Computation of the Matrix Sign Function Using Continued Fraction Expansion

Çetin K. Koç, Bertan Bakkaloğlu, and Leang S. Shieh

Abstract—**We describe an algorithm which computes the sign function of a complex matrix by using the continued fraction expansion of the inverse of the principal square root function at each step of the iteration. We show that the algorithm iteratively computes globally convergent main diagonal Padé approximants. The proposed algorithm avoids computing large matrix powers and performs fewer matrix inversions than Newton's method. The algorithm is multiplication-rich and particularly suitable for implementation on vector and parallel computers. The stability analysis of the algorithm suggests that the errors introduced during a step are either suppressed or have limited effect on the next step. Finally, we summarize the results of our experiments on computing the sign function of certain matrices.**

I. Introduction

The matrix sign function has several applications in system theory and matrix analysis, including the solution of algebraic Riccati and matrix Lyapunov equations, system decomposition, model reduction, and separation of eigenpairs. The best-known methods for computing the matrix sign function include Newton's method (with or without scaling) and other rational iterative methods. Newton's method has constant order of convergence, and the speed of convergence would be slower for matrices with eigenvalues near the imaginary axis. Also, rational approximations involve taking higher powers of a given matrix for each step of the iteration which might introduce numerical errors. Here we describe an algorithm which is based on continued fraction expansion of the inverse principal square root of a matrix. The algorithm performs fewer matrix inversions than Newton's method and iteratively computes globally convergent main diagonal Padé approximants.

The matrix sign function maps the stable and unstable eigenvalues of a given matrix to -1 and 1, respectively, while preserving the eigenvectors of the original matrix. This property of the matrix sign function is useful for studying the eigenstructures of matrices without explicitly computing the eigenvalues. The sign of a complex scalar λ is defined over $\mathrm{Re}(\lambda) \neq 0$ by

$$\mathrm{sign}(\lambda) = \begin{cases} 1 & \text{if } \mathrm{Re}(\lambda) > 0, \\ -1 & \text{if } \mathrm{Re}(\lambda) < 0. \end{cases}$$

This definition can be extended to a matrix $A \in \mathcal{C}^{n\times n}$ whose eigenvalues do not lie on the imaginary axis. Let M take A to its Jordan form J as

$$A = MJM^{-1}. \tag{1}$$

Let J be defined as

$$J = \begin{bmatrix} J_+ & 0 \\ 0 & J_- \end{bmatrix} = J_+ \oplus J_-$$

where $J_+ \in \mathcal{C}^{n_1\times n_1}$ and $J_- \in \mathcal{C}^{n_2\times n_2}$ are the Jordan blocks with $\mathrm{Re}(\sigma(A)) > 0$ and $\mathrm{Re}(\sigma(A)) < 0$, respectively, and $n = n_1 + n_2$.

Manuscript received April 12, 1993; revised July 30, 1993. This work was supported by U.S. Army Research Office Grant DAAL03-91-6-0106.

C. K. Koç and B. Bakkaloğlu are with the Department of Electrical and Computer Engineering, Oregon State University, Corvallis, OR 97331 USA.

L. S. Shieh is with the Department of Electrical Engineering, University of Houston, Houston, TX 77204 USA.

IEEE Log Number 9401686.

Applying sign function to both sides of (1) we obtain

$$\mathrm{sign}(A) = M\,\mathrm{sign}(J)M^{-1}.$$

The matrix sign of the Jordan blocks determine the sign of A as

$$\mathrm{sign}(A) = M\begin{bmatrix} \mathrm{sign}(J_+) & 0 \\ 0 & \mathrm{sign}(J_-) \end{bmatrix}M^{-1} = M\begin{bmatrix} I & 0 \\ 0 & -I \end{bmatrix}M^{-1}. \tag{2}$$

