mathematics

MDPI

Article

A Sixth-Order Iterative Scheme Through Weighted Rational Approximations for Computing the Matrix Sign Function

Ce Zhang [1], Bo Zhao [2], Wenjing Ren [2], Ruosong Cao [2] and Tao Liu [2,*]

[1] Modern Educational Technology Center, Changchun Guanghua University, Changchun 130033, China; zhangce@ghu.edu.cn
[2] School of Mathematics and Statistics, Northeastern University at Qinhuangdao, Qinhuangdao 066004, China; 202215092@stu.neuq.edu.cn (B.Z); 2472156@stu.neu.edu.cn (W.R.); 2401995@stu.neu.edu.cn (R.C.)
* Correspondence: liutao@neuq.edu.cn

Abstract

This work introduces a sixth-order multi-step iterative algorithm for obtaining the matrix sign function of nonsingular matrices. The presented methodology employs optimized rational approximations combined with strategically formulated weight functions to achieve both computational efficiency and numerical precision. We present a convergence study that includes the analytical derivation of error terms, formally proving the sixth-order convergence characteristics. Numerical simulations substantiate the theoretical results and demonstrate the algorithm's advantage over current state-of-the-art approaches in terms of both accuracy and computational performance.

Keywords: matrix sign function; sixth-order iterative methods; rational approximations; convergence; numerical linear algebra

MSC: 65F60; 41A25

check for updates

Academic Editor: Luca Gemignani

Received: 14 July 2025
Revised: 26 August 2025
Accepted: 1 September 2025
Published: 4 September 2025

Citation: Zhang, C.; Zhao, B.; Ren, W.; Cao, R.; Liu, T. A Sixth-Order Iterative Scheme Through Weighted Rational Approximations for Computing the Matrix Sign Function. *Mathematics* **2025**, *13*, 2849. https://doi.org/10.3390/math13172849

Copyright: © 2025 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (https://creativecommons.org/licenses/by/4.0/).

1. Introduction

The matrix sign function (MSF) constitutes a fundamental computational tool with far-reaching applications across scientific computing and numerical linear algebra. As established in [1] (Chapter 5), its theoretical foundation stems from the scalar sign function expressed for $\zeta \in \mathbb{C} \setminus i\mathbb{R}$, as follows:

$$\operatorname{sign}(\zeta) = \begin{cases} -1, & \Re(\zeta) < 0, \\ 1, & \Re(\zeta) > 0. \end{cases} \tag{1}$$

This concept extends to matrices through several equivalent formulations, each offering distinct computational advantages. For any matrix $L \in \mathbb{C}^{n\times n}$ with no purely imaginary eigenvalues, the MSF admits the following three principal representations [1] (Chapter 5):

1. The Jordan canonical form representation. If $L = ZJZ^{-1}$, where

$$J = \operatorname{diag}(J_1, J_2),$$

with eigenvalues in the left/right half-planes, respectively; then, the following is obtained:

$$\operatorname{sign}(L) = Z \begin{bmatrix} -I_p & 0 \\ 0 & I_q \end{bmatrix} Z^{-1}, \tag{2}$$

where I_p and I_q are identity matrices of appropriate sizes.

2. The algebraic representation.

$$\text{sign}(L) = L(L^2)^{-1/2}, \tag{3}$$

generalizing the scalar case $\text{sign}(z) = z/(z^2)^{1/2}$.

3. The integral representation.

$$\text{sign}(L) = \frac{2}{\pi} L \int_0^\infty (\theta^2 I + L^2)^{-1} d\theta. \tag{4}$$

These diverse formulations underscore the MSF's mathematical richness, while enabling varied computational approaches tailored to specific matrix structures and applications. Note that the MSF traces its origins to Roberts' seminal 1971 work [2], where it was introduced through the Cauchy integral representation (4). Key developments in its numerical computation include the following:

- 1970s: Initial applications in control theory for solving algebraic Riccati equations [2].
- 1987: Byers' discussions on the algebraic Riccati Equation [3].
- 1991: Kenney and Laub's systematic derivation of Padé iterations and comprehensive analysis of conditioning and stability [4–6].
- 2008: Higham's unified treatment in *Functions of Matrices* [1].
- 2014–2015: Soleymani and colleagues discussed that it is useful to propose new and globally convergent schemes through the developments of iterative solvers [7,8].

Control theory is instrumental in solving algebraic Riccati equations via invariant subspace computations. In quantum physics, the MSF is employed to evaluate the overlap Dirac operator within lattice QCD simulations [9]. It also arises in the analysis of matrix equations, particularly in the study of quadratic matrix equations that appear in stochastic processes [10]. Furthermore, the MSF is a valuable tool in system analysis, where it aids in assessing the stability of power networks and mechanical systems.

The MSF possesses several remarkable properties that underpin its computational utility [1] (Theorem 5.1, page 107), as follows.

