reliable iteration designed so far.

All these iterations are strictly related to Newton's method; in fact it was observed that the matrix Y_k generated by DB iteration coincides with the matrix X_k generated by Newton's method, and the matrix Z_k of Meini's iteration is $4X_{k+1}$. That is, (4) and (5) are two numerically different and more stable ways to implement Newton's iteration.

In this note we provide a direct derivation of Meini's algorithm (5) from Newton's iteration and we prove its numerical stability by means of a theoretical analysis. Then we introduce a scaled technique which substantially improves the convergence properties of the algorithm of [8] in some critical cases. The numerical experiments which we have performed, reported at the end of the paper, and the numerical experiments of [8] show that this algorithm is a reliable tool for computing the principal matrix square root.

2 Iterations derived from Newton's method

In this section we show how to obtain iterations (4) and (5) from Newton's iteration (3).

We rewrite Newton's method (3) in a slightly different but equivalent way:

$$\begin{cases} H_k = \dfrac{AX_k^{-1} - X_k}{2}, \\ X_{k+1} = X_k + H_k, \end{cases} \qquad k = 0, 1, \dots \tag{6}$$

with $X_0 = A$. We call (6) the incremental version of Newton's method. We prove the following preliminary result.

Lemma 1 *The matrices X_k and H_k generated by* (6) *with $X_0 = A$ are such that*

(I) $X_k A = A X_k$,
(II) $X_k^{-1} A = A X_k^{-1}$,
(III) $H_k X_{k+1}^{-1} = X_{k+1}^{-1} H_k$.

Proof It follows immediately that (I) and (II) are equivalent. We prove (I) by induction. If $k = 0$, $X_0 = A$ and so commutes with A. Suppose that $X_k A = A X_k$, then $X_k^{-1} A = A X_k^{-1}$ and

$$2X_{k+1}A = (AX_k^{-1}+X_k)A = AX_k^{-1}A+X_kA = A(AX_k^{-1}+X_k) = 2AX_{k+1}.$$

Now we can show that (I) implies (III). In fact,

$$H_k X_{k+1}^{-1} = (AX_k^{-1} - X_k)(AX_k^{-1} + X_k)^{-1}$$

so it is enough to prove that

$$(AX_k^{-1} - X_k)(AX_k^{-1} + X_k)^{-1} = (AX_k^{-1} + X_k)^{-1}(AX_k^{-1} - X_k).$$

Post- and pre-multiplying both sides of this equation by $(AX_k^{-1} + X_k)$ yields

$$(AX_k^{-1} + X_k)(AX_k^{-1} - X_k) = (AX_k^{-1} - X_k)(AX_k^{-1} + X_k).$$

Performing all the multiplications we obtain

$$AX_k^{-1}AX_k^{-1} + X_kAX_k^{-1} - A - X_k^2 = AX_k^{-1}AX_k^{-1} - X_kAX_k^{-1} + A - X_k^2,$$

that is,

$$2X_kAX_k^{-1} = 2A$$

and this holds because of (I). □

Now we are ready to derive an explicit iteration of the term H_k of the incremental version of Newton's method in (6). From Lemma 1 we set

$$\begin{aligned} H_{k+1} &= \frac{1}{2}(AX_{k+1}^{-1} - X_{k+1}) = \frac{1}{2}(A - X_{k+1}^2)X_{k+1}^{-1} \\ &= \frac{1}{2}\left(A - \frac{(AX_k^{-1} + X_k)^2}{4}\right)X_{k+1}^{-1} \\ &= \frac{1}{2}\left(\frac{-A^2X_k^{-2} + 2A - X_k^2}{4}\right)X_{k+1}^{-1} \\ &= -\frac{1}{2}\frac{(AX_k^{-1} - X_k)^2}{4}X_{k+1}^{-1} = -\frac{1}{2}H_k^2X_{k+1}^{-1}. \end{aligned}$$

This gives the following equivalent iterative formulas for the increment H_k:

$$H_{k+1} = \begin{cases} -\frac{1}{2}X_{k+1}^{-1}H_k^2, \\ -\frac{1}{2}H_kX_{k+1}^{-1}H_k, \\ -\frac{1}{2}H_k^2X_{k+1}^{-1}. \end{cases} \tag{7}$$

From these three expressions we obtain three different algorithms for computing the sequence of Newton's iteration. According to our numerical experiments, the best in terms of stability is:

$$\begin{cases} X_0 = A, \quad H_0 = \frac{1}{2}(I - A), \\ X_{k+1} = X_k + H_k, \quad H_{k+1} = -\frac{1}{2}H_kX_{k+1}^{-1}H_k. \end{cases} \tag{8}$$

By setting $Z_k = 4X_{k+1}$ and $Y_k = 2H_k$, we see that this iteration is that of (5).