Equation (2) shows that the Jordan blocks corresponding to positive (negative) eigenvalues are mapped to positive (negative) identity matrices, whose dimensions are the same as the number of positive (negative) eigenvalues. It follows that $S = \mathrm{sign}(A)$ is a diagonalizable matrix which commutes with A and is a square root of the identity, i.e.,

$$S^2 = I \quad \text{and} \quad AS = SA. \tag{3}$$

Equation (3) is a quadratic equation in S and can be solved by Newton's method [7]. The sign of a matrix A is defined as the limit of the Newton iteration

$$\begin{aligned} S_0 &= A, \\ S_{k+1} &= \frac{1}{2\gamma_k}(S_k + \gamma_k^2 S_k^{-1}), \\ \lim_{k\to\infty} S_k &= \mathrm{sign}(A) \end{aligned}$$

where $\gamma_k > 0$ is a scaling factor to accelerate the convergence of the eigenvalues to ± 1, which has been analyzed extensively in [5]. If A has no eigenvalues on the imaginary axis, then the limit exists, and Newton's method converges globally. But for ill-conditioned matrices with eigenvalues near zero, a considerable amount of error is propagated and the convergence slows down.

II. Derivation of the Algorithm

Let $\sqrt{\lambda}$ denote the principal square root of the scalar $\lambda \in \mathcal{C}$ with $\mathrm{Re}(\lambda) \neq 0$, i.e., $(\sqrt{\lambda})^2 = \lambda$ and $\arg(\lambda) \in (-\pi/2, \pi/2)$. Then the scalar sign function of λ can be expressed as

$$\mathrm{sign}(\lambda) = \frac{\lambda}{\sqrt{\lambda^2}}$$

where $\sqrt{\lambda^2}$ is the principal square root of λ^2. This is easily seen by taking $\lambda = \rho e^{j(\theta+\pi k)}$ with $\rho > 0$ and $|\theta| < \pi/2$, where $k = 0$ and $k = 1$ for $\mathrm{Re}(\lambda) > 0$ and $\mathrm{Re}(\lambda) < 0$, respectively. By squaring, we obtain

$$\lambda^2 = \rho^2 e^{j2(\theta+\pi k)} = \rho^2 e^{j2\theta}.$$

Since the principal square root of λ^2 is $\sqrt{\lambda^2} = \rho e^{j\theta}$, we obtain the scalar sign function as

$$\frac{\lambda}{\sqrt{\lambda^2}} = \frac{\rho e^{j(\theta+\pi k)}}{\rho e^{j\theta}} = e^{j\pi k} = \mathrm{sign}(\lambda).$$

We can extend the scalar sign function definition to a general square matrix $A \in \mathcal{C}^{n\times n}$ as

$$\mathrm{sign}(A) = A(\sqrt{A^2})^{-1} = A^{-1}(\sqrt{A^2}) \tag{4}$$

where $\sqrt{A^2}$ denotes the principal matrix square root of A^2. The principal square root of a matrix $\sqrt{A}$ is defined as the matrix that satisfies $(\sqrt{A})^2 = A$ and $\arg(\sigma(\sqrt{A})) \in (-\pi/2, \pi/2)$.

Our departure point in obtaining a fast converging iterative algorithm will be the matrix continued fraction algorithm for computing the principal square root of a complex matrix. The algorithm is

derived from Khovanskii's algorithm for computing square roots of scalars (see [6, Chapter IV, (1.5)]). The iterative algorithm for computing the inverse of the principal square root of the complex matrix $A \in C^{n\times n}$ can be stated as

$$\begin{bmatrix} P_1 \\ Q_1 \end{bmatrix} = \begin{bmatrix} I \\ I \end{bmatrix},$$

$$\begin{bmatrix} P_j \\ Q_j \end{bmatrix} = \begin{bmatrix} I & I \\ A & I \end{bmatrix} \begin{bmatrix} P_{j-1} \\ Q_{j-1} \end{bmatrix},$$

