Journal of Applied and Numerical Analysis
2025, Vol. 3, pp. 53–69
doi: 10.30970/ana.2025.3.53

An orthogonal projection iterative method and its applications

Neha Ramesh Girhe[1], Hemant Kumar Nashine[1]

[1] *Mathematics Division, School of Advanced Sciences and Languages, VIT Bhopal University, Bhopal-Indore Highway, Kothrikalan, Sehore, Madhya Pradesh, 466114, India*

Анотація. У цьому дослідженні ми вивчаємо ітераційні процеси, що включають ортогональне проєкціювання загальних обернених матриць з застосуваннями до оцінки стану, обробки сигналів та задач конструкцій решіток для обчислень з матрицями. Ми досліджуємо поведінку та властивості збіжності цих ітераційних методів. Виконується аналіз чутливості параметрів для оцінки їх впливу на характеристики збіжності ітераційних методів. Особливу увагу ми приділяємо процесу ортогонального проєкціювання, моніторячи слід і норму Фробеніуса детермінованих та випадкових матриць, що дає змогу отримати інформацію про характеристики збіжності та стійкості методів.

Abstract. In this study, we explore the iterative processes involving the orthogonal projection of generalized inverses with applications to state estimation, signal processing and truss problems for matrix computations. We investigate the behavior and convergence properties of these iterative methods. A sensitivity analysis of the parameters is performed to examine their influence on the convergence characteristics of iterative methods. We specifically examine the orthogonal projection process by monitoring the trace and Frobenius norm of the deterministic and random matrices, providing information on the convergence characteristics and stability of the methods.

1 Introduction

Orthogonal projections and generalized inverses are fundamental ideas in linear algebra, which play a key role in several scientific and engineering applications [5]. This work analyzes the relationship between these ideas, focusing on the orthogonal projection of generalized inverses, which has consequences for both theoretical comprehension and practical computation.

The act of projecting a vector onto a subspace so that the error vector-the difference between the original vector and its projection-is orthogonal to the subspace is known as orthogonal projection. This idea is essential for approximating solutions and reducing dimensionality in fields like computer graphics, signal processing, and optimization. The concept of matrix inversion is extended to non-square and singular matrices via generalized inverses [2], especially the Moore-Penrose inverse. These inverses are crucial for addressing least squares problems, ill-conditioned matrices, and systems of linear equations without unique solutions. For every matrix $\mathcal{U}$, the Moore-Penrose inverse, represented by the symbol $\mathcal{U}^{\dagger}$, is unique and satisfies certain features that extend the idea of the inverse.

The goal of this research is to examine the orthogonal projection in relation to generalized inverses. In particular, we study iterative techniques for calculating the orthogonal projection of

Key words: Generalized inverses; Moore-Penrose inverse; Orthogonal Projections; Frobenius norm.

generalized inverses, which are essential for improving numerical algorithms' accuracy and efficiency. We concentrate on error analysis and the convergence characteristics of these techniques.

The structure of the paper is as follows: The essential mathematical foundation for orthogonal projections and generalized inverses is reviewed in Section 1. The iterative techniques used to calculate these projections are explained in Section 2. Convergence and error boundaries are theoretically analyzed in Section 3. The implementation specifics are covered in Section 4, together with numerical findings that support the suggested approaches. The application portion is finally covered in Section 5, and the work is wrapped up with a summary of the results.

Consider the set $\mathbb{C}^{m\times n}$, which contains all complex $(m\times n)$ matrices, and let $\mathbb{C}_r^{m\times n}$ represent the subset of such matrices that have rank r. For any matrix $\mathcal{U}\in\mathbb{C}_r^{m\times n}$, define I as the identity matrix of the appropriate size, $\mathcal{U}^\dagger$ as the Moore-Penrose inverse, $\mathcal{U}^*$ as the conjugate transpose, $R(\mathcal{U})$ as the range space, $N(\mathcal{U})$ as the null space, rank$(\mathcal{U})$ as the rank of $\mathcal{U}$ and nullity$(\mathcal{U}^*)$ as the nullity of $\mathcal{U}^*$. Penrose [3] demonstrated that $\mathcal{U}\mathcal{U}^\dagger$ and $\mathcal{U}^\dagger\mathcal{U}$ are Hermitian idempotent matrices, often referred to as the orthogonal projections related to $\mathcal{U}$. He further proved that $\mathcal{U}\mathcal{U}^\dagger$ acts as the projection onto the $R(\mathcal{U})$ along the $N(\mathcal{U}^*)$, while $\mathcal{U}^\dagger\mathcal{U}$ projects onto the $R(\mathcal{U}^*)$ along the $N(\mathcal{U})$.

For $\mathcal{U}\in\mathbb{C}_r^{n\times n}$, we denote its eigenvalues by

$$\lambda_1(\mathcal{U}\mathcal{U}^*)\geq\ldots>\lambda_r(\mathcal{U}\mathcal{U}^*)>\lambda_{r+1}(\mathcal{U}\mathcal{U}^*)=\ldots\lambda_n(\mathcal{U}\mathcal{U}^*)=0. \tag{1.1}$$

The Moore-Penrose inverse has been extensively examined in various studies [1,4,6,8,9,12,16,17] that led to the development and analysis of numerous direct and iterative methods for their convergence properties and error estimates. It stands as the unique matrix $\mathcal{T}$ that satisfies the four Penrose equations

$$\mathcal{U}\mathcal{T}\mathcal{U}=\mathcal{U},$$

$$\mathcal{T}\mathcal{U}\mathcal{T}=\mathcal{T}, \tag{1.2}$$

$$(\mathcal{U}\mathcal{T})^*=\mathcal{U}\mathcal{T},$$

$$(\mathcal{T}\mathcal{U})^*=\mathcal{T}\mathcal{U}. \tag{1.3}$$

A series of iterative methods to approximate the inverses of square matrices was first introduced in [13]. These methods from [13] can also be utilized to compute an approximate inner inverse of a matrix for a given approximation

$$\mathcal{T}_{q+1}^{\ell} \;=\; \mathcal{T}_q^{\ell}[\ell I-\frac{\ell(\ell-1)}{2}\mathcal{U}\mathcal{T}_q^{(\ell)}+\ldots+(-1)^{\ell-1}(\mathcal{U}\mathcal{T}_q^{(\ell)})^{\ell-1}], q\in\mathbb{W},\ell\in\mathbb{N},$$

where $\mathbb{W}$ is the set of whole numbers. When $\ell=2$, the well-known Newton–Schulz method, which converges quadratically, is obtained [2]

$$\mathcal{T}_{q+1}=\mathcal{T}_q[2I-\mathcal{U}\mathcal{T}_q],\quad q\in\mathbb{W}. \tag{1.4}$$

Starting with $\mathcal{T}_0=\gamma\mathcal{U}^*$ generates a sequence $\mathcal{T}_q$ converging to $\mathcal{U}^\dagger$ if γ satisfies

$$0<\gamma<\frac{2}{\lambda_1(\mathcal{U}\mathcal{U}^*)}.$$

In [2], its aspect to the iterative method

$$\mathcal{T}_{q+1} \;=\; \mathcal{T}_q+\gamma(I-\mathcal{T}_q\mathcal{U})\mathcal{U}^*,$$

for $\mathcal{T}_0 = \gamma\mathcal{U}^*$ is exhibited to be $\mathcal{T}_q = \mathcal{T}_{2^q-1}$.

Using Penrose equations (1.2) and (1.3) another iterative method for computing the Moore-Penrose inverse given by Petković et al. [16] starting with $\mathcal{T}_0 = \gamma\mathcal{U}^*$ is

$$\mathcal{T}_{q+1} = (I - \alpha\mathcal{T}_q\mathcal{U})^*\mathcal{T}_q + \alpha\mathcal{T}_q, \quad q \in \mathbb{W}, \tag{1.5}$$

where α is a suitable real number. The error matrix $\mathcal{E}_q = \mathcal{T}_q - \mathcal{L}$ can be used to evaluate the convergence properties of the examined iterative approach if $\mathcal{L}$ is the desired limit matrix and $\mathcal{T}_q$ is the q-th estimate of $\mathcal{U}$. If an iterative process can be expressed using a simple matrix formula, then $\mathcal{E}_{q+1}$ can be represented as the sum of several terms:

- Zero-order term is a matrix independent of $\mathcal{E}_q$.
- At least first order term involving $\mathcal{E}_q$, or its conjugate transpose $\mathcal{E}_q^*$ appearing only once.
- Higher-order terms in which $\mathcal{E}_q$ or $\mathcal{E}_q^*$ appears at least twice.

