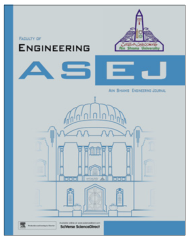
Ain Shams University

Ain Shams Engineering Journal

www.elsevier.com/locate/asej
www.sciencedirect.com



ENGINEERING PHYSICS AND MATHEMATICS

Approximating the principal matrix square root using some novel third-order iterative methods

Amir Sadeghi

Department of Mathematics, Robat Karim Branch, Islamic Azad University, Tehran, Iran

Received 7 December 2015; revised 5 April 2016; accepted 8 June 2016
Available online 28 June 2016

KEYWORDS

Matrix square root;
Positive definite matrix;
Iterative method;
Convergence;
Stability

Abstract It is known that the matrix square root has a significant role in linear algebra computations arisen in engineering and sciences. Any matrix with no eigenvalues in $\mathbb{R}^-$ has a unique square root for which every eigenvalue lies in the open right half-plane. In this research article, a relationship between a scalar root finding method and matrix computations is exploited to derive new iterations to the matrix square root. First, two algorithms that are cubically convergent with conditional stability will be proposed. Then, for solving the stability issue, we will introduce coupled stable schemes that can compute the square root of a matrix with very acceptable accuracy. Furthermore, the convergence and stability properties of the proposed recursions will be analyzed in details. Numerical experiments are included to illustrate the properties of the modified methods.

1. Introduction

Let $\mathbf{A} \in \mathbb{C}^{n\times n}$ has no eigenvalues in $\mathbb{R}^-$, and then the matrix square root may be defined as following integral representation [10]:

$$\mathbf{A}^{1/2} = \frac{2}{\pi}\mathbf{A}\int_0^\infty (t^2\mathbf{I}_n + \mathbf{A})^{-1}\,dt. \tag{1.1}$$

We will almost exclusively be concerned with the principal square root, $\mathbf{A}^{1/2}$. Recall that for $\mathbf{A}$ with no eigenvalues on $\mathbb{R}^-$, $\mathbf{A}^{1/2}$ is the unique square root $\mathbf{X}$ of $\mathbf{A}$ whose spectrum lies in the open right half-plane. The matrix $\sqrt{\mathbf{A}}$ denotes an arbitrary, possibly non-principal square root of $\mathbf{A}$. However, matrix equations particularly the equation $\mathbf{X}^2 - \mathbf{A} = 0$, often appear in mathematical models not only in applied mathematics but also in some other scientific disciplines. For this reason, this problem has been studied by many authors during the past decades, especially from the computational point of view. In general, there are four main approaches to the computation of matrix square root: series approximation (using a proper constituent matrix) [4,19], Schur decomposition methods [3,23], integral representation estimation [13,20], and iterative methods [2,6–9,12,14,15,17,18,21]. In this paper, we restrict our attention to the iterative methods.

Matrix iterations $\mathbf{X}_{k+1} = \psi(\mathbf{X}_k)$, where ψ is a polynomial or a rational function, are attractive alternatives for computing square roots for two reasons: they are readily implemented in terms of standard "building block", and they are potentially well suited to parallel computation. One of the most popular families of recursion is Newton's iteration. The classical full Newton's iteration is in the following form [7]:

E-mail address: drsadeghi.iau@gmail.com
Peer review under responsibility of Ain Shams University.

Production and hosting by Elsevier

http://dx.doi.org/10.1016/j.asej.2016.06.004

$$\begin{cases} \mathbf{X}_k\mathbf{H}_k + \mathbf{H}_k\mathbf{X}_k = \mathbf{A} - \mathbf{X}_k^2, \\ \mathbf{H}_{k+1} = \mathbf{H}_k + \mathbf{X}_k \end{cases} \quad k = 1, 2, \ldots \tag{1.2}$$

for suitable initial values $\mathbf{X}_0$, provides a sequence $\{\mathbf{X}_k\}_{k=1}^{\infty}$ which converges to $\mathbf{A}^{1/2}$. This method has good properties of convergence and stability, but it is too expensive in terms of arithmetic operations. The large cost of this computation makes Newton's iteration useful only as an iterative refinement. The Newton's iteration [7]:

$$\mathbf{X}_0 = \mathbf{A}, \quad \mathbf{X}_{k+1} = \frac{1}{2}\left(\mathbf{X}_k + \mathbf{X}_k^{-1}\mathbf{A}\right), \quad k = 1, 2, \ldots \tag{1.3}$$

has good theoretical properties. If the matrix $\mathbf{A}$ be symmetric positive definite, provided all the iterates in (1.3) are defined, then $\{\mathbf{X}_k\}_{k=1}^{\infty}$ converges quadratically to $\mathbf{A}^{1/2}$ [7,8]. Unfortunately, this iteration has such poor numerical stability that it is useless for practical computation. An alternative iteration was derived by Denman and Béavers [6] as a special case of a method for solving the algebraic Riccati equation:

$$\begin{cases} \mathbf{X}_0 = \mathbf{I}_n, \quad \mathbf{M}_0 = \mathbf{A}, \\ \mathbf{X}_{k+1} = \frac{1}{2}\left(\mathbf{X}_k + \mathbf{M}_k^{-1}\right), \quad k = 1, 2, \ldots \\ \mathbf{M}_{k+1} = \frac{1}{2}\left(\mathbf{M}_k + \mathbf{X}_k^{-1}\right). \end{cases} \tag{1.4}$$

For a general matrix, if we implement the iterations using LU factorization then the iteration (1.4) requires fifty percent more arithmetic operations per iteration than the Newton's iteration (1.3), but for symmetric positive definite matrices the operation counts are the same if we make use of Cholesky factorizations. These iterations require more storage than the Newton's iteration, but this is not a major drawback since k is not usually very large in practice. The crucial property of the Denman and Béavers iteration is that it propagates errors in a stable fashion. Lakić [15] proposed iterative methods for finding the matrix square root that possesses a high computational efficiency. Properties of this sequence are the basis for the construction of the third order algorithm for computing the matrix square root of $\mathbf{A}$,

$$\mathbf{X}_0 = \mathbf{I}_n, \; \mathbf{X}_{k+1} = \frac{3}{8}\mathbf{X}_k + \frac{3}{4}\mathbf{A}\mathbf{X}_k^{-1}\left(\mathbf{I}_n - \frac{1}{6}\mathbf{A}\mathbf{X}_k^{-2}\right), \; k = 1, 2, \ldots \tag{1.5}$$