Similarly, we can derive DB iteration. In fact, with the substitutions $Y_k = X_k$ and $Z_k = A^{-1}X_k$, from (6) we obtain

$$\begin{cases} Y_{k+1} = \dfrac{Y_k + Z_k^{-1}}{2} \\ Z_{k+1} = A^{-1}X_{k+1} = \dfrac{A^{-1}(AX_k^{-1} + X_k)}{2} = \dfrac{Z_k + X_k^{-1}}{2}. \end{cases}$$

3 Stability analysis

In this section we investigate the stability of algorithm (8). More precisely, we analyze how a small perturbation at the generic iteration step is amplified or damped along the iteration. This analysis is important in the numerical applications of Newton's iteration where roundoff errors due to floating point arithmetic may perturb the computed matrix X_k.

Let ΔX_k and ΔH_k be the perturbations introduced at the kth step of the iteration and let

$$\begin{aligned} \widetilde{X}_k &= X_k + \Delta X_k, \\ \widetilde{H}_k &= H_k + \Delta H_k. \end{aligned}$$

Here and hereafter we perform a first-order error analysis, i.e., we formally neglect quadratic terms such as $(\Delta X_k)^2$, $(\Delta H_k)^2$, $\Delta H_k \Delta X_k$ and $\Delta X_k \Delta H_k$. This formal manipulation is meaningful if ΔX_k and ΔH_k are sufficiently small. Moreover, we use the symbol $\doteq$ to denote equivalence up to within quadratic terms in the perturbations ΔX_k and ΔH_k. We do the same with the symbol $\dot{\leq}$.

We next compute $\widetilde{X}_{k+1}$ and $\widetilde{H}_{k+1}$ starting from the values $\widetilde{X}_k$ and $\widetilde{H}_k$, where we perform our computations in exact arithmetic. Then we have

$$\begin{cases} \widetilde{X}_{k+1} = \widetilde{X}_k + \widetilde{H}_k, \\ \widetilde{H}_{k+1} = -\dfrac{\widetilde{H}_k \widetilde{X}_{k+1}^{-1} \widetilde{H}_k}{2}. \end{cases}$$

For $\widetilde{X}_{k+1}$, we have

$$\Delta X_{k+1} = \widetilde{X}_{k+1} - X_{k+1} = \Delta X_k + \Delta H_k.$$

For $\widetilde{H}_{k+1}$ we find that

$$\begin{aligned} \Delta H_{k+1} = \widetilde{H}_{k+1} - H_{k+1} &= -\frac{\widetilde{H}_k \widetilde{X}_{k+1}^{-1} \widetilde{H}_k}{2} + \frac{H_k X_{k+1}^{-1} H_k}{2} \\ &= -\frac{1}{2}(H_k + \Delta H_k)(X_{k+1} + \Delta X_{k+1})^{-1}(H_k + \Delta H_k) + \frac{1}{2}(H_k X_{k+1}^{-1} H_k). \end{aligned}$$

On the other hand, it can be shown that

$$(A+K)^{-1} = A^{-1} + A^{-1}KA^{-1} + O(K^2).$$

From this relation it follows that

$$\Delta H_{k+1} \doteq -\frac{1}{2}(H_k X_{k+1}^{-1} \Delta X_{k+1} X_{k+1}^{-1} H_k + H_k X_{k+1}^{-1} \Delta H_k + \Delta H_k X_{k+1}^{-1} H_k).$$

Now, since X_{k+1}^{-1} and H_k commute, we have

$$\|\Delta H_{k+1}\| \dot{\leq} \|H_k X_{k+1}^{-1}\|^2 \|\Delta X_k\| + (\|H_k X_{k+1}^{-1}\|^2 + 2\|H_k X_{k+1}^{-1}\|)\|\Delta H_k\|,$$

for any induced matrix norm $\|\cdot\|$.