In all applicable algorithms, the zero-order term is 0. Consequently, the first-order term determines the terminal convergence behavior.

For $q = 3$ in (1.4), Li et al. [11] examined the third-order method, which is called the Chebyshev method,

$$\mathcal{T}_{q+1} = \mathcal{T}_q(3I - \mathcal{U}\mathcal{T}_q(3I - \mathcal{U}\mathcal{T}_q)), \quad q \in \mathbb{W}. \tag{1.6}$$

Using Penrose equation (1.2) given by $\mathcal{T}\mathcal{U}\mathcal{T} = \mathcal{T}$, Srivastava and Gupta [19] assisted to the following third order extension of (1.6)

$$\mathcal{T}_{q+1} = \mathcal{T}_q + \alpha\mathcal{T}_q(2I - 3\mathcal{U}\mathcal{T}_q + (\mathcal{U}\mathcal{T}_q)^2), \quad q \in \mathbb{W}.$$

If we put $\alpha = 1$, we get back (1.6). Nashine et al. [15] proposed a new iteration using Penrose equation (1.2)

$$\mathcal{T}_{q+1} = \mathcal{T}_q((1 + \alpha + 2\beta)I - (\alpha + 3\beta)\mathcal{U}\mathcal{T}_q + \beta(\mathcal{U}\mathcal{T}_q)^2), \quad q \in \mathbb{W}. \tag{1.7}$$

Also, authors proposed some lemmas and discussed the following convergence analysis of the iterative scheme with the starting value $\mathcal{T}_0 = \gamma\mathcal{U}^*$ and showed that it converges to the Moore-Penrose inverse $\mathcal{T} = \mathcal{U}^\dagger$.

Theorem 1.1. *[15] Let $\boldsymbol{O} \neq \mathcal{U} \in \mathbb{C}^{m\times n}$, $\mathcal{T} = \mathcal{U}^\dagger$, the initial approximation $\mathcal{T}_0 = \gamma\mathcal{U}^*$, for arbitary real number γ be such that the residual $R_0 = (\mathcal{T}_0 - \mathcal{T})\mathcal{U}$ satisfy $\|R_0\| < 1$. The sequence $\mathcal{T}_q$ generated by the iteration for $\alpha \geq 0$ starting with $\mathcal{T}_0 = \gamma\mathcal{U}^*$ converges to the Moore-Penrose inverse $\mathcal{U}^\dagger$. It exhibits linear convergence for $\alpha + \beta \neq 1$, quadratic convergence for $\alpha + \beta = 1$ and the third order convergence for $\alpha = 0$, and $\beta = 1$. The first, second, and third order error terms are given by*

$$\mathcal{E}_1 = (1 - \alpha - \beta)\mathcal{E}_q, \mathcal{E}_2 = \alpha\mathcal{E}_q\mathcal{U}\mathcal{E}_q \text{ and } \mathcal{E}_3 = \beta\mathcal{E}_q(\mathcal{U}\mathcal{E}_q)^2,$$

where $\mathcal{E}_q = \mathcal{T}_q - \mathcal{U}^\dagger$ denotes the error matrix.

The iteration (1.7) reduces to (1.6) for $\alpha = 0$ and $\beta = 1$. It simplifies to Newton-Schultz iteration (1.4) when $\beta = 0$ and $\alpha = 1$ are added to the (1.7) iteration. Since calculating orthogonal projections is an extremely challenging task, not much work has been done on the subject. It is crucial for solving a variety of linear equation systems, eigenvalue problems, linear least square problems, and figuring out the rank and nullity of rectangular matrices, among other practical areas of nature science. According to Golub and Kahan [7], the accurate rank determination of $\mathcal{U}$ is a critical

component of these algorithms, and it is even more significant in the direct techniques for computing $\mathcal{U}$.

Based on $\mathcal{T}_q$ derived from (1.2), Ben-Isreal and Cohen [3] created the following iterative technique for computing $\mathcal{U}\mathcal{U}^\dagger$. For $q \in \mathbb{W}$, define

$$\textbf{BI:} \begin{cases} \mathcal{Z}_0 = \gamma \mathcal{U}\mathcal{U}^*, \\ \mathcal{Z}_{q+1} = \mathcal{Z}_q - (\mathcal{Z}_q)^2, \end{cases} \tag{1.8}$$

where $\mathcal{Z}_q = \mathcal{U}\mathcal{T}_q$ and γ satisfies

$$0 < \gamma < \frac{2}{\lambda_1(\mathcal{U}^*\mathcal{U})}.$$

Further, using iterative method (1.5), Srivastava and Gupta [18] developed following iterative method for the computing $\mathcal{U}\mathcal{U}^\dagger$

$$\textbf{SD:} \begin{cases} \mathcal{Z}_0 = \gamma \mathcal{U}\mathcal{U}^*, \\ \mathcal{Z}_{q+1} = (1+\alpha)\mathcal{Z}_q - \alpha(\mathcal{Z}_q)^2, \end{cases} \tag{1.9}$$

for some appropriate real number α.

Srivastava and Gupta [18] established following lemma which is crucial to calculate the value of γ.

Lemma 1.2. *Let the eigenvalues of matrix $\mathcal{U}\mathcal{U}^*$ satisfy (1.1). The $\rho(\gamma\mathcal{U}\mathcal{U}^* - \mathcal{Z}) < 1$, where $\rho(\mathcal{U}) = \max\{|\lambda_1(\mathcal{U})|, ..., |\lambda_n(\mathcal{U})|\}$ is satisfied if*

$$\max_{i \le i \le r} \|1 - \gamma\lambda_i(\mathcal{U}\mathcal{U}^*)\| < 1. \tag{1.10}$$

Ben-Isreal and Cohen [3] and Srivastava and Gupta [18] both have demonstrated that the trace$(\mathcal{Z}_q)$ sequence increases monotonically and converges to the rank $(\mathcal{U})$, while the trace $(I - \mathcal{Z}_q)$ sequence decreases monotonically and converges to the nullity $(\mathcal{U}^*)$.

Motivated from above discussions, in this work, we are discovering a better alternative of the quadratic convergent iterative scheme to examine the orthogonal projection in relation to generalized inverses.

2 An iterative method for $\mathcal{U}\mathcal{U}^\dagger$

To determine the new iterative method for computing $\mathcal{U}\mathcal{U}^\dagger$, we begin by pre-multiplying equation (1.7) by $\mathcal{U}$ and taking $\mathcal{Z}_q = \mathcal{U}\mathcal{T}_q$, we have

$$\textbf{NH:} \begin{cases} \mathcal{Z}_0 = \gamma \mathcal{U}\mathcal{U}^*, \\ \mathcal{U}\mathcal{T}_{q+1} = \mathcal{U}\mathcal{T}_q(aI + b\,\mathcal{U}\mathcal{T}_q + c(\mathcal{U}\mathcal{T}_q)^2), \\ \mathcal{Z}_{q+1} = \mathcal{Z}_q(aI + b\mathcal{Z}_q + c(\mathcal{Z}_q)^2), \end{cases} \tag{2.1}$$

where $a = 1 + \alpha + 2\beta$, $b = -(\alpha + 3\beta)$, $c = \beta$ such that $a + b + c = 1$.

The following lemmas will be crusial in establishing the convergence analysis of the above iterative method (2.1) with $\mathcal{Z}_0 = \gamma\mathcal{U}\mathcal{U}^*$.