An analysis of its numerical stability shows that the algorithm is conditionally stable under some restrictions on the eigenvalues of $\mathbf{A}$. An alternative method for computing $\mathbf{A}^{1/2}$, which does not suffer from numerical instability as follows [15]:

$$\begin{cases} \mathbf{X}_0 = \mathbf{I}_n, \quad \mathbf{M}_0 = \mathbf{A}, \\ \mathbf{X}_{k+1} = \mathbf{X}_k\left(\frac{3}{8}\mathbf{I}_n + \frac{3}{4}\mathbf{M}_k\left(\mathbf{I}_n - \frac{1}{6}\mathbf{M}_k\right)\right), \quad k = 1, 2, \ldots \\ \mathbf{M}_{k+1} = \mathbf{M}_k\left(\frac{3}{8}\mathbf{I}_n + \frac{3}{4}\mathbf{M}_k\left(\mathbf{I}_n - \frac{1}{6}\mathbf{M}_k\right)\right)^{-2}. \end{cases} \tag{1.6}$$

More recently, Soleymani et al. [22] proposed an iterative methods to the matrix square root. They have analytically shown that their scheme is asymptotically stable. In addition, convergence analysis along with the error bounds of the main proposed method is established in their article. Moreover, Li and Shen [16] proposed two new algorithms to compute the nonsingular square root of a matrix $\mathbf{A}$ by applying Newton's method. Convergence theorems and stability analysis have been explored in depth for the new algorithms.

In this paper, we will introduce some iterative approaches for computing square root of matrix $\mathbf{A}$ which does not have eigenvalues on $\mathbb{R}^-$ based on a scalar root finding method. It is proved that the scalar iterative method is cubically convergence and then under some particular conditions the matrix case also is cubically convergence. Furthermore, convergence and stability of the proposed iteration are studied in detail by proving some theorems. For solving instability of the algorithm, utilizing axillary variable, a new coupled iterative method is introduced which is stable. The numerical experiments established that the results will be very feasible for matrices with eigenvalues less that one. For this purpose, normalization of the matrices has been used for getting better accuracy. Numerical implementations have been carried out to reveal the properties of the modified theory. Throughout this research article, the following notation will be appeared. If $\mathbf{W} \in \mathbb{C}^{n\times n}$ with eigenvalues $\lambda_1, \ldots, \lambda_n$, then the spectrum of $\mathbf{W}$ is defined by $\sigma(\mathbf{W}) = \{\lambda_1, \ldots, \lambda_n\}$, and the spectral radius of $\mathbf{W}$ is defined by $\rho(\mathbf{W}) = \max_i|\lambda_i|$.

2. New iterative methods

In this section, we provide some conditions to introduce new iterative method to evaluate the principal matrix square root. First the principal square root of scalar a is defined.

Definition 1. Let $a \in \mathbb{R}$ and $a \geqslant 0$. The principal square root of scalar a is defined as unique real number $\sqrt{a}$, where $\sqrt{a} \geqslant 0$.

In next step, we first apply the following iterative scheme introduced in [1]:

$$x_{k+1} = x_k - \mathcal{L}_f(x_k)\frac{f(x_k)}{f'(x_k)}, \quad k = 1, 2, \ldots \tag{2.1}$$

wherein

$$\mathcal{L}_f(x_k) = 1 + \frac{1}{2}\frac{f''(x_k)f(x_k)}{f'^2(x_k)} + \frac{1}{2}\frac{f''^2(x_k)f^2(x_k)}{f'^4(x_k)}, \tag{2.2}$$

to the function $f(x) = x^2 - a$. Therefore, we obtain the following iterations:

$$x_0 = 1, \; x_{k+1} = \frac{5}{16}x_k + \frac{15a}{16x_k} - \frac{5a^2}{16x_k^3} + \frac{a^3}{16x_k^5}, \; k = 1, 2, \ldots \tag{2.3}$$

for finding the square root of a. This iterative scheme plays a major role for proposing a new cubically convergent iterative method to the matrix square root, which is the aim of this paper. Thus, we present the following theorem.

Theorem 2.1. *Assume $a \in \mathbb{R}$, and $0 < a \leq 1$. For the sequence $\{x_k\}_{k=1}^{\infty}$ defined by (2.1), the following items are hold:*

1. $x_{k+1} - \sqrt{a} = \frac{10x_k^3 - 2\sqrt{a}x_k^2 - 6ax_k - 2a\sqrt{a}}{32x_k^5}(x_k - \sqrt{a})^3$;
2. $\sqrt{a} < x_k \leqslant 1$;
3. $\lim_{k\to\infty} x_k = \sqrt{a}$;*If $a = 0$, then we have*
4. $0 < x_k < 1$;
5. $\lim_{k\to\infty} x_k = 0$.

Proof. First part can be easily proved by elementary calculations. We prove the second part by induction over k. Let $\sqrt{a} < x_k \leqslant 1$. Then from part (1), it follows $x_k - \sqrt{a} > 0$. Now, consider the scalar function

$$f(t) = \frac{5t}{16} + \frac{15a}{16t} - \frac{5a^2}{16t^3} + \frac{a^3}{16t^5}. \tag{2.4}$$

It is apparent that for $a \in (0, 1]$ we have

$$x_1 = f(x_0) = f(1) = \frac{1}{16}(5 + 15a - 5a^2 + a^3) \\ = g(a) \in (g(0), g(1)] = \left(\frac{5}{16}, 1\right].$$

which means that $x_1 \leq 1 = x_0$. On the other hand, using the elementary inequality we attain $g(a) = \frac{1}{16}(5 + 15a - 5a^2 + a^3) \geqslant \sqrt{a}$ for $a \in (0, 1]$, and thus $x_1 = g(a) \geqslant \sqrt{a}$. Moreover, since

$$f'(t) = \frac{5}{16} - \frac{15a}{16t^2} + \frac{15a^2}{16t^4} - \frac{5a^3}{16t^6} = \frac{5}{16}\left(1 - \frac{a}{t^2}\right)^3 < 0,$$

it can be concluded that f is monotonously decreasing. This gives the upper bound in part (2) because

$$\sqrt{a} \leqslant x_{k+1} = f(x_k) \leqslant f(1) = f(x_0) = x_1 \leqslant 1.$$