Lemma 1. *For $L \in \mathbb{C}^{n\times n}$ with no purely imaginary eigenvalues and $V = \text{sign}(L)$, the following holds:*

1. *V is involutory: $V^2 = I_n$.*
2. *V is diagonalizable with spectrum $\sigma(V) \subseteq \{-1, 1\}$.*
3. *V commutes with L: $VL = LV$.*
4. *Reality preservation: $L \in \mathbb{R}^{n\times n} \Rightarrow V \in \mathbb{R}^{n\times n}$.*
5. *The operators $P_\pm = \frac{1}{2}(I \pm V)$ are projectors onto the invariant subspaces corresponding to the eigenvalues in the right/left half-planes.*

While $\text{sign}(L)$ is a square root of identity, it differs from $\pm I$, unless all eigenvalues of L lie strictly in one half-plane. Moreover, its norm can be arbitrarily big, despite having unit eigenvalues. The MSF exhibits particularly elegant representations for block matrices. Let us now consider the following structured matrix:

$$P = \begin{bmatrix} 0 & L \\ B & 0 \end{bmatrix}, \quad L, B \in \mathbb{C}^{n\times n}, \tag{5}$$

where LB and BL have no negative real eigenvalues. Then, Theorem 5.2 of [1] provides the following:

$$\operatorname{sign}(P) = \begin{bmatrix} 0 & C \\ C^{-1} & 0 \end{bmatrix}, \quad C = L(BL)^{-1/2}. \tag{6}$$

A special case emerges when $B = I$, and

$$\operatorname{sign}\left(\begin{bmatrix} 0 & L \\ I & 0 \end{bmatrix}\right) = \begin{bmatrix} 0 & L^{1/2} \\ L^{-1/2} & 0 \end{bmatrix}. \tag{7}$$

This reveals an intimate connection between the MSF and matrix square roots.

The numerical computation of $\operatorname{sign}(L)$ typically proceeds via two principal approaches. For dense matrices, the Schur decomposition $L = QTQ^*$ with Q unitary and T upper triangular yields the following:

$$\operatorname{sign}(L) = Q \operatorname{sign}(T) Q^*. \tag{8}$$

The triangular $\operatorname{sign}(T)$ can be computed through a recurrence relation on the blocks of T, as detailed in [1] (Section 5.4).

On the other hand, Newton's method provides the following foundational iteration:

$$X_{k+1} = \frac{1}{2}(X_k + X_k^{-1}), \tag{9}$$

with

$$X_0 = L, \tag{10}$$

which converges quadratically under appropriate conditions [1] (Theorem 5.6). The scaled Newton solver for computing the MSF is now provided in Algorithm A1.

The computational complexity of Algorithm A1 is primarily dictated by the matrix inversion performed in each iteration. For an $n \times n$ matrix, the inversion step using LU factorization entails approximately $\frac{2}{3}n^3$ floating-point operations. Additionally, the matrix multiplication and addition steps incur around $2n^3$ operations per iteration, while the computation of the Frobenius norm (denoted in this paper as $\|\cdot\|_F$) requires $O(n^2)$ operations. Consequently, for k iterations, the overall computational cost is approximately $2kn^3$ flops.

The algorithm generally converges within 5 to 20 iterations for well-conditioned matrices, resulting in a practical computational cost ranging from $10n^3$ to $40n^3$ floating-point operations. Termination is guaranteed by the stopping criteria, which are satisfied either when the relative change δ_k falls below a prescribed tolerance, or when further progress is hindered by round-off errors.

More sophisticated Padé iterations generalize this approach [4] as follows:

$$X_{k+1} = X_k q_{\ell m}(I - X_k^2)^{-1} p_{\ell m}(I - X_k^2), \tag{11}$$

where $p_{\ell m}/q_{\ell m}$ are carefully chosen rational approximants.

The motivation for developing a sixth-order scheme is both practical and theoretical. From a theoretical standpoint, high-order iterative methods, such as those of the sixth order, yield faster asymptotic convergence. Specifically, when the initial approximation lies sufficiently close to the true solution, such schemes drastically reduce the number of required iterations compared to low-order methods like Newton's method, which only offers quadratic convergence. This becomes particularly beneficial for problems involving high-precision computation or stiff spectral properties, where slow convergence may result in numerical instability or high computational cost. From a practical perspective,

MSF evaluations often appear in large-scale problems in relation to control theory, as well as in matrix equation solvers, where each iteration involves expensive operations such as matrix inversions or multiplications. Hence, a method that converges in fewer iterations—such as a sixth-order schemes—directly reduces the total computational time despite the slightly increased cost per iteration. Moreover, the proposed scheme aims to combine the robustness of rational approximants with the speed of high-order methods, offering a new balance between accuracy, global convergence, and practical runtime efficiency. This design goal aligns with the direction of recent studies (e.g., [11,12]), while advancing the state of the art by providing a novel solver that is not only sixth-order accurate but also globally convergent and algebraically structured for matrix extension.

Accordingly, the novelty of this work advances the state of the art by introducing a sixth-order iterative scheme that achieves both rapid convergence and numerical robustness. In contrast to recent sixth-order methods derived purely from rational approximants or interpolation theory (e.g., [11,12]), our proposed iteration arises from a carefully constructed four-step scalar solver built on modified divided differences and weight functions, which is then algebraically lifted to the matrix setting. This formulation results in a rational matrix iteration with explicitly optimized polynomial coefficients, ensuring both numerical stability and convergence speed.

The structure of the manuscript is as follows. Section 2 details the derivation of the proposed method and provides its convergence analysis. Section 3 explores the global convergence properties. Section 4 presents extensive numerical validations, while Section 5 concludes the paper with a summary of findings and potential directions for future works.

2. Deriving a New Iteration Solver

2.1. Foundational Equations

The computation of $\text{sign}(L)$ through iterative methods [13,14] requires solving the following fundamental matrix equation:

$$G(X) = X^2 - I = 0, \tag{12}$$

where $X \in \mathbb{C}^{n\times n}$ represents the sought-after matrix sign solution. This nonlinear matrix Equation (12) has its scalar counterpart [15], as follows:

$$g(x) = x^2 - 1 = 0, \tag{13}$$

whose solutions $x = \pm 1$ correspond to the eigenvalues of the MSF.