Lemma 2.1. *$\mathcal{U}\mathcal{U}^\dagger\mathcal{Z}_q = \mathcal{Z}_q$, for all $q \ge 0$.*

Proof. For $q = 0$, using mathematical induction, we obtain

$$\mathcal{U}\mathcal{U}^\dagger\mathcal{Z}_0 = \gamma\mathcal{U}\mathcal{U}^\dagger\mathcal{U}\mathcal{U}^* = \gamma\mathcal{U}\mathcal{U}^* = \mathcal{Z}_0.$$

Let's assume that $\mathcal{U}\mathcal{U}^\dagger\mathcal{Z}_q = \mathcal{Z}_q$ holds for some $q > 0$. It is simple to demonstrate that it is true for $q+1$ as well since,

$$\mathcal{U}\mathcal{U}^\dagger\mathcal{Z}_{q+1} = \mathcal{U}\mathcal{U}^\dagger(\mathcal{Z}_q(aI + b\mathcal{Z}_q + c(\mathcal{Z}_q)^2)) = \mathcal{Z}_{q+1}.$$

□

Lemma 2.2. *$\mathcal{Z}_q\mathcal{U}\mathcal{U}^\dagger = \mathcal{Z}_q$, for all $q \geq 0$.*

Proof. For $q = 0$, using mathematical induction, we obtain

$$\mathcal{Z}_0\mathcal{U}\mathcal{U}^\dagger = \gamma\mathcal{U}\mathcal{U}^*\mathcal{U}\mathcal{U}^\dagger = \gamma\mathcal{U}\mathcal{U}^*(\mathcal{U}\mathcal{U}^\dagger)^* = \gamma\mathcal{U}\mathcal{U}^*(\mathcal{U}^\dagger)^*\mathcal{U}^* = \gamma\mathcal{U}(\mathcal{U}\mathcal{U}^\dagger\mathcal{U})^* = \gamma\mathcal{U}\mathcal{U}^* = \mathcal{Z}_0.$$

Let's assume that $\mathcal{Z}_q\mathcal{U}\mathcal{U}^\dagger = \mathcal{Z}_q$ holds for some $q > 0$. It is simple to demonstrate that this also applies to $q+1$, as

$$\mathcal{Z}_{q+1}\mathcal{U}\mathcal{U}^\dagger = \mathcal{Z}_q(aI + b\mathcal{Z}_q + c(\mathcal{Z}_q)^2)\mathcal{U}\mathcal{U}^\dagger = \mathcal{Z}_{q+1}.$$

□

3 Convergence analysis

In this part, we will analyze the convergence of the iterative approach (2.1) with $\mathcal{Z}_0 = \gamma\mathcal{U}\mathcal{U}^\dagger$ for computing $\mathcal{U}\mathcal{U}^\dagger$.

Theorem 3.1. *The iterative method (2.1) with $\mathcal{Z}_0 = \gamma\mathcal{U}\mathcal{U}^*$ converges to $\mathcal{Z} = \mathcal{U}\mathcal{U}^\dagger$ under the assumption*

$$\|\gamma\mathcal{U}\mathcal{U}^* - \mathcal{Z}\| < 1, \quad 0 < \gamma \leq 1.$$

It exhibits first-order convergence for $\alpha + \beta \neq 1$, second-order convergence when $\alpha + \beta = 1$ and third order convergence when $\alpha = 0$ and $\beta = 1$. The first, second and the third-order error terms are:

$$error_1 = (1 - \alpha - \beta)E_q,\ error_2 = -\alpha E_q\mathcal{U}E_q \text{ and } error_3 = \beta E_q(\mathcal{U}E_q)^2,$$

where $E_q = \mathcal{Z}_q - \mathcal{Z}$ depicts the error matrix.

Proof. We shall first prove that

$$\begin{aligned}
\mathcal{Z}_{q+1} - \mathcal{Z} &= \mathcal{Z}_q\left(aI + b\mathcal{Z}_q + c(\mathcal{Z}_q)^2\right) - \mathcal{U}\mathcal{U}^\dagger \\
&= a\mathcal{Z}_q + b(\mathcal{Z}_q)^2 + c(\mathcal{Z}_q)^3 - \mathcal{U}\mathcal{U}^\dagger \\
&= a\mathcal{Z}_q - \mathcal{Z} + b(\mathcal{Z}_q)^2 + c(\mathcal{Z}_q)^3 \\
&= (1 + \alpha + 2\beta)\mathcal{Z}_q - \mathcal{Z} - (\alpha + 3\beta)(\mathcal{Z}_q)^2 + \beta(\mathcal{Z}_q)^3 \\
&= (\mathcal{Z}_q - \mathcal{Z}) + \alpha(\mathcal{Z}_q - (\mathcal{Z}_q)^2) + \beta(2\mathcal{Z}_q - 3(\mathcal{Z}_q)^2 + (\mathcal{Z}_q)^3) \\
&= (\mathcal{Z}_q - \mathcal{Z}) - \alpha\left[(\mathcal{Z}_q - \mathcal{Z})^2 + (\mathcal{Z}_q - \mathcal{Z})\right] + \beta\left[(\mathcal{Z}_q - \mathcal{Z})^3 - (\mathcal{Z}_q - \mathcal{Z})\right].
\end{aligned}$$

Thus, the sequence of residual matrices defined by $R_q = \mathcal{Z}_q - \mathcal{Z}$ satisfies the following recurrence relation

$$\begin{aligned}
R_{q+1} &= R_q - \alpha[(R_q)^2 + R_q] + \beta[(R_q)^3 - R_q] \\
&= (1 - \alpha - \beta)R_q - \alpha(R_q)^2 + \beta(R_q)^3
\end{aligned}$$

from this we can see that it has first order of convergence for $\alpha + \beta \neq 1$, second order of convergence for $\alpha + \beta = 1$ and third order of convergence for $\alpha = 0$ and $\beta = 1$. The sequence of error matrices E_q satisfies

$$\begin{aligned}
E_{q+1} &= E_q - \alpha((E_q)^2 + E_q) + \beta((E_q)^3 - (E_q)), \\
E_{q+1} &= (1 - \alpha - \beta)E_q - \alpha(E_q)^2 + \beta(E_q)^3,
\end{aligned}$$

$$\begin{aligned} error_1 &= (1-\alpha-\beta)E_q, \\ error_2 &= -\alpha E_q(\mathcal{U}E_q) \\ \text{and} \quad error_3 &= \beta E_q(\mathcal{U}E_q)^2. \end{aligned}$$

Clearly, $error_1$ vanishes if and only if $\alpha+\beta=1$, while $error_2$ is always vanishing when $\alpha=0$ and $error_3$ is always vanishing for $\beta=0$. □

4 Numerical illustrations

We will give some examples in this section to show how effective our work is. MATHEMATICA 11 and an Intel Core 6 Duo CPU operating at 3.30 GHz are used for all of these examples. The orthogonal projections are calculated using a different approach, and the results are contrasted with those from the (2.1) method. A variety of tables show performance metrics for the computation of orthogonal projection in terms of CPU time(CPU time is the total time taken to run the iterations.), iterations, and error bounds ($\|\mathcal{Z}_q - \mathcal{Z}_{q-1}\|$) under the Frobenius norm. The horizontal axis indicates the number of iterations, while the vertical axis indicates the sequence. The figures are displayed for trace($\mathcal{Z}_q$) and trace($I-\mathcal{Z}_q$), where $\mathcal{Z}_q$, $q=0,1,2\ldots$ represents the q-th iterate obtained by our approach. To illustrate the efficiency of the iterative method for $\mathcal{U}\in\mathbb{C}^{m\times n}$, we consider $\mathcal{Z}_0=\gamma\mathcal{U}\mathcal{U}^*$, where $\gamma=\frac{1}{2\lambda_1(\mathcal{U}\mathcal{U}^*)}$. The stopping criterion is $\|\mathcal{Z}_q-\mathcal{Z}_{q-1}\|<\epsilon$, $\epsilon=10^{-3}$.

Example 4.1. [19]. Consider the matrix $\mathcal{U}$ of order 5×4 of $rank(\mathcal{U})=4$ given by

$$\mathcal{U}=\begin{pmatrix} 0.2794 & 0.1676 & 0.0645 & 0.2326 \\ 0.0065 & 0.2365 & 0.2274 & 0.1261 \\ 0.2271 & 0.1430 & 0.1009 & 0.2867 \\ 0.1265 & 0,1015 & 0.1806 & 0.2846 \\ ,0.2773 & 0.0632 & 0.0503 & 0.1979 \end{pmatrix}.$$

Consider $\gamma=\frac{1}{2\lambda_1(\mathcal{U}\mathcal{U}^*)}=0.8127$ which satisfy the condition (1.10) of Lemma 1.2.

The iteration-generated sequence of iterates $\mathcal{Z}_q$ that converge to the orthogonal projection $\mathcal{U}\mathcal{U}^\dagger$ is given by

$$\mathcal{U}\mathcal{U}^\dagger=\begin{pmatrix} 0.638154 & 0.0855455 & 0.378436 & -0.234353 & 0.159562 \\ 0.0855455 & 0.979776 & -0.0894678 & 0.0554044 & -0.0377227 \\ 0.378436 & -0.0894678 & 0.604213 & 0.245068 & -0.166878 \\ -0.234353 & 0.0554044 & 0.245098 & 0.848219 & 0,103342 \\ 0.159562 & -0.0377227 & -0.166878 & 0.103342 & 0.929639 \end{pmatrix}.$$

From the Table 4.1, it is clear that number of iterations, CPU time and the error bounds of NH method (2.1) is better than the SD method (1.9) and BI method (1.8) for computing orthogonal projection. The Figure 4.1 and Figure 4.2 plot the trace ($\mathcal{Z}_q$) and trace($I-\mathcal{Z}_q$) against the number of iterations. As anticipated, the sequence trace($\mathcal{Z}_q$) is monotonic increasing and converges to the rank ($\mathcal{U}$), while the sequence trace($I-\mathcal{Z}_q$) is a monotonic decreasing and converges to the nullity ($\mathcal{U}^*$).