From part (2) and the monocity, it follows that the sequence x_k is convergent. Setting $\lim_{k\to\infty} x_k = \ell$, we obtain the following equation

$$11\ell^6 - 15a\ell^4 + 5a^2\ell^2 - a^3 = 0, \tag{2.5}$$

which has the solutions $\pm\sqrt{a}$, $\pm\sqrt{\frac{2-i\sqrt{7}a}{11}}$, and $\pm\sqrt{\frac{2+i\sqrt{7}a}{11}}$. Since only $\sqrt{a}$ belongs to the interval $[\sqrt{a}, 1]$; therefore, it follows that $\ell = \lim_{k\to\infty} x_k = \sqrt{a}$. In particular case $a = 0$, we have

$$0 < x_{k+1} = \frac{5}{16}x_k \leqslant \frac{5}{16} < 1$$

which implies part (4) and part (5). □

Theorem 2.1 shows that the sequence x_k is cubically convergent for $a > 0$, and only linearly with the rate $\frac{5}{16}$ for $a = 0$. Now, we ready to generalize the iterations from the scalar case to matrix version. First, the principal matrix square root is defined in the following more precisely.

Definition 2. Let $\mathbf{A} \in \mathbb{C}^{n\times n}$ has no eigenvalues on $\mathbb{R}^-$. There is a unique square root $\mathbf{X}$ of $\mathbf{A}$ which all of its eigenvalues lie in the right half-plane, is the solution of the matrix equation $\mathbf{X}^2 - \mathbf{A} = 0$, and it is a primary function of $\mathbf{A}$. The matrix $\mathbf{X} = \mathbf{A}^{1/2}$ is called the principal matrix square root.

It should be pointed out that in Definition 2, if $\mathbf{A}$ is real then $\mathbf{X}$ also is real, and then each eigenvalue of $\mathbf{X}$ is the principal square root of λ_i. Now, let $\mathbf{X}_0 = \alpha\mathbf{I}_n$ or $\mathbf{X}_0 = \beta\mathbf{A}$ denote initial approximation, whenever α and β are appropriate parameters to ensure convergence, and $\mathbf{I}_n$ is the identity matrix of order n. Consequently, new iterative method based on real sequence (2.1) will propose in the following algorithm.

Algorithm 2.1. Let $\mathbf{A}$ be an $n \times n$ matrix. The new iterative method for computing the square root of $\mathbf{A}$ is expressed by:

$$\mathbf{X}_0 = \alpha\mathbf{I}_n, \quad \text{or } \mathbf{X}_0 = \beta\mathbf{A},$$

$$\mathbf{X}_{k+1} = \frac{5}{16}\mathbf{X}_k + \frac{1}{16}\mathbf{A}\mathbf{X}_k^{-1}(15\mathbf{I}_n - 5\mathbf{A}\mathbf{X}_k^{-2} + \mathbf{A}^2\mathbf{X}_k^{-4}) \quad k = 1, 2, \ldots$$

Lemma 2.1. *Let $\mathbf{A} \in \mathbb{C}^{n\times n}$ is a nonsingular matrix which has not eigenvalues on $\mathbb{R}^-$. If $\mathbf{A}\mathbf{X}_0 = \mathbf{X}_0\mathbf{A}$ is valid, then for the sequence $\{\mathbf{X}_k\}_{k=0}^{\infty}$ in Algorithm 2.1, one has that*

$$\mathbf{A}\mathbf{X}_k = \mathbf{X}_k\mathbf{A} \tag{2.6}$$

holds for all $k = 1, 2, \ldots$.

Proof. By using a similar strategy that is mentioned in [7], lemma can be easily proven. □

In the following theorem, we establish the cubically convergence of the sequence $\{\mathbf{X}_k\}_{k=1}^{\infty}$.

Theorem 2.2. *Let $\mathbf{A} \in \mathbb{C}^{n\times n}$ is a diagonalizable matrix, then for the matrix sequence $\{\mathbf{X}_k\}_{k=1}^{\infty}$ defined by Algorithm 2.1, we have $\lim_{k\to\infty} \mathbf{X}_k = \mathbf{A}^{1/2}$, where $\mathbf{A}^{1/2}$ is the principal square root of $\mathbf{A}$, and for $\lambda_i > 0$ we have the following upper bound*

$$\|\mathbf{X}_{k+1} - \mathbf{A}^{1/2}\| \leqslant \frac{1}{32}\|\mathbf{X}_k^{-1}\|^5 \\ \cdot \left\|10\mathbf{X}_k^3 - 2\mathbf{A}^{1/2}\mathbf{X}_k - 6\mathbf{A}\mathbf{X}_k - 2\mathbf{A}\mathbf{A}^{1/2}\right\| \\ \cdot \left\|\mathbf{X}_k - \mathbf{A}^{1/2}\right\|^3. \tag{2.7}$$

Proof. From diagonalizability of the matrix $\mathbf{A}$, it follows that if $\mathbf{A}$ is singular then $\mathbf{0}$ is at most a simple zero of the minimal polynomial of $\mathbf{A}$ which means that there exist $\mathbf{A}^{1/2}$ in the singular case. Hence, consider $\mathbf{A}$ is a regular matrix such that

$$\mathbf{W}^{-1}\mathbf{A}\mathbf{W} = \mathbf{\Lambda} = \operatorname{diag}\{\lambda_1, \ldots, \lambda_n\}, \tag{2.8}$$

where $\mathbf{W}$ is a nonsingular matrix. Define $\mathbf{D}_k = \mathbf{W}^{-1}\mathbf{X}_k\mathbf{W}$. According to iterative scheme in Algorithm 2.1, we then have

$$\mathbf{D}_0 = \mathbf{I}_n,$$

$$\mathbf{D}_{k+1} = \frac{5}{16}\mathbf{D}_k + \frac{1}{16}\mathbf{\Lambda}\mathbf{D}_k^{-1}(15\mathbf{I}_n - 5\mathbf{\Lambda}\mathbf{D}_k^{-2} + \mathbf{\Lambda}^2\mathbf{D}_k^{-4}),$$

where $\{\mathbf{D}_k\}_{k=1}^{\infty}$ is the sequence of real block diagonal matrices $\mathbf{D}_k = \operatorname{diag}\left\{d_1^{(k)}, \ldots, d_n^{(k)}\right\}$. Thus, the recursive relation $\{\mathbf{D}_{k+1}\}_{k=1}^{\infty}$ is equivalent to n scalar iterative scheme

$$d_0^{(k)} = 1,$$

$$d_i^{(k+1)} = \frac{5}{16}\left(d_i^{(k)}\right) + \frac{15\lambda_i}{16\left(d_i^{(k)}\right)} - \frac{5\lambda_i^2}{16\left(d_i^{(k)}\right)^3} + \frac{\lambda_i^3}{16\left(d_i^{(k)}\right)^5}.$$