2.2. Developed Iterative Scheme

Building upon established root-finding techniques [16,17] and recent advances in matrix function computation [18], we propose an enhanced multi-step iterative scheme combining rational approximations with optimized weight functions, as follows:

$$\begin{cases} h_k = x_k - f_k, \\ z_k = x_k - \left(\dfrac{151g(h_k) - 150g(x_k)}{301g(h_k) - 150g(x_k)}\right) f_k, \\ v_k = z_k - \dfrac{g(z_k)}{g[z_k, x_k]}, \\ x_{k+1} = v_k - \dfrac{g(v_k)}{g[v_k, h_k]}, \end{cases} \tag{14}$$

where $f_k = \dfrac{g(x_k)}{g'(x_k)}$ and the divided difference operator is given as follows:

$$g[c_1, c_2] := \frac{g(c_1) - g(c_2)}{c_1 - c_2}.$$

This carefully constructed four-step framework achieves the following two critical objectives:

1. It establishes a new class of iterative methods distinct from traditional Padé approximations.
2. It guarantees computational efficiency while maintaining global convergence properties.

2.3. Convergence Analysis

Theorem 1. *Let $x^* \in D$ be a simple root of the sufficiently smooth function $g : D \subseteq \mathbb{C} \to \mathbb{C}$. For initial estimates x_0 in a sufficiently small neighborhood of x^*, the iteration scheme (14) converges with sixth-order accuracy.*

Proof. Through Taylor series expansion about the root x^*, we derive the error propagation equation, as follows:

$$e_{k+1} = x_{k+1} - x^* = -\frac{1}{150}\mu_2^5 e_k^6 + \left(\frac{23399\mu_2^6}{22500} - \frac{157}{150}\mu_2^4\mu_3\right)e_k^7 + O(e_k^8), \tag{15}$$

where $\mu_k = \frac{1}{k!}\frac{g^{(k)}(x^*)}{g'(x^*)}$ for $k \geq 2$. This process is summarized in the following Mathematica 14.0 piece of code:

```
ClearAll["Global‘*"]
g[e_] :=
 dfa (e^1 + Subscript[\[Mu], 2] e^2 + Subscript[\[Mu], 3] e^3 +
    Subscript[\[Mu], 4] e^4 + Subscript[\[Mu], 5] e^5 +
    Subscript[\[Mu], 6] e^6)
ge = g[e]; g1e = g'[e];
d = e - Series[(ge/g1e), {e, 0, 7}] // FullSimplify;
gd = g[d];
x = e - ((151 gd - 150 ge)/(301 gd - 150 ge)) ge/g1e // FullSimplify;
gx = g[x]; DD01 = (gx - gd)/(x - d);
p = x - gx/DD01 // FullSimplify;
gp = g[p]; DD02 = (gp - ge)/(p - e);
e1 = p - gp/DD02 // FullSimplify
```

To obtain (15), we employ the iterative method (14) and write its Taylor expansions for all the involved terms up to order six or seven. After simiplifications, as made in the program above, the final error Equation (15) is derived. The dominant sixth-order error term confirms the claimed convergence rate. □

2.4. Matrix Iteration Formulation

The scalar iteration (14) naturally extends to the matrix case through the following polynomial/rational implementation:

$$X_{k+1} = X_k(1055I + 5255X_k^2 + 3141X_k^4 + 149X_k^6)[151I + 3159X_k^2 + 5245X_k^4 + 1045X_k^6]^{-1}, \tag{16}$$

initialized with $X_0 = L$. The construction of (16) is outlined in the following Mathematica 14.0 code, where the main iterative solver for the scalar case (14) is symbolically implemented to address a specific nonlinear equation:

```
ClearAll["Global`*"]
g[x_] := x^2 - 1
gt = g[X]; g1t = g'[X];
y = X - gt/g1t // FullSimplify;
gy = g[y];
h = X - ((-150 gt + 151 gy)/(-150 gt + 301 gy)) gt/g1t // FullSimplify;
gh = g[h]; ddo = (y - h)^-1 (gy - gh);
e1 = h - gh/ddo // Simplify;
ddo2 = (e1 - X)^-1 (g[e1] - gt);
e2 = e1 - g[e1]/ddo2 // FullSimplify
```

In this code, the second line defines the target nonlinear equation. The program's output, obtained by substituting (14) into all lines except the second, yields (16). Thus, the use of Mathematica software to derive the formulas here does not only serve as a validation result. Furthermore, if (16) is expressed in its equivalent reciprocal formulation, we obtain the following:

$$X_{k+1} = (151I + 3159X_k^2 + 5245X_k^4 + 1045X_k^6)[X_k(1055I + 5255X_k^2 + 3141X_k^4 + 149X_k^6)]^{-1}, \tag{17}$$

providing computational flexibility depending on matrix conditioning and implementation architecture.

Remark 1. *The coefficients in (16) and (17) result from the meticulous optimization of the weight functions to achieve the balance between convergence rate and numerical stability.*

Theorem 2. *Let $L \in \mathbb{C}^{n\times n}$ be nonsingular with no pure imaginary eigenvalues, and let X_0 be an initial approximation sufficiently close to* $\operatorname{sign}(L)$. *The iterative scheme defined by (17) converges to $V = \operatorname{sign}(L)$ with sixth-order convergence.*

Proof. The proof proceeds through spectral decomposition and error analysis.