If we set $\alpha_k = \|H_k X_{k+1}^{-1}\|$, we obtain

$$\|\Delta H_{k+1}\| \dot{\leq} \alpha_k^2 \|\Delta X_k\| + (\alpha_k^2 + 2\alpha_k)\|\Delta H_k\|,$$

but from [8] it follows that

$$\alpha_k = O\big(\tau^{2^k}\big)$$

with $\rho(W) < \tau < 1$, where $W = \big(A^{1/2} - I\big)\big(A^{1/2} + I\big)^{-1}$ and $\rho(W)$ denotes the spectral radius of W. Then the error $\|\Delta H_{k+1}\|$ is under control for large values of k or for small τ and it tends to zero. It follows that, for large n, $\|\Delta X_{n+1}\| \simeq \|\Delta X_n\|$ so that the error at step k does not amplify.

4 Scaling

In this section we answer affirmatively the question set in [8] as to whether it is possible to derive a scaling technique for the iteration (5) which holds for other iterative methods. The scaled iteration, which we propose here, is derived from (8). Such a scaling enables to improve the convergence of the iterative method in a wide class of problems.

We start from the scaling parameter introduced by Byers [2] for the matrix sign iteration and then adapted to Denman and Beavers iteration. Even though this scaling is not optimal as shown in [7], it is easy to compute. We recall that the scaled DB iteration [6] is:

$$\begin{cases} \gamma_k = |\det(Y_k)\det(Z_k)|^{-1/2n}, \\ Y_{k+1} = \frac{1}{2}\big(\gamma_k Y_k + \gamma_k^{-1} Z_k^{-1}\big), \\ Z_{k+1} = \frac{1}{2}\big(\gamma_k Z_k + \gamma_k^{-1} Y_k^{-1}\big), \end{cases} \quad k = 0, 1, \ldots \tag{9}$$

where $Y_k = X_k$ and $Z_k = A^{-1}X_k$. Rewriting this iteration in terms of the sequence $\{X_k\}$, we obtain the scaled Newton method:

$$\begin{cases} \gamma_k = \left|\dfrac{\det(X_k)^2}{\det(A)}\right|^{-1/2n}, \\ X_{k+1} = \dfrac{\gamma_k X_k + \gamma_k^{-1} A X_k^{-1}}{2}. \end{cases} \tag{10}$$

The factor γ_k is easily computed because $\det(X_k)$ is obtained for free during the inversion of X_k. Scaling Newton's method does not solve its stability problems, but leads to a scaling for Meini's iteration. In fact, scaling Newton's method is equivalent to replacing the value of X_k at each step with a new suitable value $\widehat{X}_k = \gamma_k X_k$ such that convergence becomes faster. Thus, if in the iteration (8) we set $\widehat{X}_k = \gamma_k X_k$, then we obtain a new value for the increment that we call $\widehat{H}_k$, namely,

$$\begin{aligned} \widehat{H}_k &= \frac{A\widehat{X}_k^{-1} - \widehat{X}_k}{2} = \frac{\gamma_k^{-1} A X_k^{-1} - \gamma_k X_k}{2} \\ &= \gamma_k^{-1}\left(\frac{AX_k^{-1} - X_k}{2} + \frac{X_k}{2}\right) - \gamma_k \frac{X_k}{2} = \gamma_k^{-1}\left(H_k + \frac{X_k}{2}\right) - \gamma_k \frac{X_k}{2}. \end{aligned} \tag{11}$$

This immediately provides the scaled Meini iteration:

$$\begin{cases} X_0 = A, \quad H_0 = \dfrac{1}{2}(I - A), \\ \gamma_k = \left|\dfrac{\det(X_k)^2}{\det A}\right|^{-1/2n}, \\ \widehat{X}_k = \gamma_k X_k, \\ \widehat{H}_k = \gamma_k^{-1}\left(H_k + \dfrac{X_k}{2}\right) - \gamma_k \dfrac{X_k}{2}, \\ X_{k+1} = \widehat{X}_k + \widehat{H}_k, \\ H_{k+1} = -\dfrac{1}{2}\widehat{H}_k X_{k+1}^{-1} \widehat{H}_k. \end{cases} \tag{12}$$

We see in the next section how this method takes advantage of scaling by speeding up convergence in some critical cases.