$$\lim_{j\to\infty} P_j Q_j^{-1} = (\sqrt{A})^{-1} \tag{5}$$

where P_j, $Q_j \in C^{n\times n}$. To see what the above algorithm computes, we write

$$P_j Q_j^{-1} = (P_{j-1} + Q_{j-1})(AP_{j-1} + Q_{j-1})^{-1}$$

from (5). Post-multiplying the expressions in the parentheses with Q_j^{-1}, we have

$$P_j Q_j^{-1} = (I + P_{j-1}Q_{j-1}^{-1})(I + AP_{j-1}Q_{j-1}^{-1})^{-1}. \tag{6}$$

Let $\lim_{j\to\infty} P_j Q_j^{-1} = X$, then from (6), we obtain

$$X = (I + X)(I + AX)^{-1}$$

which implies $AX^2 = I$. Therefore if $\lim_{j\to\infty} P_j Q_j^{-1}$ exists, then it is equal to $(\sqrt{A})^{-1}$. An extension analysis of the necessary and sufficient conditions for this limit to exist is given in [6]. Briefly, the limit for the scalar iteration exists provided that A is nonzero. For the matrix case, this translates to saying that the matrix A should not have zero eigenvalues.

III. Description of the Algorithm

By replacing the block element A with A^2 in the iteration matrix of (5), we obtain an iterative algorithm for computing the inverse square root of the square of a complex matrix, which in turn can be used for the computation of the matrix sign function. Following the matrix sign function definition (4), we expect an iteration of the form

$$S[0] = A,$$

$$S[k+1] = S[k](\sqrt{S^2[k]})^{-1},$$

$$\lim_{k\to\infty} S[k] = \operatorname{sign}(A).$$

For each value of k, we compute an approximation for the inverse of the principal square root of $S^2[k]$. This is achieved by iterating the continued fraction algorithm r times for $j = 1, 2, \cdots, r$. The matrix sign function algorithm starts with $S = A$ and iteratively computes the sign function of A by going through a series of baby steps and giant steps, corresponding to the computation of an r-step approximation for the inverse of the principal square root and the computation of the new value of S, respectively. Thus, the iteration for the inverse square root is modified for computing the matrix sign function as

$$\begin{aligned} \text{Start: } & S[0] = A, \\ \text{Baby Step: } & P_1[k] = I, \\ & Q_1[i] = I, \\ j = 2, 3, \cdots, r\text{: } & P_j[k] = P_{j-1}[k] + Q_{j-1}[k], \\ & Q_j[k] = S^2[k]P_{j-1}[k] + Q_{j-1}[k], \\ \text{Giant Step: } & S[k+1] = S[k]P_r[k]Q_r^{-1}[k]. \end{aligned} \tag{7}$$

A baby step corresponds to the computation of $P_r[k]$ and $Q_r[k]$ using $S[k]$ and the initial values $P_1[k] = Q_1[k] = I$. There are no matrix inversions during this phase. The number of iterations during the baby-step phase is equal to r, which is predetermined. We start with $P_1[k] = I$ and $Q_1[k] = I$ and compute $P_j[k]$ and $Q_j[k]$ for $j = 2, 3, \cdots, r$, using the continued fraction-based iterative algorithm for the inverse square root. A giant step, on the other hand, corresponds to the computation of $S[k+1]$, using $S[k]$, $P_r[k]$, and $Q_r[k]$. There is a single matrix inversion during the giant step. The number of giant step iterations is a function of the convergence properties of the algorithm, the input matrix, and the baby-step length r.

Theorem 1: The above algorithm iteratively computes the globally convergent main diagonal $[k, m]$ Padé approximants for $k = \lfloor (r-1)/2 \rfloor$ and $m = \lfloor r/2 \rfloor$.