1. Spectral Decomposition: Consider the Jordan decomposition $L = ZJZ^{-1}$, where $J = \operatorname{diag}(J_1, J_2, \ldots, J_k)$ contains Jordan blocks. The iteration preserves this structure, as follows:
$$X_k = Z\operatorname{diag}(X_k^{(1)}, X_k^{(2)}, \ldots, X_k^{(k)})Z^{-1}. \tag{18}$$
2. Eigenvalue Analysis: For each eigenvalue ω_k^i of X_k, the iteration maps the following:
$$\omega_{k+1}^i = \frac{151 + 3159(\omega_k^i)^2 + 5245(\omega_k^i)^4 + 1045(\omega_k^i)^6}{\omega_k^i(1055 + 5255(\omega_k^i)^2 + 3141(\omega_k^i)^4 + 149(\omega_k^i)^6)}. \tag{19}$$
The fixed points satisfy $\omega = \operatorname{sign}(\Re(\omega))$, with the following asymptotic behavior:
$$\lim_{k\to\infty}\left|\frac{\omega_{k+1}^i - s_i}{\omega_{k+1}^i + s_i}\right| = 0, \quad s_i = \pm 1. \tag{20}$$
3. Error Propagation: The iteration kernel is defined as follows:
$$R_k = X_k(1055I + 5255X_k^2 + 3141X_k^4 + 149X_k^6). \tag{21}$$

The error evolution is as follows:

$$\begin{aligned} X_{k+1} - V &= [151I + 3159X_k^2 + 5245X_k^4 + 1045X_k^6 - R_k X_k]R_k^{-1} \\ &= (X_k - V)^6[-151I + 149X_k]R_k^{-1}. \end{aligned} \tag{22}$$

4. Convergence Rate: Taking 2-norms yields the following sixth-order convergence:

$$\|X_{k+1} - V\|_2 \leq \|R_k^{-1}\|_2\| - 151I + 149X_k\|_2\|X_k - V\|_2^6. \tag{23}$$

The proof is finished. □

The developed iteration scheme possesses several advantages. Its sixth-order convergence rate enables a rapid progression toward the solution in machine precision. When complemented by suitable scaling strategies [19], the method exhibits robust stability, with the error bound in (23) remaining tightly controlled throughout the computation. A key strength of the algorithm lies in its reliance solely on matrix multiplications and inversions, thereby eliminating the need for computationally intensive eigenvalue evaluations. For large-scale problems, these operations can be further optimized by employing blocked matrix multiplication algorithms, parallelized matrix inversion techniques, and leveraging sparsity where applicable. To further determine the reason why the computational load for the proposed algorithm, which involves matrix power operations and inverse operations, is still not problematic, we mention that the essence of all iterative methods for computing the MSF is based on matrix multiplication, which is not a difficult task in numerical linear algebra. The number of matrix products is also balanced with the convergence rate, as will be observed in Section 4. The only remaining issue may be rooted in the computation of the matrix inverse, which is carried out through numerical linear algebra packages that can be handled quite fast and in a reliable manner for almost all iterative methods for this task.

A potential concern is whether the proposed algorithm, which involves both matrix power and matrix inverse operations, imposes some computational burden. While one might argue that if these operations are negligible, the four-step Newton method, with its eighth-order convergence, could yield a superior performance, this is not the case in practice. Specifically, the four-step Newton scheme requires four matrix inversions per iteration cycle, thereby significantly increasing the computational cost. In effect, such a procedure is equivalent to executing four consecutive Newton iterations (see the results of Newton's method in Section 4). Moreover, employing a four-step Newton scheme offers no conceptual novelty. As previously discussed, any new iterative method for this problem must demonstrate competitiveness with the broader class of Padé-based solvers (of which Newton's method is a member) in terms of algorithmic structure, convergence properties, and total computational time to reach the final solution. The elapsed time measurements presented in Section 4 substantiate this analysis.

3. Global Convergence and Stability

The construction and analysis of attraction basins play a pivotal role when developing iterative schemes for computing the MSF, as they provide critical insights into the global performance of the proposed methods. Beyond merely confirming convergence away from the imaginary axis, these basins offer a visual and quantitative understanding of how the iteration behaves across the complex plane.

A method's practical applicability is strongly tied to the extent of its attraction basin; that is, a larger basin implies that the method can achieve convergence for a wider variety of initial matrices. The existing literature has demonstrated that Newton's method and its higher-order extensions—such as those derived from Padé approximations—exhibit

distinctive attraction basin patterns, each with specific convergence properties that can be graphically assessed to compare performance [20].

More than a tool for proving global convergence, attraction basins allow for a direct and intuitive comparison of different iterative schemes in terms of efficiency. While some methods converge rapidly over most of the complex domain, others may be hampered by regions where they require substantially more iterations [21]. Additionally, regions with smooth and sharply bounded basins often correspond to rapid convergence, while intricate or fractal-like boundaries tend to indicate slow or unstable behavior [22].

Another advantage of attraction basin analysis lies in its capacity to expose the sensitivity of iterative solvers to initial conditions. Certain schemes may experience instability, oscillatory behavior, or outright divergence, particularly when initialized near difficult regions such as the imaginary axis [23]. Furthermore, attraction basins help to identify critical or exceptional points where the convergence rate deteriorates significantly or fails altogether. By scrutinizing these convergence maps, one can diagnose and remedy these issues using techniques such as preconditioning, scaling strategies, or adaptive step-size control.

In this investigation, we emphasize the global convergence behavior of the proposed schemes given in Equations (16) and (17). These solvers are specifically constructed to widen the basin of attraction when applied to the model nonlinear equation $g(x) = x^2 - 1 = 0$. To rigorously assess their performance, we examine their global convergence radii through the lens of attraction basin plots over the domain, as follows:

$$[-4, 4] \times [-4, 4] \subset \mathbb{C}.$$

To this target, the complex plane is discretized into a uniform mesh, and each grid point is treated as an initial guess in the iterative scheme. The convergence behavior is then recorded based on whether the iteration satisfies the following convergence criterion:

$$|g(x_k)| \leq 10^{-2}.$$

Points that fail to satisfy this condition within the prescribed iteration limit are marked in black, indicating divergence.