It is worth pointing out that in (12) we may replace the expression for γ_k with any other formula designed for scaling the matrix sign iteration [7].

5 Numerical experiments

We illustrate, by means of tests, the effects of scaling on the iteration (8). If the matrix A has no non-positive real eigenvalues, iteration (8) has quadratic convergence and, from [8],

$$\|X_k - A^{1/2}\| = O\left(\tau^{2 \cdot 2^k}\right) \tag{13}$$

for any matrix norm $\|\cdot\|$ and $\rho(W) < \tau < 1$, where

$$W = \left(A^{1/2} - I\right)\left(A^{1/2} + I\right)^{-1}.$$

This implies a slow convergence if an eigenvalue of W lies in the thin annulus $\mathcal{A}_\epsilon = \{\lambda \in \mathbb{C} : \quad 1 - \epsilon \leq |\lambda| < 1\}$ for "small" $\epsilon > 0$. This happens if an eigenvalue of $A^{1/2}$ lies in the pre-image of $\mathcal{A}_\epsilon$ under the Cayley transform

$$z \to w = \frac{z-1}{z+1},$$

that is, if $A^{1/2}$ has eigenvalue close to the imaginary axis and/or large in modulus.

To show the effect of scaling we present an example in which the eigenvalues of $A^{1/2}$ are close to the imaginary axis.

Table 1. Comparison of methods

	M		Ms		Schur
t	Iter	Err	Iter	Err	Err
1	6	1.9×10^{-16}	2	1.1×10^{-16}	2.1×10^{-16}
10	15	5.1×10^{-14}	2	1.1×10^{-18}	7.9×10^{-16}
10^2	25	2.4×10^{-10}	2	1.4×10^{-16}	3.3×10^{-13}
10^3	35	4.1×10^{-10}	2	7.9×10^{-17}	1.8×10^{-11}
10^4	45	6.3×10^{-9}	3	1.5×10^{-16}	4.8×10^{-9}
10^5	55	1.1×10^{-7}	3	1.1×10^{-16}	3.9×10^{-7}
10^6	65	1.7×10^{-5}	2	1.1×10^{-16}	7.3×10^{-6}
10^7	75	1.9×10^{-3}	2	1.1×10^{-16}	1.3×10^{-3}

Test 1 We denote by $\mathbf{i}$ the imaginary unit, $\mathbf{i}^2 = -1$, and consider the complex matrix

$$K = \begin{bmatrix} 1/t + t\mathbf{i} & 0 \\ 0 & 1/t - t\mathbf{i} \end{bmatrix} \tag{14}$$

with real parameter $t > 0$; moreover, define

$$Y = MKM^{-1}, \quad A = Y^2,$$

where

$$M = \begin{bmatrix} 1 & 2 \\ 3 & -1 \end{bmatrix}. \tag{15}$$

In this way, the principal square root of the matrix A is Y and its eigenvalues are $\lambda_1 = 1/t + t\mathbf{i}$ and $\lambda_2 = 1/t - t\mathbf{i}$; for large values of t these are large in modulus and close to the imaginary axis.

In Table 1 we compare the number of steps needed for the convergence of Meini's iteration (M) and the convergence of the scaled iteration (Ms) and we report the final precision expressed through the relative error

$$\text{err} = \frac{\|X - Y\|_F}{\|Y\|_F},$$

where $\|\cdot\|_F$ is the Frobenius norm. We also compare relative errors with those provided by the `sqrtm` function of MATLAB (Schur).

From this example the benefit of the scaling is evident. In fact, the number of steps of the unscaled iteration depends on the location of the eigenvalues of the square root, and grows rapidly when there are eigenvalues close to the imaginary axis. The scaled iteration seems to overcome this problem and does not suffer from bad positioning of the eigenvalues.