Since $\{\mathbf{D}_k\}_{k=1}^{\infty}$ is a real sequence, employing Theorem 2.1, we obtain

$$\lim_{k\to\infty} \mathbf{D}_k = \mathbf{\Lambda}^{1/2} = \operatorname{diag}\left\{\sqrt{\lambda_1}, \ldots, \sqrt{\lambda_n}\right\}.$$

Consequently, it can be yielded that $\lim_{k\to\infty} \mathbf{X}_k = \mathbf{V}\mathbf{\Lambda}^{1/2}\mathbf{V}^{-1} = \mathbf{A}^{1/2}$. According to first part of Theorem 2.1, we have

$$\mathbf{D}_{k+1} - \mathbf{\Lambda}^{1/2} = \frac{1}{32}(10\mathbf{D}_k^3 - 2\mathbf{\Lambda}^{1/2}\mathbf{D}_k - 6\mathbf{\Lambda}\mathbf{D}_k - 2\mathbf{\Lambda}\mathbf{\Lambda}^{1/2}) \\ \times (\mathbf{D}_k^{-5})(\mathbf{D}_k - \mathbf{\Lambda}^{1/2})^3.$$

Performing similarity transformation by $\mathbf{W}$ gives us

$$\mathbf{X}_{k+1} - \mathbf{A}^{1/2} = \frac{1}{32}\left(10\mathbf{X}_k^3 - 2\mathbf{A}^{1/2}\mathbf{X}_k - 6\mathbf{A}\mathbf{X}_k - 2\mathbf{A}\mathbf{A}^{1/2}\right) \times \left(\mathbf{X}_k^{-5}\right)\left(\mathbf{D}_k - \mathbf{A}^{1/2}\right)^3.$$

Subsequently, taking any subordinate norms in the last relation we get

$$\|\mathbf{X}_{k+1} - \mathbf{A}^{1/2}\| \leqslant \frac{1}{32}\|\mathbf{X}_k^{-1}\|^5 \cdot \|10\mathbf{X}_k^3 - 2\mathbf{A}^{1/2}\mathbf{X}_k - 6\mathbf{A}\mathbf{X}_k - 2\mathbf{A}\mathbf{A}^{1/2}\| \cdot \|\mathbf{X}_k - \mathbf{A}^{1/2}\|^3.$$

This ends the proof. □

Theorem 2.2 is illustrated that the order of convergence of the matrix sequence $\{\mathbf{X}_k\}_{k=1}^{\infty}$ given by Algorithm 2.1 is equal to 3, provided that $\mathbf{A}$ is a nonsingular matrix. It must be emphasized that if $\mathbf{A}$ is singular matrix, according to Theorem 2.1, $\{\mathbf{X}_k\}_{k=1}^{\infty}$ converges only linearly with the rate $\frac{5}{16}$. Now we are interested to apply normalization of the matrix as follows.

Remark 1. If $\rho(\mathbf{A}) > 1$, then we substitute $\mathbf{B} = \mathbf{A}/\|\mathbf{A}\|$. In this case, it is clear that $\rho(\mathbf{B}) \leqslant 1$. Subsequently, the matrix sequence $\{\mathbf{R}_k\}_{k=1}^{\infty}$ can be computed as following.

Algorithm 2.2. Let $\mathbf{A} \in \mathbb{C}^{n\times n}$ and $\mathbf{B} = \mathbf{A}/\|\mathbf{A}\|$. The iterative method for computing the square root of $\mathbf{A}$ with $\rho(\mathbf{A}) > 1$ is expressed as:

$$\mathbf{R}_0 = \mathbf{I}_n,$$

$$\mathbf{R}_{k+1} = \frac{5}{16}\mathbf{R}_k + \frac{1}{16}\mathbf{B}\mathbf{R}_k^{-1}\left(15\mathbf{I}_n - 5\mathbf{B}\mathbf{R}_k^{-2} + \mathbf{B}^2\mathbf{R}_k^{-4}\right),$$

$$\mathbf{X}_k = \sqrt{\|\mathbf{A}\|}\mathbf{R}_k.$$

In Algorithm 2.2, it is apparent that $\lim_{k\to\infty}\mathbf{R}_k = \mathbf{B}^{1/2}$, and $\lim_{k\to\infty}\mathbf{X}_k = \mathbf{A}^{1/2}$. We then have

$$\|\mathbf{X}_{k+1} - \mathbf{A}^{1/2}\| = \sqrt{\|\mathbf{A}\|}\cdot\|\mathbf{R}_{k+1} - \mathbf{B}^{1/2}\| = \mathcal{O}\left(\|\mathbf{X}_k - \mathbf{A}^{1/2}\|^3\right), \tag{2.9}$$

where $\mathcal{O}$ denotes big O. Notice that the matrix $\mathbf{B}$ could also be proposed by $\mathbf{B} = \mathbf{A}/\rho(\mathbf{A})$ if $\rho(\mathbf{A})$ is available. Since $\rho(\mathbf{A}) \leqslant \|\mathbf{A}\|$, it is more appropriate that the upper bound $\|\mathbf{A}\|$ will be considered. It should be pointed out that when $\mathbf{A}$ is nonsingular, since $\lim_{k\to\infty}\mathbf{X}_k = \mathbf{A}^{1/2}$, this concludes that $\lim_{k\to\infty}\mathbf{X}_k^{-1} = \mathbf{A}^{-1/2}$, that is called the principal inverse matrix square root.