Figures 1 and 2 illustrate the attraction basins for the proposed methods (16) and (17), alongside four well-established high-order Padé-type solvers. The graphical evidence supports the claim that the newly developed schemes possess considerably broader and more uniformly convergent regions. This not only affirms their global convergence properties but also highlights their practical advantages in terms of computational reliability and robustness.

The iteration formula of Padé [0,4] coming from (11) is defined as follows:

$$x_{k+1} = \frac{128x_k}{-5x_k^8 + 28x_k^6 - 70x_k^4 + 140x_k^2 + 35}.$$

Additionally, the iteration relation for [4,0] coming from (11) is defined as follows:

$$x_{k+1} = \frac{1}{128}\left(35x_k^9 - 180x_k^7 + 378x_k^5 - 420x_k^3 + 315x_k\right).$$

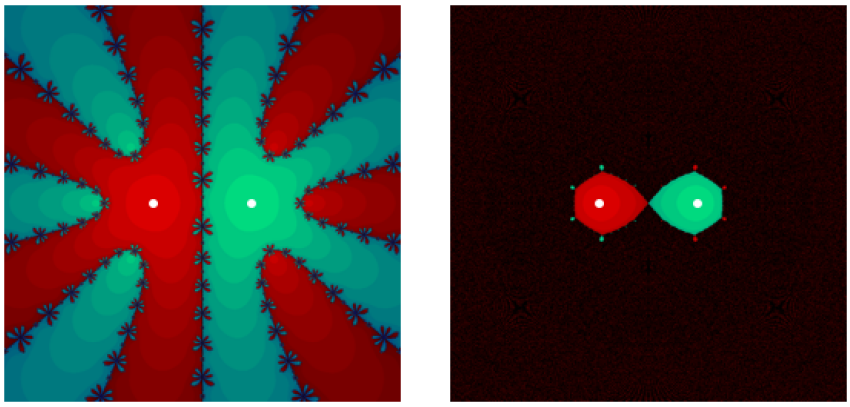


Figure 1. Basins of attractions and fractal convergence behavior for Padé [0,4] in (**left**) and Padé [4,0] in (**right**).

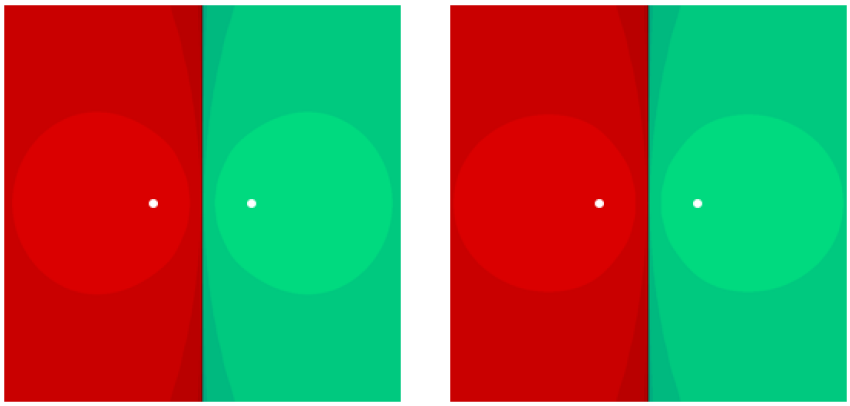


Figure 2. Basins of attractions and fractal convergence behavior for (16) in (**left**) and (17) in (**right**).

Theorem 3. *Let L be an invertible matrix. Next, $\{X_k\}_{k=0}^{\infty}$, which is generated by the iterative scheme defined in Equation (17), with the initial guess $X_0 = L$, is stable asymptotically.*

Proof. We analyze the stability of the iterative solver by examining the behavior of a perturbed iteration. Let $\widetilde{X}_k$ denote the perturbed version of X_k at the k^{th} iteration, modeled as follows:

$$\widetilde{X}_k = X_k + W_k, \tag{24}$$

where W_k represents a small perturbation or error introduced at iteration k. In line with standard first-order error analysis assumptions, we suppose that all higher-order powers of the perturbation can be neglected, i.e., $(W_k)^i \approx 0$ for $i \geq 2$.

Applying this perturbed form to the iterative scheme (17), the next iteration is obtained as follows:

$$\widetilde{X}_{k+1} = \left[11I + 106\,\widetilde{X}_k^2 + 51\,\widetilde{X}_k^4\right]\left[2\widetilde{X}_k\left(27I + 52\,\widetilde{X}_k^2 + 5\,\widetilde{X}_k^4\right)\right]^{-1}. \tag{25}$$

As the iteration approaches convergence, we assume that $X_k \approx \text{sign}(L) = V$. To simplify the inverse term involving the perturbation, we employ the first-order matrix perturbation identity (see [24]), which states that for an invertible matrix F and a small matrix E, the following is true:

$$(F + E)^{-1} \approx F^{-1} - F^{-1}EF^{-1}. \tag{26}$$

We also recall that for the MSF V, we have $V^{-1} = V$ and $V^2 = I$, implying more generally that $V^{2j} = I$ and $V^{2j+1} = V$ for $j \geq 1$. Applying these properties yields the following approximation for the next iterate:

$$\widetilde{X}_{k+1} \approx V + \frac{1}{2}W_k - \frac{1}{2}VW_kV. \tag{27}$$

The perturbation at the next step is thus as follows:

$$W_{k+1} = \widetilde{X}_{k+1} - X_{k+1} \approx \frac{1}{2}W_k - \frac{1}{2}VW_kV. \tag{28}$$