Proof: As noted in [4], the numerator and the denominator of the main diagonal Padé approximants $sP_r(s^2)$ and $Q_r(s^2)$ have the following interesting property: The polynomials $-sP_r(s^2)$ and $Q_r(s^2)$ are the odd and even parts of $(1-s)^r$, respectively. Let $sP_r(s^2)$ and $Q_r(s^2)$ be the numerator and the denominator of the rational approximants computed by the iteration of (7) for the computation of the sign function of s. Also let P_r and Q_r denote $P_r(s^2)$ and $Q_r(s^2)$. Consider the following rational polynomial

$$\begin{aligned} \frac{P_r}{Q_r} - \frac{1}{s} &= \frac{P_{r-1} + Q_{r-1}}{Q_{r-1} + s^2 P_{r-1}} - \frac{1}{s} \\ &= \frac{s(P_{r-1} + Q_{r-1}) - (Q_{r-1} + s^2 P_{r-1})}{s(Q_{r-1} + s^2 P_{r-1})} \\ &= \frac{(s - s^2)P_{r-1} - (1-s)Q_{r-1}}{sQ_r} \\ &= (1-s)\left(\frac{P_{r-1}}{Q_r} - \frac{Q_{r-1}}{sQ_r}\right) \\ &= (1-s)\frac{Q_{r-1}}{Q_r}\left(\frac{P_{r-1}}{Q_{r-1}} - \frac{1}{s}\right). \end{aligned}$$

Similarly, we have

$$\frac{P_{r-1}}{Q_{r-1}} - \frac{1}{s} = (1-s)\frac{Q_{r-2}}{Q_{r-1}}\left(\frac{P_{r-2}}{Q_{r-2}} - \frac{1}{s}\right)$$

which implies

$$\begin{aligned} \frac{P_r}{Q_r} - \frac{1}{s} &= (1-s)\frac{Q_{r-1}}{Q_r}(1-s)\frac{Q_{r-2}}{Q_{r-1}}\left(\frac{P_{r-2}}{Q_{r-2}} - \frac{1}{s}\right) \\ &= (1-s)^2\frac{Q_{r-2}}{Q_r}\left(\frac{P_{r-2}}{Q_{r-2}} - \frac{1}{s}\right). \end{aligned}$$

Iterating the above $r-1$ times, we obtain

$$\frac{P_r}{Q_r} - \frac{1}{s} = (1-s)^{r-1}\frac{Q_1}{Q_r}\left(\frac{P_1}{Q_1} - \frac{1}{s}\right).$$

Since $P_1 = Q_1 = 1$, we obtain the equality

$$(1-s)^r = Q_r - sP_r$$

i.e., the polynomials $-sP_r(s^2)$ and $Q_r(s^2)$, computed by the continued fraction algorithm, are odd and even parts of $(1-s)^r$. Thus, the matrix iteration (7) indeed computes the numerator and the denominator of the main diagonal Padé approximants. Also note that $sP_r(s^2)/Q_r(s^2)$ is a main diagonal Padé approximant of order $[m-1, m]$ or $[m, m]$, hence we have $r = 2m$ or $2m+1$; in other words, $k = \lfloor (r-1)/2 \rfloor$ and $m = \lfloor r/2 \rfloor$. In Table I, we list the main diagonal Padé approximants for the matrix sign function for $[k, m]$ up to $[3, 3]$. □

TABLE I
MAIN DIAGONAL PADÉ RECURSIONS FOR THE MATRIX SIGN FUNCTION

$[k, m]$	[0, 1]	[1, 1]	[1, 2]	[2, 2]	[2, 3]	[3, 3]
$\frac{sP_k(s^2)}{Q_m(s^2)}$	$\frac{2s}{1+s^2}$	$\frac{s(3+s^2)}{1+3s^2}$	$\frac{4s(1+s^2)}{1+6s^2+s^4}$	$\frac{s(5+10s^2+s^4)}{1+10s^2+5s^4}$	$\frac{2s(3+10s^2+3s^4)}{1+15s^2+15s^4+s^6}$	$\frac{s(7+35s^2+21s^4+s^6)}{1+21s^2+35s^4+7s^6}$