Remark 2. It is straightforward that even the scalar iterations of (2.3) has a fractal behavior. It leads us to explore the existence of regions where the iterates converge to fixed points of the function. We utilized MATLAB (2014Ra) with a square of 250,000 points to generate plots of the set of points for which the iteration converges to a specific root and also their boundary points of the iterates. The associated picture for $a = 1$ is shown in Fig. 1. It must be pointed out that we would like to obtain the principal root. As can be seen for any point belongs to nonnegative real axis, the iteration (2.3) converge to the principal square root. Thus, for positive definite matrix, the iterations (2.3) converges to the unique positive definite square root of $\mathbf{A}$ where the initial guess matrix is positive definite. The

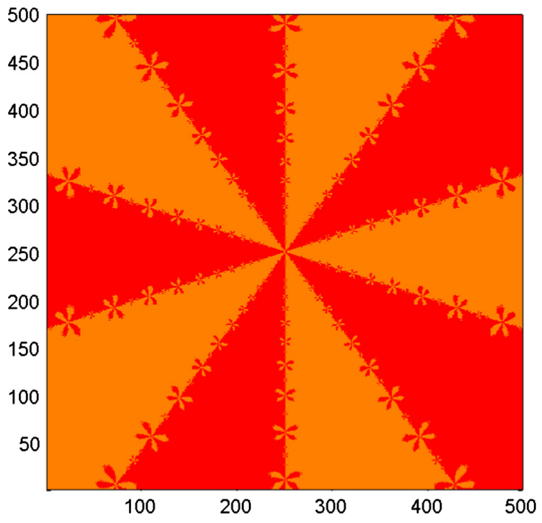


Figure 1 Fractal behavior of iterations (2.3) for $a = 1$.

similar result is given in [23] for the computation of matrix p^{th} root by Newton's iterations.

Now, we are interested to know that whether the iterations in Algorithm 2.1 are stable or not? First we give the following theorem.

Theorem 2.3. *The sequence* $\{\mathbf{X}_k\}_{k=1}^{\infty}$ *introduced in Algorithm 2.1 is conditionally stable.*

Proof. The proof of this theorem is based on strategy that has been applied in [7,15]. It is hence omitted. □

As can be seen in Remark 2, the proposed iterations in Algorithm 2.1 may have fractal behavior. Moreover, according to Theorem 2.3, it was seen that the proposed scheme has turn out to be stable whenever $\mathbf{A}$ is ill-conditioned or the size of the input matrix $\mathbf{A}$ is large. Hence, the condition on the matrix $\mathbf{A}$ is very restrictive and consequently the proposed iteration is not much practical use for calculating the principal matrix square root. For solving this issue, two alternative stable and convergent iterations will be proposed by employing matrix auxiliary variables in the next section.

3. New stable iterative schemes

In this section, new stable variant of iterative method for computing the principal matrix square root will be introduced. For this purpose, we first consider an auxiliary variable in the form $\mathbf{M}_k = \mathbf{A}\mathbf{X}_k^{-2}$. It can be easily shown that $\lim_{k\to\infty}\mathbf{X}_k = \mathbf{I}_n$ and $\lim_{k\to\infty}\mathbf{M}_k = \mathbf{A}$. Furthermore, the set of the matrices $\{\mathbf{X}_k, \mathbf{M}_k, \mathbf{A}\}$ commuting set. Now, the new variant of the matrix iterations is obtained as following:

$$\begin{aligned}\mathbf{X}_{k+1} &= \frac{5}{16}\mathbf{X}_k + \frac{1}{16}\mathbf{A}\mathbf{X}_k^{-1}\left(15\mathbf{I}_n - 5\mathbf{A}\mathbf{X}_k^{-2} + \mathbf{A}^2\mathbf{X}_k^{-4}\right)\\ &= \mathbf{X}_k\left(\frac{5}{16}\mathbf{I}_n + \frac{1}{16}\mathbf{A}\mathbf{X}_k^{-2}\left(15\mathbf{I}_n - 5\mathbf{A}\mathbf{X}_k^{-2} + \mathbf{A}^2\mathbf{X}_k^{-4}\right)\right)\\ &= \mathbf{X}_k\left(\frac{5}{16}\mathbf{I}_n + \frac{1}{16}\mathbf{M}_k\left(15\mathbf{I}_n - 5\mathbf{M}_k + \mathbf{M}_k^2\right)\right),\end{aligned}$$

and

$$\begin{aligned}\mathbf{M}_{k+1} &= \mathbf{A}\mathbf{X}_{k+1}^{-2}\\ &= \mathbf{A}\mathbf{X}_k^{-2}\left(\frac{5}{16}\mathbf{I}_n+\frac{1}{16}\mathbf{A}\mathbf{X}_k^{-2}\left(15\mathbf{I}_n-5\mathbf{A}\mathbf{X}_k^{-2}+\mathbf{A}^2\mathbf{X}_k^{-4}\right)\right)^{-2}\\ &= \mathbf{M}_k\left(\frac{5}{16}\mathbf{I}_n+\frac{1}{16}\mathbf{M}_k\left(15\mathbf{I}_n-5\mathbf{M}_k+\mathbf{M}_k^2\right)\right)^{-2}.\end{aligned}$$

Consequently, the following algorithm will be given.

Algorithm 3.1. Let $\mathbf{A}\in\mathbb{C}^{n\times n}$. The coupled iterative method for computing the square root of $\mathbf{A}$ is defined by the recursive relations

$$\mathbf{X}_0=\mathbf{I}_n,\quad \mathbf{M}_0=\mathbf{A}$$

$$\mathbf{X}_{k+1}=\mathbf{X}_k\left(\frac{5}{16}\mathbf{I}_n+\frac{1}{16}\mathbf{M}_k\left(15\mathbf{I}_n-5\mathbf{M}_k+\mathbf{M}_k^2\right)\right),$$

$$\mathbf{M}_{k+1}=\mathbf{M}_k\left(\frac{5}{16}\mathbf{I}_n+\frac{1}{16}\mathbf{M}_k\left(15\mathbf{I}_n-5\mathbf{M}_k+\mathbf{M}_k^2\right)\right)^{-2}.$$

In Algorithm 3.1, it is straightforward that whenever $\lim_{k\to\infty}\mathbf{X}_k=\mathbf{A}^{1/2}$, then $\lim_{k\to\infty}\mathbf{M}_k=\mathbf{I}_n$. In this part, we present stability analysis of the coupled iteration for computing matrix square roots. According to [5], an iteration $\mathbf{X}_{k+1}=\psi(\mathbf{X}_k)$ is stable in a neighborhood of a solution $\mathbf{X}=\psi(\mathbf{X})$, if the error matrices $\mathbf{E}_k=\mathbf{X}_k-\mathbf{X}$ satisfy

$$\mathbf{E}_{k+1}=L(\mathbf{E}_k)+\mathcal{O}\left(\|\mathbf{E}_k\|^2\right), \tag{3.1}$$

where L is a linear operator with bounded powers. In other words, there exists a constant $\varepsilon>0$ such that for all $k>0$ and an arbitrary unit norm, we have $L^k(\mathbf{E})<\varepsilon$. This means small perturbation introduced in a certain step will not be amplified in the subsequent iterations. Thus, we give the following theorem.