Taking norms on both sides of (28) leads to a bound on the propagated error, as follows:

$$\|W_{k+1}\| \leq \frac{1}{2}\|W_k - VW_kV\|. \tag{29}$$

Now, it is enough to repeat the process for the remaining k iteration in order to obtain the following:

$$\|W_k\| \leq \frac{1}{2}\|W_{k-1} - VW_{k-1}V\|, \tag{30}$$

and in a similar fashion to the following:

$$\|W_{k+1}\| \leq \frac{1}{2}\|W_0 - VW_0V\|. \tag{31}$$

This inequality confirms that the perturbation diminishes under iteration, implying that the sequence $\{X_k\}_{k=0}^{\infty}$ generated by (17) remains bounded and stable, thereby establishing the asymptotic stability of the proposed method. □

4. Computational Performance

In this section, a comprehensive assessment of the developed iterative algorithms is carried out by systematically evaluating their performance across a diverse array of test problems. All numerical experiments have been executed using the computational environment provided by Mathematica 14.0 [25]. The adopted methodology incorporates a variety of numerical diagnostics, including the analysis of convergence profiles and stability properties, in order to ensure a thorough comparison of the solvers.

The benchmark includes several prominent schemes. Specifically, the classical Newton-type iteration given in Equation (9) is labeled as N2. The Halley-type method, originally introduced in [4] and defined by the following iteration:

$$X_{k+1} = X_k(I + 3X_k^2)(3I + X_k^2)^{-1}, \tag{32}$$

is referred to as H3. In addition, two newly proposed methods—denoted as P61 and P62—correspond to the iterative formulations presented in Equations (16) and (17), respectively. The very recent method introduced by Zaka Ullah et al. in [26], which possesses fourth-order convergence, is also included and is labeled as Z4. This method is defined by the following update rule:

$$X_{k+1} = \left(5I + 42X_k^2 + 17X_k^4\right)\left[X_k\left(23I + 38X_k^2 + 3X_k^4\right)\right]^{-1}. \tag{33}$$

For all Newton-like iterations considered in this study, the initial approximation X_0 is selected based on the strategy described in Equation (10). The accuracy of the computed results is measured using the following error metric:

$$E_{k+1} = \|X_{k+1}^2 - I\|_2 \leq \zeta, \tag{34}$$

where ζ is a user-defined tolerance that determines the stopping condition. This criterion ensures that the approximation converges to a matrix whose square is sufficiently close to the identity matrix, thereby verifying the correctness of the computed square root.

Example 1. *To ensure reproducibility, a set of ten real-valued matrices is generated using the Mathematica command* `SeedRandom[12]`*. These matrices are constructed by drawing entries from the numerical range* $[-15, 15]$ *and vary in size from* 100×100 *to* 1000×1000*. Once assembled, the corresponding MSFs of these matrices are computed for further analysis. This framework allows for a detailed comparison among different iterative schemes. All numerical experiments are executed under a convergence threshold specified by the tolerance parameter* $\zeta = 10^{-4}$*.*

In the following discussion, we extract and emphasize the most significant patterns, trends, and implications that emerge from the presented data, thereby shedding light on the practical advantages and numerical stability of the new solvers when applied to both real and complex matrix inputs.

The numerical outcomes related to Example 1 are systematically summarized in Tables 1 and 2. These results offer strong empirical evidence supporting the performance of the developed algorithms. Among the tested methods, P61 consistently exhibits a superior numerical efficiency, requiring the least number of iterations to compute the MSF across all test cases. This advantage is further corroborated by a marked decrease in the average CPU time—measured in seconds—across the ensemble of ten test matrices, each differing in dimension. Consequently, the findings underscore the computational competitiveness and robustness of the proposed method in large-scale scenarios.

Table 1. The efficiency is evaluated through a comparative analysis based on the number of iterations needed to reach convergence in Example 1.

Size	N2	H3	Z4	P61	P62
100×100	15	9	7	6	6
200×200	16	11	7	6	6
300×300	14	9	6	5	5
400×400	17	11	7	6	6
500×500	18	11	8	6	6
600×600	23	14	10	8	8
700×700	17	11	8	8	7
800×800	18	11	8	7	7
900×900	18	12	8	7	7
1000×1000	22	14	10	8	7
Average	17.8	11.3	7.9	6.7	6.5

Table 2. A comparison based on CPU execution time for Example 1.

Size	N2	H3	Z4	P61	P62
100×100	0.01	0.01	0.01	0.00	0.01
200×200	0.05	0.04	0.05	0.04	0.03
300×300	0.13	0.10	0.08	0.08	0.08
400×400	0.26	0.25	0.21	0.18	0.20
500×500	0.48	0.41	0.36	0.31	0.33
600×600	0.92	0.77	0.69	0.62	0.65
700×700	0.98	0.85	0.81	0.89	0.82
800×800	1.42	1.17	1.10	1.09	1.15
900×900	1.88	1.74	1.48	1.45	1.51
1000×1000	3.06	2.68	2.41	2.12	1.89
Average	0.92	0.80	0.72	0.68	0.67