Table 4.1. Comparison of Methods for Different Values of α,β for example 4.1

Method	Parameters	Number of Iterations	CPU Time	Error Bounds
BI	-	19	2.074×10^{-2}	5.933×10^{-5}
SD	$\alpha=0.7$	20	4.446×10^{-4}	3.059×10^{-4}
NH	$\beta=0.3$	**12**	$\mathbf{2.840\times10^{-4}}$	$\mathbf{9.292\times10^{-6}}$

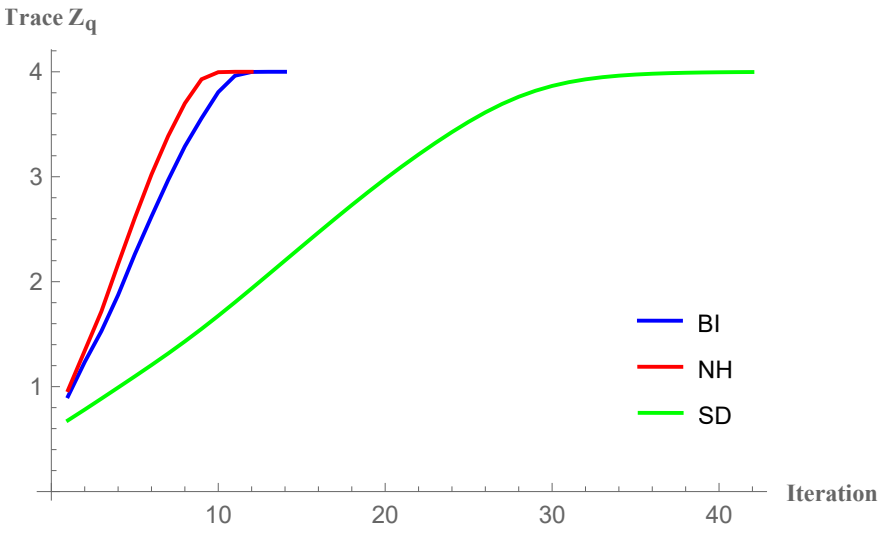

Fig. 4.1: Trace of $\mathcal{Z}_q$ for example 4.1

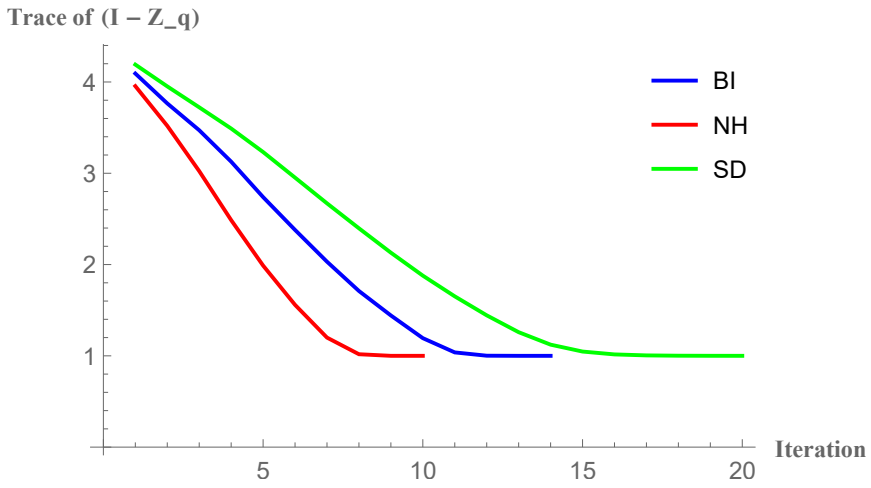

Fig. 4.2: Trace of $I - \mathcal{Z}_q$ for example 4.1

Example 4.2. Consider a randomly generated real matrix of size 30×30, with elements drawn from the interval $[-0.2, 0.2]$, and where $\text{rank}(\mathcal{U}) = 30$. Consider $\gamma = \frac{1}{2\lambda_1(\mathcal{U}\mathcal{U}^*)} = 0.8127$ which satisfies the condition (1.10) of Lemma 1.2. The CPU time, number of iterations, and the error bounds for NH method (2.1) and the SD method (1.9) and the BI method (1.8) for assessing the orthogonal projection are noted in Table 4.2. Figure 4.3 and Figure 4.4 illustrate trace $(\mathcal{Z}_q)$ and trace $(I - \mathcal{Z}_q)$ plotted against number of iterations. As expected, the sequence of trace$(\mathcal{Z}_q)$ monotonically increases, converging to the rank $(\mathcal{U})$, while the sequence trace$(I - \mathcal{Z}_q)$ monotonically decreases, converging to the nullity $(\mathcal{U}^*)$.

We can see from the Table 4.2 that the SD method (1.9) and BI method (1.8) are more expensive and needs more iteration than the NH method (2.1) to attains the stopping criteria.

Additionally, Figure 4.5, Figure 4.6 and Figure 4.7 demonstrate the behavior of NH method (2.1) for number of iterations, CPU time and error bounds for different values of α and β respectively. It is also observed that the error bounds for $(\alpha, \beta) = (0.3, 0.7), (0.5, 0.5), (0.8, 0.2), (0.9, 0.1)$ is less for NH method compared to BI method and SD method.

Table 4.2: Comparison of Methods for Different Values of α, β for example 4.2

Method	Parameters	Number of Iterations	CPU Time	Error Bounds
BI	-	19	2.074×10^{-2}	5.933×10^{-5}
SD	$\alpha = 0.01016$	1215	1.204×10^{0}	9.947×10^{-4}
NH	$\beta = 0.9$	14	0.841×10^{-2}	1.100×10^{-4}
SD	$\alpha = 0.1$	152	1.794×10^{0}	9.540×10^{-4}
NH	$\beta = 0.9$	13	0.852×10^{-2}	5.700×10^{-4}
SD	$\alpha = 0.2$	81	8.154×10^{-2}	9.460×10^{-4}
NH	$\beta = 0.8$	13	0.790×10^{-2}	7.200×10^{-4}
SD	$\alpha = 0.3$	57	7.64×10^{-2}	7.034×10^{-4}
NH	$\beta = 0.7$	**14**	1.197×10^{-2}	0.190×10^{-4}
SD	$\alpha = 0.4$	44	7.018×10^{-2}	6.826×10^{-4}
NH	$\beta = 0.6$	14	1.761×10^{-2}	7.100×10^{-4}
SD	$\alpha = 0.5$	36	3.706×10^{-2}	6.591×10^{-4}
NH	$\beta = 0.5$	**15**	5.492×10^{-2}	0.425×10^{-4}
SD	$\alpha = 0.6$	31	3.850×10^{-2}	4.141×10^{-4}
NH	$\beta = 0.4$	16	1.603×10^{-2}	7.133×10^{-4}
SD	$\alpha = 0.7$	27	2.804×10^{-2}	3.434×10^{-4}
NH	$\beta = 0.3$	16	2.137×10^{-2}	3.500×10^{-4}
SD	$\alpha = 0.8$	24	2.866×10^{-2}	2.284×10^{-4}
NH	$\beta = 0.2$	**17**	1.674×10^{-2}	0.900×10^{-4}
SD	$\alpha = 0.9$	21	2.893×10^{-2}	4.251×10^{-4}
NH	$\beta = 0.1$	**18**	1.825×10^{-2}	0.450×10^{-4}
SD	$\alpha = 1$	19	2.338×10^{-2}	0.594×10^{-4}
NH	$\beta = 0$	19	1.862×10^{-2}	0.598×10^{-4}