Theorem 3.1. *The iterations $\{\mathbf{X}_{k+1}\}_{k=1}^{\infty}$ and $\{\mathbf{M}_{k+1}\}_{k=1}^{\infty}$ in Algorithm 3.1 are stable.*

Proof. Consider the iterations in Algorithm 3.1 and introduce the error matrices $\mathbf{E}_k=\mathbf{X}_k-\mathbf{A}^{-1/2}$, and $\mathbf{F}_k=\mathbf{M}_k-\mathbf{I}_n$. For the sake of simplicity, perform a first order error analysis and remove all the terms that are quadratic in the errors. Assume equality up to second order terms are denoted with the symbol $\approxeq$. Thus, from $\mathbf{M}_k=\mathbf{I}_n+\mathbf{F}_k$, one has

$$\begin{aligned}\mathbf{E}_{k+1} &= \mathbf{X}_{k+1}-\mathbf{A}^{1/2}=\mathbf{X}_k\left(\frac{5}{16}\mathbf{I}_n+\frac{1}{16}\mathbf{M}_k\left(15\mathbf{I}_n-5\mathbf{M}_k+\mathbf{M}_k^2\right)\right)-\mathbf{A}^{1/2}\\ &= \mathbf{X}_k\left(\frac{5}{16}\mathbf{I}_n+\frac{1}{16}(\mathbf{I}+\mathbf{F}_k)\left(15\mathbf{I}_n-5(\mathbf{I}_n+\mathbf{F}_k)+(\mathbf{I}_n+\mathbf{F}_k)^2\right)\right)\\ &\quad -\mathbf{A}^{1/2}\approxeq\mathbf{X}_k\left(\frac{5}{16}\mathbf{I}_n+\frac{1}{16}(\mathbf{I}_n+\mathbf{F}_k)(-3\mathbf{F}_k+11\mathbf{I}_n)\right)\\ &\quad -\mathbf{A}^{1/2}\approxeq\mathbf{X}_k\left(\frac{5}{16}\mathbf{I}_n+\frac{11}{16}\mathbf{I}_n+\frac{1}{2}\mathbf{F}_k\right)\\ &\quad -\mathbf{A}^{1/2}=\mathbf{E}_k+\frac{\mathbf{X}_k}{2}\mathbf{F}_k.\end{aligned}$$

Furthermore, we yield

$$\begin{aligned}\mathbf{F}_{k+1} &= \mathbf{M}_{k+1}-\mathbf{I}_n\\ &= \mathbf{M}_k\left(\frac{5}{16}\mathbf{I}_n+\frac{1}{16}\mathbf{M}_k\left(15\mathbf{I}_n-5\mathbf{M}_k+\mathbf{M}_k^2\right)\right)^{-2}\\ &\quad -\mathbf{I}_n\approxeq(\mathbf{I}_n+\mathbf{F}_k)\left(\frac{5}{16}\mathbf{I}_n+\frac{11}{16}\mathbf{I}_n+\frac{1}{2}\mathbf{F}_k\right)\\ &\quad -\mathbf{I}_n=(\mathbf{I}_n+\mathbf{F}_k)(\mathbf{I}_n-\mathbf{F}_k)-\mathbf{I}_n\approxeq\mathbf{0}.\end{aligned}$$

In conclusion, it can be written as

$$\begin{pmatrix}\mathbf{E}_{k+1}\\ \mathbf{F}_{k+1}\end{pmatrix}=\begin{pmatrix}\mathbf{I}_n & \frac{\mathbf{X}_k}{2}\\ \mathbf{0} & \mathbf{0}\end{pmatrix}\begin{pmatrix}\mathbf{E}_k\\ \mathbf{F}_k\end{pmatrix}=L\begin{pmatrix}\mathbf{E}_k\\ \mathbf{F}_k\end{pmatrix}. \tag{3.2}$$

The coefficient matrix L is idempotent ($L^2=L$) and hence has bounded powers. Thus the proposed iterations are stable. □

Once again if $\rho(\mathbf{A})>1$, therefore the substitution $\mathbf{B}=\mathbf{A}/\|\mathbf{A}\|$ can be applied. Hence, we propose the following algorithm.

Algorithm 3.2. Let $\mathbf{A}\in\mathbb{C}^{n\times n}$ and $\mathbf{B}=\mathbf{A}/\|\mathbf{A}\|$. The stable coupled iterative method for computing the principal square root of $\mathbf{A}$ is expressed as follows:

$$\mathbf{R}_0=\mathbf{I}_n,\quad \mathbf{M}_0=A$$

$$\mathbf{R}_{k+1}=\mathbf{R}_k\left(\frac{5}{16}\mathbf{I}_n+\frac{1}{16}\mathbf{M}_k\left(15\mathbf{I}_n-5\mathbf{M}_k+\mathbf{M}_k^2\right)\right),$$

$$\mathbf{M}_{k+1}=\mathbf{M}_k\left(\frac{5}{16}\mathbf{I}_n+\frac{1}{16}\mathbf{M}_k\left(15\mathbf{I}_n-5\mathbf{M}_k+\mathbf{M}_k^2\right)\right)^{-2}.$$

$$\mathbf{X}_k=\sqrt{\|\mathbf{A}\|}\mathbf{R}_k.$$

In Algorithm 3.2, it is apparent that $\lim_{k\to\infty}\mathbf{R}_k=\mathbf{B}^{1/2}$, $\lim_{k\to\infty}\mathbf{X}_k=\mathbf{A}^{1/2}$. Thus, the principal matrix square root can be computed efficiently.

4. Numerical experiments and application

In this section, we support the theory which has been developed so far with several numerical implementations. All computations have been carried out by using MATLAB (2014Ra). Matrices that were used are either well-conditioned or ill-conditioned. We also used Higham's Matrix Function Toolbox [11] which is a collection of programs to implement matrix computation methods. In addition, the accuracy is measured by means of the size of:

$$E_k\left(\widehat{\mathbf{X}}\right)=\frac{\left\|\widehat{\mathbf{X}}_k^2-\mathbf{A}\right\|_F}{\|\mathbf{A}\|_F}, \tag{4.1}$$

whenever $\widehat{\mathbf{X}}$ is the computed square root of $\mathbf{A}$, and $\|\cdot\|_F$ is Frobenius norm.