IV. STABILITY ANALYSIS

We analyze the stability of the algorithm, following a method similar to the one in [3]. We consider the iteration of (7) subject to perturbations arising from rounding errors at a given step k. Let $\tilde{S}[k] = S[k] + E[k]$ where $E(k)$ is the error at step k. Since the error terms get complicated for higher values of r, here we analyze the algorithm for $r = 2$. We have

$$\tilde{P}_2[k] = I + I = 2I,$$

$$\tilde{Q}_2[k] = \tilde{S}^2[k] + I = (S[k] + E[k])^2 + I$$

Thus the error expression for $S[k+1]$ can be written as

$$\begin{aligned}\tilde{S}[k+1] &= (S[k] + WE[k])(2I)(I + (S[k] + E[k])^2)^{-1} \\ &\cong 2(S[k] + E[k])(I + S^2[k] + S[k]E[k] \\ &\quad + E[k]S[k])^{-1}\end{aligned}$$

by ignoring the second-order error term. Assuming

$$\|S^2[k] + I\| > \|E[k]S[k] + S[k]E[k]\|$$

we utilize the perturbation formula [10], [9]

$$(A + \epsilon)^{-1} = A^{-1} - A^{-1}\epsilon A^{-1} + O(\|\epsilon\|^2) \tag{8}$$

and obtain the error at step $(k+1)$ as

$$\begin{aligned}\tilde{S}[k+1] &= 2(S[k] + E[k])((I + S^2[k])^{-1} \\ &\quad - (I + S^2[k])^{-1}(S[k]E[k] + E[k]S[k])(I + S^2[k])^{-1}) \\ &= S[k+1] - \frac{1}{2}S[k+1]E[k]S[k+1] \\ &\quad - (S[k+1]S[k] - 2I)E[k](I + S^2[k])^{-1}.\end{aligned}$$

This gives an expression for $E[k+1]$ as

$$\begin{aligned}E[k+1] &= \tilde{S}[k+1] - S[k+1] \\ &= (2I - S[k+1]S[k])E[k](I + S^2[k])^{-1} \\ &\quad - \frac{1}{2}S[k+1]E[k]S[k+1].\end{aligned}$$

Making use of (8) and the matrix sign function property

$$\text{im}_{k\to\infty} S^2[k] = \lim_{k\to\infty} S[k]S[k+1] = I$$

we obtain the error term

$$E[k+1] \cong \frac{1}{2}(E[k] - S[k+1]E[k]S[k+1]). \tag{9}$$

We define M as the modal matrix of $S[k+1]$, i.e.,

$$D[k+1] = M^{-1}S[k+1]M = \text{diag}\,(\lambda_1^{(k+1)}, \cdots, \lambda_n^{(k+1)}).$$

Let $\hat{E}[k]$ and $\hat{E}[k+1]$ be the matrices such that $\hat{E}[k] = M^{-1}E[k]M$ and $\hat{E}[k+1] = M^{-1}E[k+1]M$. Then the error expression (9) can be written as

$$\begin{aligned}\hat{E}[k+1] &= M^{-1}E[k+1]M \\ &= \frac{1}{2}(\hat{E}[k] - D[k+1]\hat{E}[k+1]D[k+1])\end{aligned}$$

TABLE II
THE NUMBER OF STEPS FOR EXAMPLES 1, 2, AND 3

Ex.	Order of A	Newton's Method: Determinantal Scaling	Spectral Scaling	2-norm Scaling	Frobenious Scaling	No Scaling	New Algorithm No Scaling ($r = 5$)
1	10	8	8	8	8	13	6
	20	10	9	9	9	14	7
	40	10	10	10	10	16	7
2	10	8	8	8	8	15	6
	20	11	15	15	12	12	7
	40	12	13	13	13	15	8
3	4	6	7	7	7	12	7

which can be written elementwise as

$$\hat{e}_{ij}^{(k+1)} = \frac{1}{2}(\hat{e}_{ij}^{(k)} - \lambda_i^{(k+1)}\lambda_j^{(k+1)}\hat{e}_{ij}^{(k)}). \tag{10}$$