Test 2 We next consider the 5×5 matrix R of pseudo-random numbers

$$R = \begin{bmatrix} 0.3759 & 0.9200 & 0.1939 & 0.5488 & 0.6273 \\ 0.1099 & 0.8447 & 0.9048 & 0.9316 & 0.6991 \\ 0.4199 & 0.3678 & 0.5692 & 0.3352 & 0.3972 \\ 0.7537 & 0.6208 & 0.6318 & 0.6555 & 0.4136 \\ 0.7939 & 0.7313 & 0.2344 & 0.3919 & 0.6552 \end{bmatrix}$$

and let $A = \alpha R$, $\alpha \in \mathbb{R}$, be the matrix whose square root we compute. This class of matrices was considered in [8]. We can see that the number of iterations in the scaled version of the algorithm remains constant, whereas in the unscaled version this number grows when α is large or small as one can see in Fig. 1.

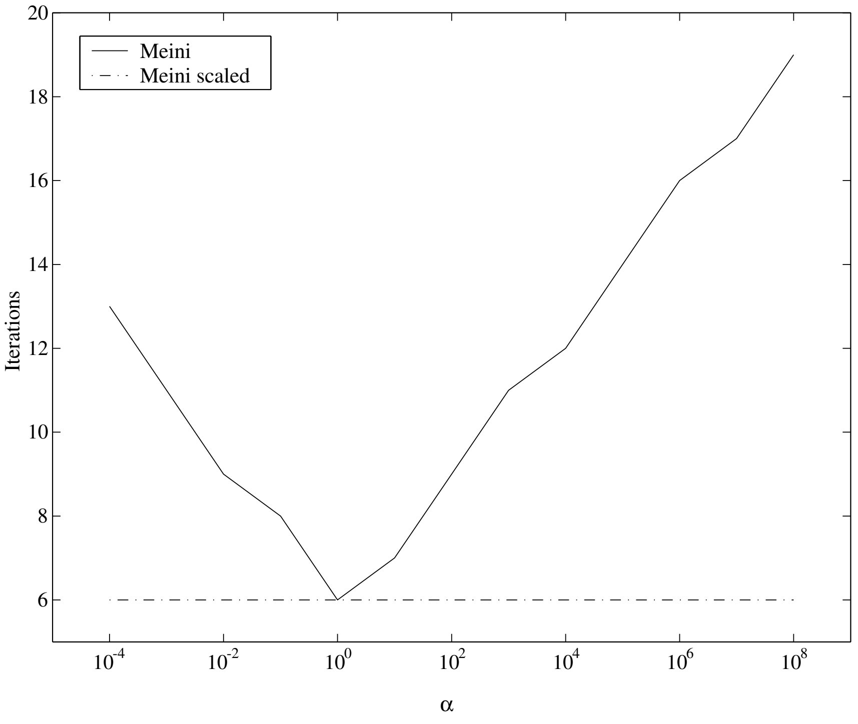

Fig. 1. Comparison of number of steps with (*dotted line*) and without (*solid line*) scaling

6 Conclusion

We have presented a direct derivation of Meini's method from Newton's iteration for the computation of the matrix square root. We performed a stability analysis, which confirms the good numerical properties of the method, and we provided a scaled version which overcomes the slow convergence encountered in certain cases.

From the numerical experiments performed in [8] and in this paper, the algorithm (8) with the scaling (12) seems to be the most reliable method for computing the matrix square root.

References

[1] Bini, D.A., Iannazzo, B.: A cyclic reduction method for solving algebraic Riccati equations. Submitted for publication, 2003

[2] Byers, R.: Solving the algebraic Riccati equation with the matrix sign function. Linear Algebra Appl. **85**, 267–279 (1987)

[3] Denman, E., Beavers, N.: The matrix sign function and computations in systems. Appl. Math. Comput. **2**, 63–94 (1976)

[4] Higham, N.: Newton's method for the matrix square root. Math. Comp. **46**, 537–549 (1986)

[5] Higham, N.: Computing real square roots of a real matrix, Linear Algebra Appl. **88/89**, 405–430 (1987)

[6] Higham, N.: Stable iterations for the matrix square root. Numer. Algorithms **15**, 227–242 (1997)

[7] Kenney, C., Laub, A.: On scaling Newton's method for polar decomposition and the matrix sign function. SIAM J. Matrix Anal. Appl. **13**, 688–706 (1992)

[8] Meini, B.: The matrix square root from a new functional perspective: theoretical results and computational issues. Technical Report 1455. Pisa: Dipartimento di Matematica, Università di Pisa 2003