Test 1. In this example, three matrices $\mathbf{A},\mathbf{B}$ and $\mathbf{C}$ that have been utilized in [15] are considered as follows:

$$\mathbf{A} = \begin{pmatrix} 1 & 1 & 1 \\ 1 & 2 & 3 \\ 1 & 3 & 6 \end{pmatrix}, \quad \mathbf{B} = \begin{pmatrix} 5+i & 2+i & 3i \\ 2+i & 5+i & 4+1i \\ 1-2i & 3-2i & 6-2i \end{pmatrix},$$
$$\mathbf{C} = \begin{pmatrix} 4 & 1 & 1 \\ 2 & 4 & 1 \\ 0 & 1 & 4 \end{pmatrix}.$$

The square root of these matrices that have the condition number $\kappa(\mathbf{A}) = 61.9839$, $\kappa(\mathbf{B}) = 9.6468$ and $\kappa(\mathbf{C}) = 2.4642$ is evaluated by numerous methods. We have compared the number of iterations, residual errors and also CPU time in second by performing our proposed iterations and other schemes. The result has been reported in Table 1. It should be noted that in spite of we have used double precision arithmetic precision, we present error by short form. According to the results, we can see that Algorithm 3.2 has very accurate advantage with less number of iterations (and consequently less time consuming in most cases) in comparison with other methods.

Test 2 (*An application*). The quadratic matrix equation $\mathbf{AX}^2 + \mathbf{BX} + \mathbf{C} = \mathbf{0}$, whenever all matrices are $n \times n$ arisen in various applications. Unfortunately, there is no closed-form expression for $\mathbf{X}$ generally, and the theory of such equations is nontrivial (For more information refer to [8,10]). A special case in which the usual quadratic formula generalizes is whenever we put $\mathbf{A} = \mathbf{I}_n$, $\mathbf{B}$ commutes with $\mathbf{C}$, and also $\mathbf{B}^2 - 4\mathbf{C}$ has a square root. Then the solution can be obtained as following:

$$\mathbf{X} = -\frac{1}{2}\mathbf{B} + \frac{1}{2}(\mathbf{B}^2 - 4\mathbf{C})^{1/2}. \tag{4.2}$$

Now, assume $n \times n$ following tridiagonal matrices as

$$\mathbf{B} = \begin{pmatrix} 4 & -2 & & & \\ -2 & 4 & -2 & & \\ & \ddots & \ddots & \ddots & \\ & & -2 & 4 & -2 \\ & & & -2 & 4 \end{pmatrix}_{n\times n},$$
$$\mathbf{C} = \begin{pmatrix} 10 & 1 & & & \\ 1 & 10 & 1 & & \\ & \ddots & \ddots & \ddots & \\ & & 1 & 10 & 1 \\ & & & 1 & 10 \end{pmatrix}_{n\times n}.$$

Notice that $\mathbf{BC} = \mathbf{CB}$. We have computed the solution of the matrix equation $\mathbf{X}^2 + \mathbf{BX} + \mathbf{C} = \mathbf{0}$ by using (4.2). In our implementations, we apply Algorithm 3.2 and well-known Schur method (in MATLAB it is given by `sqrtm`) for computing the matrix $(\mathbf{B}^2 - 4\mathbf{C})^{1/2}$. The residual error for both algorithms has been measured by increasing dimension of the matrices. Results are reported in Fig. 2. It is clear that Algorithm 3.2 can compute square root of a matrix with very feasible accuracy. It must be mentioned that in our implementation, some well-known methods such as Newton's iteration, full Newton's method and DB method have been broke down. Finally, for the special case $n = 6$, we attain:

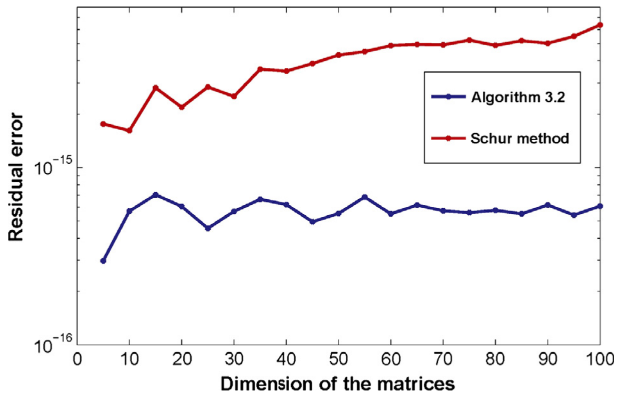


Figure 2 Comparison residual error in Test 2.

$$\mathbf{X} = \begin{pmatrix} -0.9735 & -0.3142 & -0.0792 & -0.0231 & -0.0074 & -0.0023 \\ -0.3142 & -1.0527 & -0.3373 & -0.0866 & -0.0254 & -0.0074 \\ -0.0792 & -0.3373 & -1.0601 & -0.3396 & -0.0866 & -0.0231 \\ -0.0231 & -0.0866 & -0.3396 & -1.0601 & -0.3373 & -0.0792 \\ -0.0074 & -0.0254 & -0.0866 & -0.3373 & -1.0527 & -0.3142 \\ -0.0023 & -0.0074 & -0.0231 & -0.0792 & -0.3142 & -0.9735 \end{pmatrix}.$$

Test 3. This example made considering 20×20 matrices

$$\mathbf{A}_{n\times n} = \mathtt{n} * \mathtt{eye(n)} + \mathtt{hilb(n)}, \quad \text{and}$$
$$\mathbf{B}_{n\times n} = \frac{1}{\sqrt{n}} * \mathtt{rand(n)} + 3 * \mathtt{eye(n)}.$$

We have computed the square root of these matrices by the proposed algorithms and we compared the results by Newton's and Schur methods. The number of iterations, residual error and CPU time in second is given in Table 2. Once again, it is clear that the proposed iterations can compute matrix square root by less step with very feasible accuracy.

Table 1 Comparison errors and iterations number in Test 1.