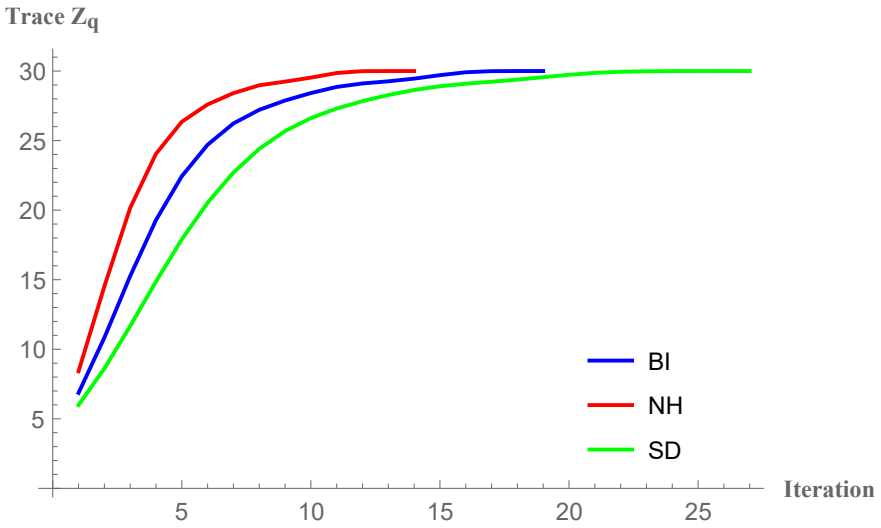

Fig. 4.3: Trace of $\mathcal{Z}_q$ for example 4.2

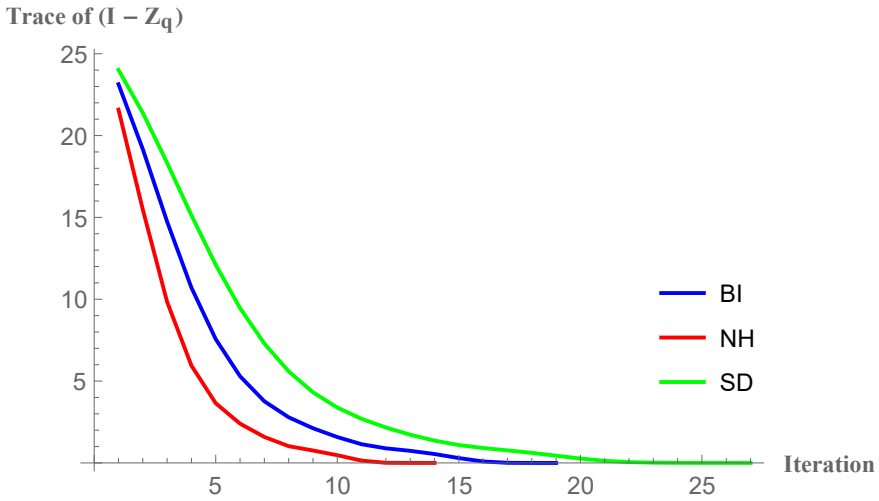

Fig. 4.4: Trace of $I - \mathcal{Z}_q$ for example 4.2

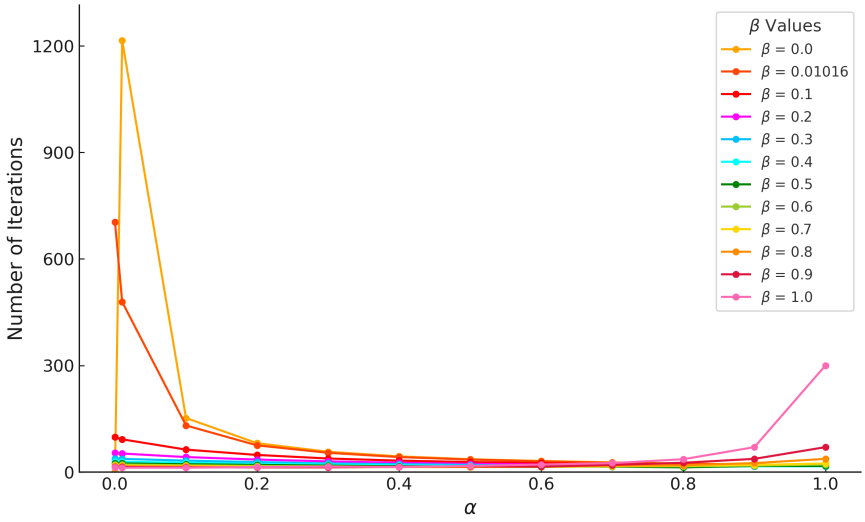

Fig. 4.5: Parameters Impact of NH Method on Number of Iterations

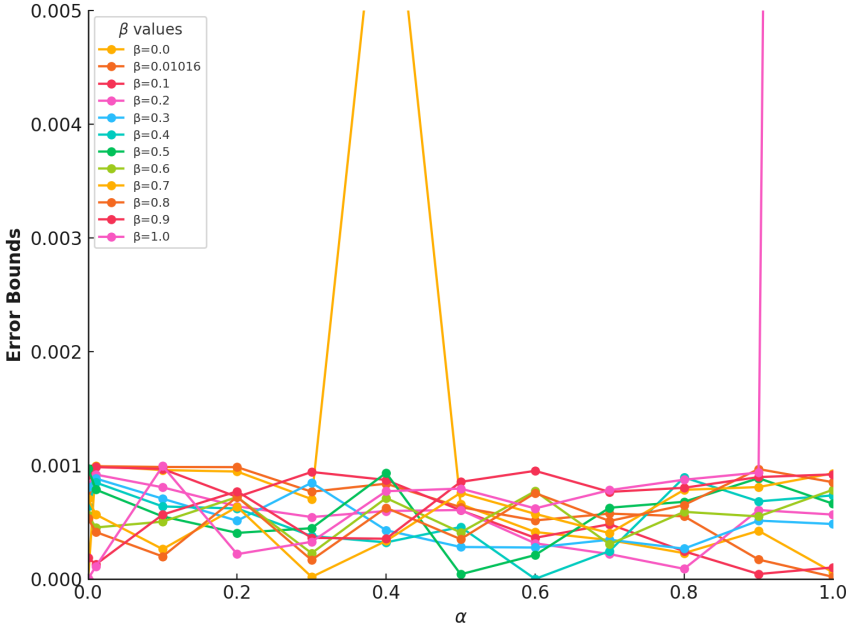

Fig. 4.7: Parameters Impact of NH Method on Error Bounds

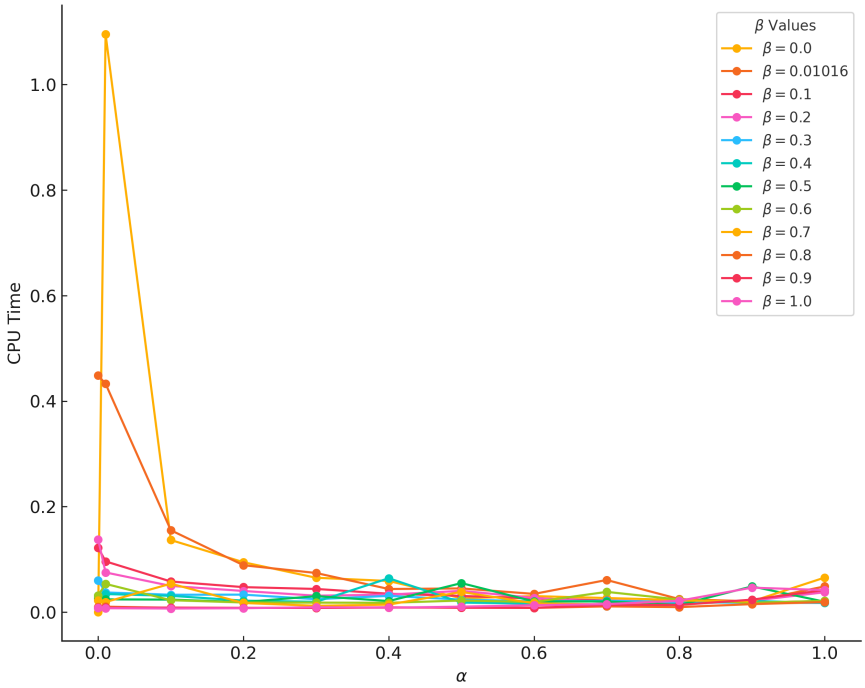

Fig. 4.6: Parameters Impact of NH Method on CPU Time

Example 4.3. Consider a randomly generated real matrix of size 100×101, with element drawn from the interval $[-0.1, 0.1]$ and where $rank(\mathcal{U}) = 100$. Consider $\gamma = \frac{1}{2\lambda_1(\mathcal{U}\mathcal{U}^*)} = 0.8127$ which satisfies the condition (1.10) of Lemma 1.2. The CPU time, number of iterations, and the error bounds of NH method (2.1), along with the SD method (1.9) and BI method (1.8) for assessing the orthogonal projection, are noted in Table 4.3. Figure 4.8 and Figure 4.9 illustrate trace($\mathcal{Z}_q$) and trace($I-\mathcal{Z}_q$) plotted against number of iterations. As expected, the sequence trace($\mathcal{Z}_q$) monotonically increases, converging to the rank($\mathcal{U}$), while the sequence trace($I - \mathcal{Z}_q$) is a monotonically decreases, converging to the nullity($\mathcal{U}^*$).