The sign function iteration has the property that

$$\lim_{k\to\infty} \lambda_i^{(k+1)}\lambda_j^{(k+1)} = \pm 1.$$

Thus, (10) suggests that errors arising at the kth step are either suppressed or have limited effect on the following iteration. We note that this analysis can be applied for $r > 2$ by using the formulas

$$\tilde{S}[k]\tilde{P}_r[k] = \frac{1}{2}((I + \tilde{S}[k])^r - (I - \tilde{S}[k])^r),$$

$$\tilde{Q}_r[k] = \frac{1}{2}((I + \tilde{S}[k])^r + (I - \tilde{S}[k])^r)$$

which are easily derived from Theorem 1.

V. NUMERICAL EXAMPLES

Example 1: This example is from [5]. Let $A = D + T$, where $D = \text{diag}\,(\lambda_1, \cdots, \lambda_m)$ with $\lambda_i = \pm x \pm yj$ for $1 \le i \le m$, and T is strictly upper triangular. The values x and y are randomly and uniformly distributed in the interval [0, 100], and the entries of T are uniformly distributed in the interval $[-1, 1]$. We have tested our algorithm in comparison to Newton's method with the following scaling methods: Determinantal, spectral, 2-norm, and Frobenious. The baby-step length is fixed at $r = 5$, and the giant-step iterations are terminated when $\|S[k] - S^{-1}[k]\| \le 10^{-10}$. The number of giant steps are given in Table II.

Example 2: Let $A = UDU^*$, where $D = \text{randsvd}\,(n, \kappa, \text{mode}, kl, ku)$ is a (kl, ku) banded random matrix from [2], and U is an arbitrary unitary matrix. We choose $\kappa(D) = 1000$ and equal lower and upper bandwidth of 5, 10 and 20 for orders of $n = 10$, 20 and 40, respectively. The mode is chosen so that the matrix will have arithmetically distributed singular values. The same termination criterion as in Example 1 is used, and the results are shown in Table II.

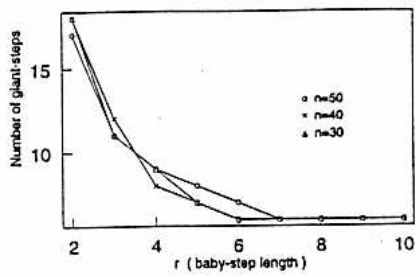

Fig. 1. Number of giant steps versus baby-step length r for Example 4.

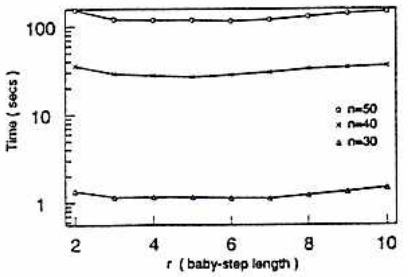

Fig. 2. Total CPU time versus baby-step length r for Example 4.

Example 3: We use the stiff matrix

$$A = \begin{bmatrix} 1 & 0 & 0 & 0 \\ -1 & 0.01 & 0 & 0 \\ -1 & -1 & 100 & 100 \\ -1 & -1 & -100 & 100 \end{bmatrix}$$

where $\sigma(A) = \{.01, 1, 100 \pm 100j\}$, as given in [1, 8]. The last row of Table II gives the number of giant steps for the aforementioned accelerated Newton methods and the new algorithm with $r = 5$ for computing the signal function of this matrix. The stopping criterion is the same as in Example 1.