Example 2. *In this experiment, the MSF is computed for a set of ten randomly generated complex matrices. To maintain numerical accuracy throughout the computational process, a convergence tolerance of* $\zeta = 10^{-4}$ *is enforced. The matrix samples are generated using the following Wolfram Mathematica 14.0 code snippet, which ensures the consistency and reproducibility of the test cases:*

```
SeedRandom[12];
numb = 10;
Table[L[n] = RandomComplex[{-15 - 15 I,
      15 + 15 I}, {100 n, 100 n}];, {n, 1, numb}];
```

Each complex matrix is sampled from the rectangular region in the complex plane bounded by $[-15 - 15i, 15 + 15i]$*, with sizes ranging from* 100×100 *to* 1000×1000*.*

A detailed evaluation of the numerical results corresponding to Example 2 is provided in Tables 3 and 4. These tables offer a thorough comparison of the competing iterative schemes in terms of their accuracy and computational performance for MSF evaluation on complex matrices. The data convincingly support the effectiveness of the proposed algorithms. In particular, the method referred to as P61 exhibits a consistent advantage over the other schemes, delivering faster convergence and higher precision. The robustness of this approach is further validated by its reliable performance across a broad spectrum of problem sizes and matrix configurations, demonstrating its potential for tackling complex-valued problems with computational efficiency and stability.

Table 3. The efficiency is evaluated through a comparative analysis based on the number of iterations needed to reach convergence in Example 2.

Size	N2	H3	Z4	P61	P62
100×100	19	12	8	7	7
200×200	18	11	8	7	7
300×300	16	11	7	6	6
400×400	22	14	10	8	7
500×500	19	12	9	7	7
600×600	22	14	10	8	9
700×700	19	12	8	7	7
800×800	26	16	11	9	9
900×900	20	13	9	8	8
1000×1000	20	13	9	8	8
Average	20.1	12.8	8.9	7.5	7.5

Table 4. A comparison based on CPU execution time for Example 2.

Size	N2	H3	Z4	P61	P62
100×100	0.03	0.03	0.02	0.03	0.04
200×200	0.11	0.11	0.11	0.12	0.11
300×300	0.30	0.30	0.27	0.25	0.27
400×400	0.79	0.74	0.69	0.64	0.58
500×500	1.16	1.10	1.10	0.93	1.16
600×600	2.12	2.01	1.83	1.73	2.21
700×700	2.79	2.59	2.21	2.24	2.39
800×800	5.42	4.82	4.34	3.92	4.07
900×900	6.00	5.51	4.76	4.79	4.94
1000×1000	8.68	7.64	6.70	6.75	6.89
Average	2.74	2.48	2.20	2.14	2.26

The numerical outcomes summarized in Tables 1–4 offer a detailed and systematic comparative evaluation of the effectiveness and computational efficiency of the proposed iterative schemes in computing the MSF. These tabulated results serve as empirical evidence supporting the performance claims of the developed methods.

5. Conclusions and Future Perspectives

This paper introduced a sixth-order iterative method for computing the MSF, offering both theoretical innovation and practical benefits. The proposed method, derived from a scalar four-step solver and extended to matrices via a rational formulation, achieves higher convergence rates and improved numerical stability compared to traditional Newton- or Padé-based approaches. The relevance of this technique is underscored by the computational burden of matrix sign evaluations in large-scale applications such as algebraic Riccati equations, as well as the analysis of control systems. By requiring significantly fewer iterations for convergence, our method reduces computational cost without sacrificing accuracy, making it particularly useful in high-precision and high-dimensional problems. We established sixth-order convergence through scalar- and matrix-level error analysis and confirmed the method's performance through numerical experiments. Future work includes the following:

- Development of adaptive-order schemes to optimize performance based on spectral properties.
- Implementation on GPU platforms and distributed systems for large-scale problems.
- Extension of the methodology to other matrix functions such as square roots, sector functions, or logarithms.

Author Contributions: Conceptualization, C.Z. and T.L.; methodology, C.Z. and T.L.; software, C.Z.; validation, B.Z.; formal analysis, B.Z.; investigation, W.R. and T.L.; resources, W.R.; data curation, W.R.; writing and original draft, R.C.; writing and review and editing, R.C.; visualization, R.C. and T.L.; supervision, T.L.; project administration, T.L.; funding acquisition, T.L. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by the Research Project on Graduate Education and Teaching Reform of Hebei Province of China (YJG2024133), the Open Fund Project of Marine Ecological Restoration and Smart Ocean Engineering Research Center of Hebei Province (HBMESO2321), and the Technical Service Project of Eighth Geological Brigade of Hebei Bureau of Geology and Mineral Resources Exploration (KJ2025-037, KJ2025-029, KJ2024-012).

Data Availability Statement: The original contributions presented in this study are included in the article. Further inquiries can be directed to the corresponding author.

Acknowledgments: The authors would like to thank all three referees for their constructive suggestions relating to this work.

Conflicts of Interest: The authors declare no conflicts of interest.

Appendix A. The Algorithm of Newton

This is the algorithm used in Section 1.