Method	Iter.	Err. for **A**	CPU time	Iter.	Err. for **B**	CPU time	Iter.	Err. for **C**	CPU time
Algorithm 2.2	5	1.0637×10^{-13}	0.007562	4	1.5384×10^{-15}	0.008056	3	3.0013×10^{-16}	0.005954
Algorithm 3.2	5	3.3100×10^{-16}	0.006792	4	5.5801×10^{-16}	0.006824	3	1.4983×10^{-16}	0.007523
Full Newton's iteration	7	4.4232×10^{-17}	0.014478	7	1.7287×10^{-16}	0.011903	7	8.3924×10^{-17}	0.018647
Newton's iteration (1.3)	–	Not converge	–	–	Not converge	–	5	1.5131×10^{-16}	0.006523
Lakic iteration (1.5)	5	2.4779×10^{-15}	0.007585	5	4.7399×10^{-16}	0.008133	3	2.8149×10^{-16}	0.007395
Lakic iteration (1.6)	5	2.7552×10^{-16}	0.007958	4	4.4074×10^{-16}	0.007116	5	2.7552×10^{-16}	0.007540
DB iteration	5	2.2204×10^{-16}	0.006816	5	4.8942×10^{-16}	0.008036	4	2.1807×10^{-16}	0.007145
Schur method (MATLAB)	–	2.4064×10^{-16}	0.009965	–	2.3540×10^{-16}	0.009939	–	6.8853×10^{-16}	0.013713

Table 2 Comparison error and iterations number in Test 3.

Method	Iter	Err for matrix **A**	CPU time	Iter	Err for matrix **B**	CPU time
Algorithm 2.2	4	3.4411×10^{-16}	0.006858	4	3.8333×10^{-16}	0.034201
Algorithm 3.2	3	2.9010×10^{-16}	0.005970	3	4.6231×10^{-16}	0.006304
Newton's iteration (1.3)	5	1.9861×10^{-16}	0.006892	5	2.2644×10^{-16}	0.018014
Schur method (MATLAB)	–	5.6446×10^{-15}	0.009111	–	5.4266×10^{-15}	0.017298

5. Conclusions

In this work, we have developed the root finding approach to obtain some iterative methods to the square root of matrices that have no eigenvalues on $\mathbb{R}^-$. Numerical experiments indicate that the coupled method which has been proposed give an accurate solution with less number of iterations in comparison with other schemes. Nevertheless, in spite of doing numerous experiments, the flop comparisons between the presented methods cannot give a reliable and definitive answer: which method is the best for general $n \times n$ matrices? It depends on many features such as the structure and order of a given matrix, the wanted accuracy, and perturbations arising from rounding errors.

Acknowledgments

Special thanks go to the anonymous referee for some valuable suggestions, which have resulted in the improvement of this work.

References

[1] Abbasbandy S. Improving Newton–Raphson method for nonlinear equations by modified Adomian decomposition method. Appl Math Comput 2003;145:887–93.
[2] Assimakis N, Adam M. Inversion free algorithms for computing the principal square root of a matrix. Int J Math Math Sci 2014;2014:8. http://dx.doi.org/10.1155/2014/613840 613840.
[3] Björck Å, Hammarling S. A Schur method for the square root of a matrix. Linear Algebra Appl 1983;52/53:127–40.
[4] Chang FC. Function of a square matrix with repeated eigenvalues. Appl Math Comput 2005;160:819–49.
[5] Cheng SH, Higham NJ, Kenney CS, Laub AJ. Approximating the logarithm of a matrix to specified accuracy. SIAM J Matrix Anal Appl 2001;22:1112–25.
[6] Denman E, Béavers N. The matrix sign function and computations in systems. Appl Math Comput 1976;2:63–94.
[7] Higham NJ. Newton's method for the matrix square root. Math Comput 1986;46:537–49.
[8] Higham NJ, Kim HM. Numerical analysis of a quadratic matrix equation. IMA J Numer Anal 2000;20(4):499–519.
[9] Higham NJ. Stable iterations for the matrix square root. Numer Algorithms 1997;15:227–42.
[10] Higham NJ. Functions of matrices: theory and computation. Philadelphia (PA, USA): Society for Industrial and Applied Mathematics; 2008.
[11] Higham NJ. The matrix function toolbox. < http://www.ma.man.ac.uk/higham/mctoolbox > [retrieved on November 3, 2009].
[12] Iannazzo B. A note on computing the matrix square root. Calcolo 2003;40:273–83.
[13] Kellems A. Computing functions of matrices via contour integrals and the Trapezoid rule. Work J 2005.
[14] Lainiotis DG, Assimakis ND, Katsikas SK. Fast and stable algorithm for computing the principal square root of a complex matrix. Neural Parall Sci Comput 1993;1:467–76.
[15] Lakić S. On the matrix square root. ZAMM Z Angew Math Mech 1998;78:173–82.
[16] Li CM, Shen SQ. Newton's method for the matrix nonsingular square root. J Appl Math 2014:7. http://dx.doi.org/10.1155/2014/267042 [Article ID 267042].
[17] Meini B. The matrix square root from a new functional perspective: theoretical results and computational issues. SIAM J Matrix Anal Appl 2005;26(2):362–76.
[18] Psarrakos PJ. On the m^{th} roots of a complex matrix. Electron J Linear Algebra 2002;9:32–41.
[19] Sadeghi A, Ismail AI, Ahmad A. Computing the p^{th} roots of a matrix with repeated eigenvalues. Appl Math Sci 2011;5(53):2645–61.
[20] Sadeghi A, Ismail AI. Approximation of the p^{th} roots of a matrix by using Trapezoid rule. J Math Math Sci 2012;2012:13. http://dx.doi.org/10.1155/2012/634698 634698.
[21] Sadeghi A. A stable coupled Newton's iteration for the matrix inverse p^{th} root. Int J Math Model Comput 2015;5(1):1–11.
[22] Soleymani F, Shateyi S, Khaksar Haghani F. A numerical method for computing the principal square root of a matrix. Abstr Appl Anal 2014;2014:7 . http://dx.doi.org/10.1155/2014/525087 525087.
[23] Smith MI. A Schur algorithm for computing matrix p^{th} roots. SIAM J Matrix Anal Appl 2003;24(4):971–89.

Dr. Amir Sadeghi was born on 1982, Iran. He received his B.Sc. (2005) in Applied Mathematics and he finished his M.Sc. (2007) in Numerical Analysis from Amirkabir University of Technology. Dr. Sadeghi completed his Ph.D from University Science of Malaysia in numerical linear algebra (2012). His interests include numerical methods, matrix equations, Function of matrices, fuzzy sets and systems, and control theory.