Example 4: Finally we investigate the relationship between the number of giant steps and the baby-step length r. We have measured the CPU time as a function of r for matrices of various dimensions. The matrices have real entries uniformly distributed in the interval $[-1000, 1000]$. We use the same stopping criterion as the previous examples. Figs. 1 and 2 show the number of giant steps and the computation time for matrices of dimensions 10, 30, and 50. Fig. 1 indicates that the best values of r are between two and eight, while Fig. 2 indicates that total CPU time is almost constant for different values of r. Small values of r introduce large values of giant steps, and as r increases, there is a reduction in the number of giant steps: This results in approximately the same amount of computational work for each case. The total CPU time is minimized, however, when r is approximately equal to five or six.

VI. Conclusion

We have described an iterative algorithm for the computation of the matrix sign function, which is based on the computation of the inverse square root using the continued fraction expansion. The algorithm iteratively computes the main diagonal Padé approximants and has the following properties:

- The convergence rate is a function of the predetermined baby-step length r. For various values of r, we obtain iterative algorithms with different orders of convergence. Our experiments indicate that for matrices of moderate size, one should choose $r \leq 8$ (see Fig. 1). When scaling is not employed, the iteration converges after approximately seven giant-steps for the examples we have used.
- The algorithm is multiplication-rich. Matrix inversions are performed only for each giant-step iteration. During the baby-step phases, we perform only matrix multiplications and additions. It is thus very suitable for implementation on vector and parallel computers.
- The algorithm seems to have good error propagation properties: Errors which occur during a step are either suppressed or have limited effect on the next step.
- The algorithm without any scaling favorably compares to the scaling versions of Newton's method. The speed of convergence of the proposed algorithm can further be improved by scaling the matrix S at each step of the iteration.

We are currently working on parallel implementation of the algorithm on an iPSC/860 and investigating the effects of scaling on the speed of convergence. Another modification of the proposed algorithm can be obtained by changing the value of r at each giant step. This feature of the algorithm provides flexibility in dealing with ill-conditioned matrices: We may start with a small value of r and increase it at each giant step. This way the algorithm computes a different main diagonal Padé approximant at each iteration. We are also investigating this aspect of the algorithm in relation to the speed of the convergence.

References

[1] N. J. Higham, "Newton's method for the matrix square root," *Math. Computation*, vol. 46, no. 174, pp. 537–549, 1986.
[2] ——, "A collection of test matrices in MATLAB," *ACM Trans. Math. Software*, vol. 17, no. 3, pp. 289–305, 1991.
[3] C. Kenney and A. J. Laub, "Polar decomposition and matrix sign function condition estimates," *SIAM J. Scientific and Statist. Computing*, vol. 12, no. 3, pp. 488–504, 1991.
[4] C. Kenney and A. J. Laub, "Rational iterative methods for the matrix sign function, *SIAM J. Matrix Anal. Appl.*, vol. 12, no. 2, pp. 273–291, 1991.
[5] C. Kenney and A. J. Laub, "On scaling Newton's method for polar decomposition and the matrix sign function," *SIAM J. Matrix Anal. Appl.*, vol. 13, no. 3, pp. 688–706, 1992.
[6] A. N. Khovanskii, *The Application of Continued Fractions and Their Generalizations to Problems in Approximation Theory.* Groningen, The Netherlands: P. Noordhof, Ltd., 1963.
[7] J. D. Roberts, "Linear model reduction and solution of the algebraic Riccati equation by use of the sign function," *Int. J. Contr.*, vol. 32, no. 4, pp. 677–687, 1980.
[8] L. S. Shieh, S. R. Lian, and B. C. McInnis, "Fast and stable algorithms for computing the principle square root of a complex matrix," *IEEE Trans. Automat. Contr.*, vol. 32, no. 9, pp. 819–822, 1987.
[9] G. W. Stewart, *Introduction to Matrix Computations.* New York: Academic, 1973.
[10] J. H. Wilkinson, *The Algebraic Eigenvalue Problem.* Oxford, London: Oxford Univ. Press, 1965.