We can see from the Table 4.3 that the SD method (1.9) and BI method (1.8) are more expensive and needs more iteration than the NH method (2.1) to attains the stopping criteria. It is also observed that the error bound for $\alpha = 0.9$, $\beta = 0.1$ is less for NH method compared to BI method and SD method.

Table 4.3: Comparison of Methods for Different Values of α, β for example 4.3

Method	Parameters	Number of Iterations	CPU Time	Error Bounds
BI	-	20	1.145×10^{-2}	4.1197×10^{-6}
SD	$\alpha = 0.01016$	1266	0.4346	9.954×10^{-4}
NH	$\beta = 0.9$	15	1.037×10^{-2}	1.289×10^{-4}
SD	$\alpha = 0.1$	157	5.741×10^{-2}	9.612×10^{-4}
NH	$\beta = 0.9$	13	7.140×10^{-3}	5.671×10^{-4}
SD	$\alpha = 0.2$	84	2.245×10^{-2}	8.602×10^{-4}
NH	$\beta = 0.8$	14	6.212×10^{-3}	1.075×10^{-5}
SD	$\alpha = 0.3$	58	1.878×10^{-2}	9.209×10^{-4}
NH	$\beta = 0.7$	14	7.256×10^{-3}	4.918×10^{-4}
SD	$\alpha = 0.4$	45	1.325×10^{-2}	8.191×10^{-4}
NH	$\beta = 0.6$	15	8.240×10^{-3}	1.933×10^{-5}
SD	$\alpha = 0.5$	37	1.323×10^{-2}	7.161×10^{-4}

Method	Parameters	Number of Iterations	CPU Time	Error Bounds
NH	$\beta = 0.5$	15	1.883×10^{-2}	9.508×10^{-4}
SD	$\alpha = 0.6$	31	1.088×10^{-2}	9.970×10^{-4}
NH	$\beta = 0.4$	15	8.093×10^{-3}	9.508×10^{-4}
SD	$\alpha = 0.7$	27	9.239×10^{-3}	9.528×10^{-4}
NH	$\beta = 0.3$	17	7.958×10^{-3}	1.537×10^{-5}
SD	$\alpha = 0.8$	24	6.389×10^{-3}	7.818×10^{-4}
NH	$\beta = 0.2$	17	7.958×10^{-3}	1.537×10^{-5}
SD	$\alpha = 0.9$	22	8.311×10^{-3}	2.147×10^{-4}
NH	$\beta = 0.1$	18	9.830×10^{-3}	$\mathbf{3.352 \times 10^{-6}}$
SD	$\alpha = 1$	20	5.830×10^{-3}	4.119×10^{-6}
NH	$\beta = 0$	20	7.597×10^{-3}	4.119×10^{-6}

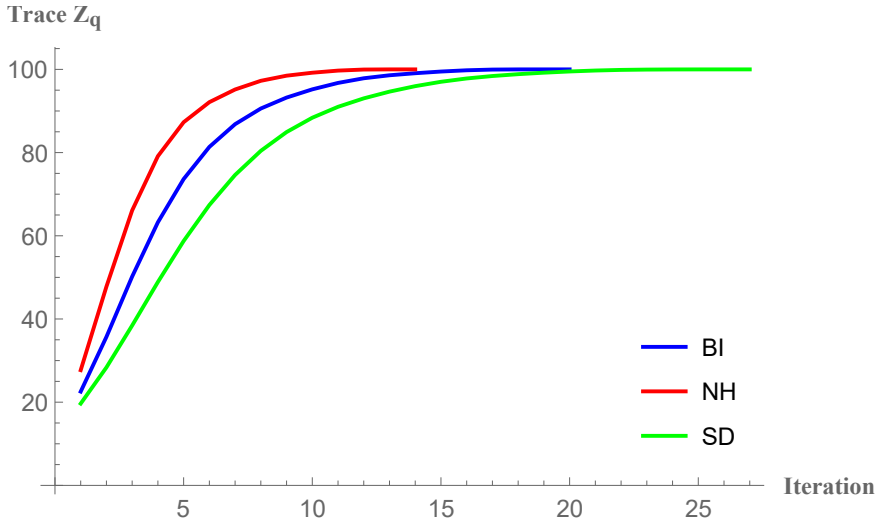

Fig. 4.8: Trace of $\mathcal{Z}_q$ for example 4.3

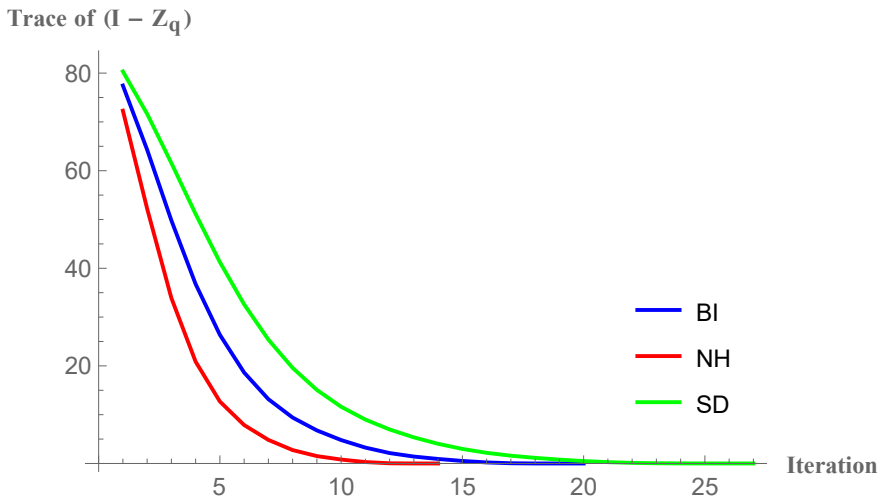

Fig. 4.9: Trace of $I - \mathcal{Z}_q$ for example 4.3

5 Applications of orthogonal projection

5.1 Signal filtering process

The use of a projection matrix in signal processing serves several important purposes. A projection matrix is used to project or map data onto a lower-dimensional subspace or to filter out certain components of a signal. Specifically, it can be applied for tasks such as noise reduction, signal filtering, and dimensionality reduction. In the next we see how NH method is useful in the context of signal filtering.

We generate a signal with 100 samples and signal X is composed of three sine waves with frequencies of $10Hz$, $20Hz$ and $30Hz$.

Mathematically, the signal can be expressed as

$$X(t) = \sin[2\pi 10t] + 0.5\sin[2\pi 20t] + 0.2\sin[2\pi 30t],$$

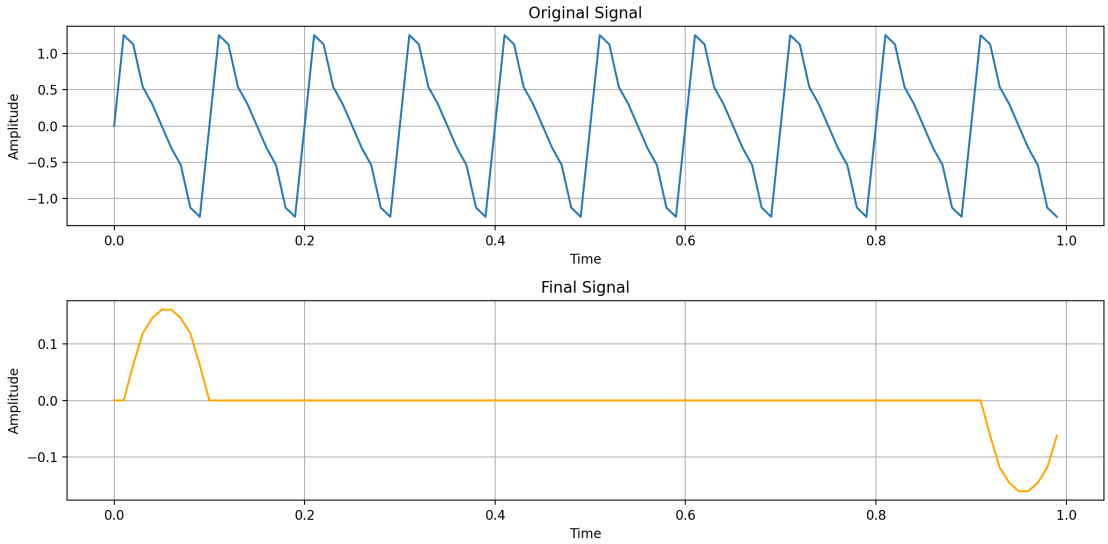

Fig. 5.1: Comparison of Original and Filtered Signal

where t represents time and $t \in [0.1, 0.01]$. The Toplitz matrix $\mathcal{U}$ is constructed from the first n filters samples of the signal. A Toplitz matrix is a matrix in which each descending diagonal from left to right is constant. In the case, the matrix $\mathcal{U}$ represents the autocorrelation structure of the signal over a window of n filter samples. Then, the projection, matrix is computed using the iterative scheme and the filtered signal y is obtained by applying the orthogonal projection matrix to the original signal. The original and filtered signals are plotted in Figure 5.1 for comparison. This simulates a signal with multiple frequency components.