Algorithm A1 Scaled Newton Iteration for MSF

Require: $L \in \mathbb{C}^{n\times n}$ with no pure imaginary eigenvalues
Require: tol_cgce, tol_scale > 0 (tolerances)
Ensure: $X = \text{sign}(L)$
$X_0 \leftarrow L$
scale $\leftarrow$ true
for $k = 0, 1, 2, \ldots$ **do**
  $Y_k \leftarrow X_k^{-1}$ ▷ Compute inverse
  if scale $=$ true **then**
    $\mu_k \leftarrow$ Compute scaling factor
  else
    $\mu_k \leftarrow 1$
  end if
  $X_{k+1} \leftarrow \frac{1}{2}(\mu_k X_k + \mu_k^{-1} Y_k)$
  $\delta_{k+1} \leftarrow \|X_{k+1} - X_k\|_F / \|X_{k+1}\|_F$
  if scale $=$ true and $\delta_{k+1} \leq$ tol_scale **then**
    scale $\leftarrow$ false
  end if
  if $\|X_{k+1} - X_k\|_F \leq (\text{tol_cgce}\|X_{k+1}\|_F / \|Y_k\|_F)^{1/2}$ **then**
    break
  end if
  if scale $=$ false and $\delta_{k+1} > \delta_k/2$ **then**
    break ▷ Roundoff dominates
  end if
end for
return X_{k+1}

References

1. Higham, N.J. *Functions of Matrices: Theory and Computation*; Society for Industrial and Applied Mathematics: Philadelphia, PA, USA, 2008.
2. Roberts, J.D. Linear model reduction and solution of the algebraic Riccati equation by use of the sign function. *Int. J. Cont.* **1980**, *32*, 677–687.
3. Byers, R.; Solving the algebraic Riccati equation with the matrix sign function. *Linear Algebra Appl.* **1987**, *85*, 267–279.
4. Kenney, C.S.; Laub, A.J. Rational iterative methods for the matrix sign function. *SIAM J. Matrix Anal. Appl.* **1991**, *12*, 273–291.
5. Iannazzo, B. A family of rational iterations and its application to the computation of the matrix pth root. *SIAM J. Matrix Anal. Appl.* **2009**, *30*, 1445–1462.
6. Gomilko, O.; Greco, F.; Ziętak, K. A Padé family of iterations for the matrix sign function and related problems. *Numer. Lin. Alg. Appl.* **2012**, *19*, 585–605.
7. Soleymani, F.; Stanimirović, P.S.; Shateyi, S.; Haghani, F.K. Approximating the matrix sign function using a novel iterative method. *Abstr. Appl. Anal.* **2014**, *2014*, 105301.
8. Soheili, A.R.; Toutounian, F.; Soleymani, F. A fast convergent numerical method for matrix sign function with application in SDEs. *J. Comput. Appl. Math.* **2015**, *282*, 167–178.
9. Neuberger, H. Exactly Massless Quarks on the Lattice. *Phys. Lett. B* **1998**, *417*, 141–144.
10. Bini, D.; Iannazzo, B.; Meini, B. *Numerical Solution of Algebraic Riccati Equations*; SIAM: Bangkok, Thailand, 2012.
11. Rani, L.; Kansal, M. Numerically stable iterative methods for computing matrix sign function. *Math. Meth. Appl. Sci.* **2023**, *46*, 8596–8617.

12. Kansal, M.; Sharma, V.; Sharma, P. Computation of invariant subspaces associated with certain eigenvalues using an approach based on matrix sign function. *J. Comput. Appl. Math.* **2026**, *472*, 116800.
13. Soleymani, F. An efficient twelfth-order iterative method for finding all the solutions of nonlinear equations. *J. Comput. Methods Sci. Eng.* **2013**, *13*, 309–320.
14. Ogbereyivwe, O.; Izevbizua, O. A three-free-parameter class of power series based iterative method for approximation of nonlinear equations solution. *Iran. J. Numer. Anal. Optim.* **2023**, *13*, 157–169.
15. Soleymani, F.; Stanimirović, P.S.; Stojanović, I. A novel iterative method for polar decomposition and matrix sign function. *Discrete Dyn. Nat. Soc.* **2015**, *2015*, 649423.
16. McNamee, J.M. *Numerical Methods for Roots of Polynomials—Part I*; Elsevier: London, UK, 2007.
17. McNamee, J.M.; Pan, V.Y. *Numerical Methods for Roots of Polynomials—Part II*; Elsevier: London, UK, 2013.
18. Soleymani, F.; Haghani, F.K.; Shateyi, S. Several numerical methods for computing unitary polar factor of a matrix. *Adv. Differ. Equ.* **2016**, *2016*, 4.
19. Rani, L.; Soleymani, F.; Kansal, M.; Kumar Nashine, H. An optimized Chebyshev-Halley type family of multiple solvers: Extensive analysis and applications. *Math. Meth. Appl. Sci.* **2025**, *48*, 7037–7055.
20. Jung, D.; Chun, C. A general approach for improving the Padé iterations for the matrix sign function. *J. Comput. Appl. Math.* **2024**, *436*, 115348.
21. Liu, T.; Zaka Ullah, M.; Alshahrani, K.M.A.; Shateyi, S. From fractal behavior of iteration methods to an efficient solver for the sign of a matrix. *Fractal Fract.* **2023**, *7*, 32.
22. Zainali, N.; Lotfi, T. A globally convergent variant of mid-point method for finding the matrix sign. *Comp. Appl. Math.* **2018**, *37*, 5795–5806.
23. Feng, Y.; Zaka Othman, A. An accelerated iterative method to find the sign of a nonsingular matrix with quartical convergence. *Iran. J. Sci.* **2023**, *47*, 1359–1366.
24. Bhatia, R. *Matrix Analysis*; Graduate Texts in Mathematics; Springer: New York, NY, USA, 1997; Volume 169.
25. Styś, K.; Styś, T. *Lecture Notes in Numerical Analysis with Mathematica*; Bentham eBooks: Sharjah, United Arab Emirates, 2014.
26. Zaka Ullah, M.; Muaysh Alaslani, S.; Othman Mallawi, F.; Ahmad, F.; Shateyi, S.; Asma, M. A fast and efficient Newton-type iterative scheme to find the sign of a matrix. *AIMS Math.* **2023**, *8*, 19264–19274.

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.