The top plot presents the original signal as a function of time. It appears to be a composite waveform, formed by combining several sinusoidal components of different frequencies. The signal exhibits a repetitive pattern with oscillations of varying amplitudes and frequencies. The dominant frequency corresponds to the component with the highest amplitude.

The bottom plot shows the filtered signal, illustrating the effect of the applied filter in attenuating specific frequency components. The signal has a repetitive pattern with oscillations at different amplitudes and frequencies. The dominant frequency is the one with the highest amplitude. The signal in this plot has a drastically different shape. It starts with a peak, flattens out for a large portion, and then dips near the end before returning to zero. This could indicate that the iterative

method has filtered out certain frequency components or transformed the signal to highlight specific features.

5.2 State estimation [14]

Fred Schweppe and his research group first formulated state estimation for electric transmission grids, as shown in Figure 5.2. The primary motivation for state estimation is to conduct computer analysis of the network based on the current set of measurements. Specifically, the goal is to determine the bus voltage phasor magnitudes $|V_k|$ for all $k = 1, ..., N$ buses in the network. Consider the DC circuit depicted below, where current injections I_1, I_2 and voltage e are unknown. Let the resistances $R_1 = R_2 = R_3 = 1.0\,\Omega$ The measurements are as follows:

- Meter B1: $i_{1,2} = 1.0\,\text{A}$.
- Meter B2: $i_{3,1} = -3.2\,\text{A}$.
- Meter B3: $i_{2,3} = 0.8\,\text{A}$.
- Meter V: $e = 1.1\,\text{V}$.

Finding the circuit's state, in particular the nodal voltages V_1, V_2 and the voltage e across the voltage source, is the work at hand.

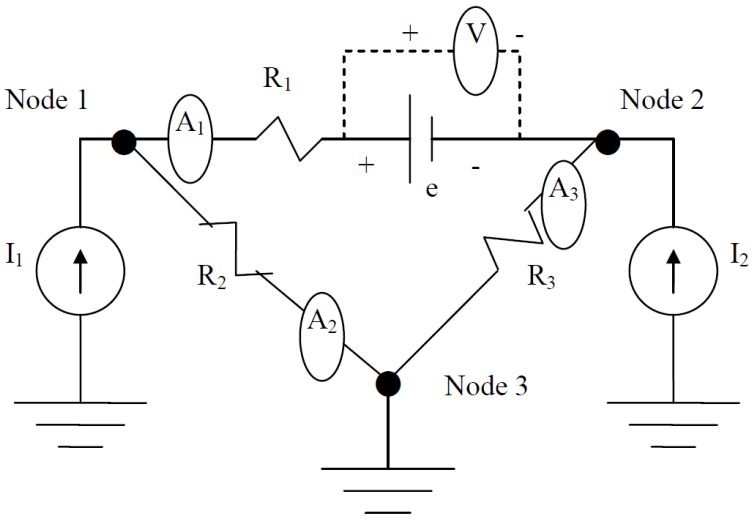

Fig. 5.2: State Estimation

We may write down all of the recorded currents in terms of node voltages, as well as our single voltage measurement

$$\begin{aligned} i^m_{1,2} &= v_1 - v_2 - e = 1.0, \\ i^m_{3,1} &= -v_1 = -3.2, \\ i^m_{2,3} &= v_2 = 0.8, \\ e &= 1.1. \end{aligned}$$

Putting everything above into matrix form: $\mathcal{U}x = b$

$$\begin{bmatrix} 1 & -1 & -1 \\ -1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} v_1 \\ v_2 \\ e \end{bmatrix} = \begin{bmatrix} 1.0 \\ -3.2 \\ 0.8 \\ 1.1 \end{bmatrix}.$$

We have a measurement matrix B and a measurement vector b which includes noise. First we'll form a projection matrix (P) using our iterative method (2.1)

$$P = \begin{bmatrix} 0.75 & -0.25 & -0.25 & -0.25 \\ -0.25 & 0.75 & -0.25 & -0.25 \\ 0.25 & -0.25 & 0.75 & -0.25 \\ -0.25 & -0.25 & -0.25 & 0.75 \end{bmatrix}.$$

Then we compute the best estimate of the state vector

$$b_{est} = \begin{bmatrix} 1.075 \\ 3.125 \\ 0.875 \\ 1.675 \end{bmatrix}$$

to get least square solution $\hat{x}$. For this we solve $\mathcal{U}\hat{x} = b$

$$\hat{x} = \begin{bmatrix} 0.25 & -0.75 & 0.25 & 0.25 \\ -0.25 & -0.25 & 0.75 & -0.25 \\ -0.25 & -0.25 & -0.25 & 0.75 \end{bmatrix} \begin{bmatrix} 1.0 \\ -3.2 \\ 0.8 \\ 1.1 \end{bmatrix} = \begin{bmatrix} 3.125 \\ 0.875 \\ 1.175 \end{bmatrix}.$$

The residual, $\hat{r}$, is given by

$$\hat{r} = b - b_{\text{est}} = \begin{bmatrix} 1.0 \\ -3.2 \\ 0.8 \\ 1.1 \end{bmatrix} - \begin{bmatrix} 1.075 \\ 3.125 \\ 0.875 \\ 1.675 \end{bmatrix} = \begin{bmatrix} -0.075 \\ -0.075 \\ -0.075 \\ -0.075 \end{bmatrix}.$$

This residual is orthogonal to the column space of U. Understanding the residual is crucial for assessing the quality of the approximate solution.

5.3 Statically determinate truss problem [10]

A truss is a fundamental type of engineering structure that offers cost-effective solutions for various engineering constructions, especially in the design of bridges and buildings with large spans. Consequently, the statically determined truss is a significant concern in structural engineering to evaluate related forces and responses. In this context, we consider a specific truss problem illustrated in Figure 5.3, which will support the development and implementation of Schulz-type iterative schemes.

The structure is attached as a stationary support with bottom left and a roller support with bottom right. Assume the truss is subjected to the setup as a 500-pound force acting at node 3 and node 5, whereas node 4 forces 1000-pound. The direction of all forces is vertical downward.

Our goal is to calculate the resulting forces on the rollar and stationary support with the members and the reaction forces. The direction of the action forces shown in Figure 5.3. There are twelve linear equations with twelve unknowns to balance the vertical and horizontal parts of force at each node corresponding to the linear system $\mathcal{U}Y = B$

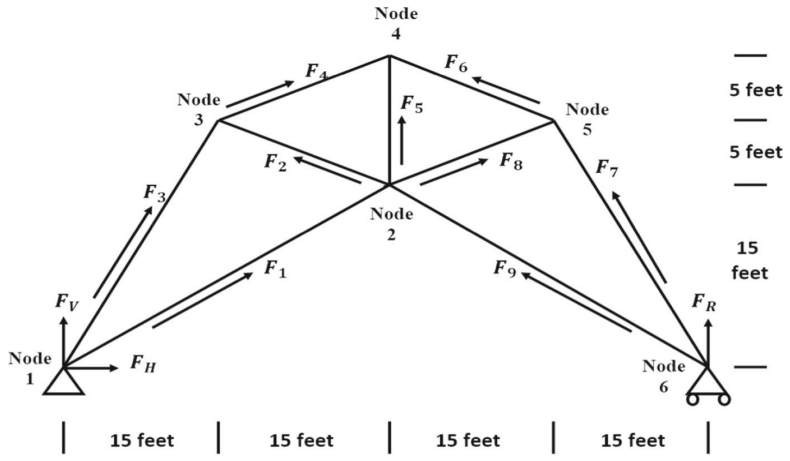

Fig. 5.3: Forces on a statically determinate truss

$$\mathcal{U} = \begin{bmatrix}
b_1 & 0 & b_3 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\
b_2 & 0 & b_4 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\
-b_2 & -b_6 & 0 & 0 & 0 & 0 & 0 & b_6 & b_2 & 0 & 0 & 0 \\
-b_1 & b_5 & 0 & 0 & 1 & 0 & 0 & b_5 & -b_1 & 0 & 0 & 0 \\
0 & b_6 & -b_4 & b_6 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & -b_5 & -b_3 & b_5 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & -b_6 & 0 & b_6 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & -b_5 & -1 & b_5 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & -b_6 & b_4 & -b_6 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & b_5 & -b_3 & -b_5 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & -b_4 & 0 & b_2 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & b_3 & 0 & b_1 & 0 & 1 & 0
\end{bmatrix},$$

$$Y = \begin{bmatrix} E_1 & E_2 & E_3 & E_4 & E_5 & E_6 & E_7 & E_8 & E_9 & E_{10} & E_{11} & E_{12} \end{bmatrix}^\top,$$

$$B = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 500 & 0 & 1000 & 0 & 500 & 0 & 0 \end{bmatrix}^\top$$

and $b_1 = \sin(\theta_1)$, $b_2 = \cos(\theta_1)$, $b_3 = \sin(\theta_2)$, $b_4 = \cos(\theta_2)$, $b_5 = \sin(\theta_3)$, $b_6 = \cos(\theta_3)$, $\theta_1 = \arctan\left(\frac{1}{2}\right)$, $\theta_2 = \arctan\left(\frac{4}{3}\right)$ and $\theta_3 = \arctan\left(\frac{1}{3}\right)$.
Expressing in the form $\mathcal{U}Y = B$ where

$$\mathcal{U} = \begin{bmatrix}
0.9904 & 0 & 0.3401 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\
0.1380 & 0 & -0.9403 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\
-0.1380 & 0.8328 & 0 & 0 & 0 & 0 & 0 & 0.83 & 0.1380 & 0 & 0 & 0 \\
-0.9904 & -0.5534 & 0 & 0 & 1 & 0 & 0 & -0.5534 & -0.9904 & 0 & 0 & 0 \\
0 & 0.8328 & 0.9403 & 0.8328 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0.5534 & -0.3401 & -0.5534 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & -0.8328 & 0 & 0.8328 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0.5534 & -1 & 0.5534 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & -0.8328 & -0.9403 & 0.8328 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & -0.5534 & -0.3401 & 0.5534 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0.9403 & 0 & 0.1380 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0.3401 & 0 & 0.9904 & 0 & 1 & 0
\end{bmatrix}.$$

From NH method (2.1), we can find projection matrix P, such that $PB = B$. This property can be used to verify whether a given B lies in the column space of $\mathcal{U}$.

6 Conclusion

An iterative technique for determining the orthogonal projection of any given matrix $\mathcal{U}$ is established in this study. Convergence analysis of the approach is investigated, and it is of second and third orders. The efficiency of the our method is illustrated with numerical examples. The outcomes of this approach are contrasted with existing iterative methods. Tables are used to provide performance, such as number of iterations, CPU time and error bounds for calculating orthogonal projections. One set of traces is found to be monotonically growing and converge to the matrix's rank, while another is monotonically falling and converges to nullity ($\mathcal{U}^*$). Furthermore, our method worked well for all the parameters ranging between $0-1$, while SD method (1.9) performed well when α is close to 1. Our work is validated in the domains of state simulation, data reduction, and signal filtering procedure.The new method improved performance over the SD method.

The proposed iterative method demonstrates improved performance over the SD method and a better alternative to the quadratic convergent iterative scheme. The enhancement can be attributed to the introduction of an additional parameter. The new parameter plays a crucial role in the refinement of the efficiency of the method. On the other hand, compared to (1.8) and (1.9), the method (2.1) is more expensive in terms of the number of computing resources, which can lead to the occurrence of a rounding error.

Acknowledgement

The authors thank anonymous referees for valuable and useful suggestions and comments that led to a great improvement in the article.

Declarations

Conflict of Interest.
The authors declare that they have no competing interests.
Funding.
Authors declare that there is no funding available for this article.
Author Contributions.
All Authors (N.G. and H.N.) have contributed as follows: methodology, N.G. and H.N.; formal analysis and software, N.G.; validation, H.N.; writing – original draft preparation, N.G. and H.N.; writing – review and editing, H.N. All authors have read and approved the published version of the manuscript.

References

1. Andruchow, E., Corach, G., Mbekhta, M.: On the geometry of generalized inverses. Math. Nachr. **278** (7-8), 756-770 (2005)
2. Ben-Israel, A., Charnes, A.: Contributions to the theory of generalized inverses. J. Soc. Ind. Appl. Math. **11** (3), 667–699 (1963)
3. Ben-Israel, A., Cohen, D.: On iterative computation of generalized inverses and associated projections. SIAM J. Numer. Anal. **3** (3), 410–419 (1966)
4. Ben-Israel, A., Greville, T.N.E.: Generalized Inverses: Theory and Applications. Springer, New York (2006)
5. Breger, A., Orlando, J.I., Harar, P., Dörfler, M., Klimscha, S., Grechenig, C., Gerendas, B.S., Schmidt-Erfurth, U., Ehler, M.: On orthogonal projections for dimension reduction and applications in augmented target loss functions for learning problems. J. Math. Imaging Vis. **62**, 376-394 (2020)

6. Chen, H., Wang, Y.: A family of higher-order convergent iterative methods for computing the Moore-Penrose inverse. Appl. Math. Comput. **218** (8), 4012-4016 (2011)
7. Golub, G., Kahan, W.: Calculating the singular values and pseudo-inverse of a matrix. J. Soc. Ind. Appl. Math. Ser. B Numer. Anal. **2** (2), 205-224 (1965)
8. Guo, W., Huang, T.: Method of elementary transformation to compute Moore-Penrose inverse. Appl. Math. Comput. **216** (5), 1614-1617 (2010)
9. Katsikis, V.N., Pappas, D., Petralias, A.: An improved method for the computation of the Moore-Penrose inverse matrix. Appl. Math. Comput. **217** (23), 9828-9834 (2011)
10. Kaur, M., Kansal, M.: An efficient class of iterative methods for computing generalized outer inverse MT, S (2). J. Appl. Math. Comput. **64** (10), 709-736 (2020)
11. Li, H.B., Huang, T.Z., Zhang, Y., Liu, X.P., Gu, T.X.: Chebyshev-type methods and preconditioning techniques. Appl. Math. Comput. **218**, 260-270 (2011)
12. Li, W., Juan, L., Tiantian, Q.: A family of iterative methods for computing Moore-Penrose inverse of a matrix. Linear Algebra Appl. **438** (1), 47-56 (2013)
13. Li, W., Li, Z.: A family of iterative methods for computing the approximate inverse of a square matrix and inner inverse of a non-square matrix. Appl. Math. Comput. **215** (9), 3433–3442 (2010)
14. Monticelli, A.: State estimation in electric power systems: a generalized approach. Springer Science & Business Media (2012)
15. Nashine, H.K., Kansal, M., Kaur, M., Kanwar, V.: A generalized second order iterative algorithm for computing the Moore-Penrose inverse, under review in Filomat.
16. Petković, M.D., Stanimirović, P.S.: Iterative method for computing the Moore-Penrose inverse based on Penrose equations. J. Comput. Appl. Math. **235** (6), 1604-1613 (2011)
17. Stanimirović, P.S., Petković, M.D., Mosić, D.: Exact solutions and convergence of gradient based dynamical systems for computing outer inverses. Appl. Math. Comput. **412**, 126588 (2022)
18. Srivastava, S., Gupta, D.K.: An iterative method for orthogonal projections of generalized inverses. J. Appl. Math. Inform. **32** (1-2), 61-74 (2014)
19. Srivastava, S., Gupta, D.K.: Higher order iterations for Moore-Penrose inverses. J. Appl. Math. Inform. **32** (1-2), 171-184 (2014)

Received 23.11.2024
Revised 01.03.2025