Term Project

Teona Zurabashvili, Ariel Liu *
Department of Mathematics
Virginia Polytechnique Institute and State University
Blacksburg, VA 24060
teona94@vt.edu
lius6469@vt.edu

Abstract

In this term project, we explore the fundamentals of classical iterative methods for solving linear system of equations, $\mathbf{Ax} = \mathbf{b}$ where $\mathbf{A} \in \mathbb{R}^{n\times n}$, $\mathbf{b} \in \mathbb{R}^n$, and the solution $\mathbf{x} \in \mathbb{R}^n$. We first discuss basic construction, convergence criterion, and smallest number of iterations to reduce error from the initial by Jacobi method and Gauss-Seidel method. Then we discover Successive Over Relaxation (SOR) method, and lastly we use Richardson iteration and Chebyshev iteration to accelerate convergence.

Introduction

Throughout the semester, we have explored direct methods; such as Gaussian Elimination, Cholesky Decomposition, and QR Decomposition. According to our analysis, these methods are expensive but they guarantee maximal accuracy in approximation approximation to the exact solution $\mathbf{x}$. So in this term project, we explore two general frameworks for solving a linear system of equations, $\mathbf{Ax} = \mathbf{b}$ where $\mathbf{Ax} = \mathbf{b}$ where $\mathbf{A} \in \mathbb{R}^{n\times n}$, $\mathbf{b} \in \mathbb{R}^n$, and the solution $\mathbf{x} \in \mathbb{R}^n$: (i) Direct methods and (ii) Iterative methods.

In contrast to direct methods, iterative methods generate a sequence of vectors $\mathbf{x}_0, \mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_k, \ldots$ that in many cases provide increasingly accurate approximations to $\mathbf{x}$. Practical iterative methods are important because they generate elements of this vector cheaply, at a cost comparable to the cost of a single matrix-vector multiply for each new vector iterate. If a moderate requirement of accuracy is sufficient, iteration can be stopped at an early stage, at a net cost far below that a direct method (such as Gaussian elimination) might have required. The danger of these iterative methods that need to be accounted, is that convergence of $\mathbf{x}_k$ to $\mathbf{x}$ may be extremely slow; in such cases, many iterations may be required driving the net cost up to an unacceptable level.

This term project is organized as follows. Section I analyses basic construction, particularly considering consider scenarios when $\mathbf{A} \in \mathbb{R}^{n\times n}$ is singular and proving convergence of specific iterative methods using the concept of spectral radius. Section II explores two basic iteration methods, "The Jacobi Iteration" and "Gaus-Siedel" Iteration. Section III discusses the relevance of Successive Over-Relaxation, which refers to acceleration of the convergence process, by extrapolating an improved estimate. And we also discuss the challenges associated with SOR methods. Section IV dives deep into "Vector Acceleration" and explore several relevant methods along with their pros and cons. Section V concludes the paper.

*Ph.D. on track in mathematics

1 Basic Construction for Iterative Methods

We begin by writing $\mathbf{A}$ as the difference of two matrices, $\mathbf{A} = \mathbf{M} - \mathbf{N}$ where $\mathbf{M}$ is invertible, which is called a splitting of $\mathbf{A}$. Obviously there are many ways one could do this and we will have to explore ways of distinguishing useful ways of defining a splitting for $\mathbf{A}$. $\mathbf{M}$ is called the preconditioning matrix and $\mathbf{N}$ is the residual matrix.

Once a splitting has been selected, i.e.., $\mathbf{A} = \mathbf{M} - \mathbf{N}$, one may then rearrange the equation $\mathbf{Ax} = \mathbf{b}$ to get $\mathbf{Mx} = \mathbf{Nx} + \mathbf{b}$ or equivalently

$$\mathbf{x} = \mathbf{M}^{-1}\mathbf{N}\mathbf{x} + \mathbf{M}^{-1}\mathbf{b}$$

Note that the equation above is exactly satisfied for the true solution $\mathbf{x}$. Thus, if an initial approximation $\mathbf{x}_0$ is close to the exact solution $\mathbf{x}$ and $\mathbf{x}_0$ is substituted for $\mathbf{x}$ on the right-hand side then the resulting " $\mathbf{x}$ " on the left-hand side should be close to $\mathbf{x}$ as well. Furthermore, if it is possible to insure that the resulting left-hand side actually is closer to the exact solution $\mathbf{x}$ than the original approximation $\mathbf{x}_0$, we could repeat the process to get a still better approximation. This leads to the basic iteration for all the iterative methods we will consider here:

$$\mathbf{x_{k+1}} = \mathbf{M^{-1}N x_k} + \mathbf{M^{-1}b}, \quad k = 0, 1, 2, 3, \ldots$$

But what determines whether or not the iterates do get closer to $\mathbf{x}$ as the method proceeds? And then how quickly will they get close to $\mathbf{x}$?

Problem 1.1

Assume that the splitting $\mathbf{A} = \mathbf{M} - \mathbf{N}$ to solve the linear system $\mathbf{Ax} = \mathbf{b}$. Let $\mathbf{x_0}$ be the initial iterate for the basic iteration. Define the error vector for the iteration index k as $\mathbf{e_k} = \mathbf{x_k} - \mathbf{x}$. Then, show that $\mathbf{e_k} = (\mathbf{M^{-1}N})^{\mathbf{k}}\mathbf{e_0}$.

Proof. Consider the base step when $\mathrm{k} = 0$, then we have $\mathbf{e_0} = \mathbf{x_0} - \mathbf{x}$. Now let's consider when $\mathrm{k} = 1$, we get

$$\begin{aligned}
\mathbf{e_1} &= \mathbf{x_1} - \mathbf{x} \\
&= \mathbf{M^{-1}N x_0} + \mathbf{M^{-1}b} - \mathbf{M^{-1}Nx} - \mathbf{M^{-1}b} \\
&= \mathbf{M^{-1}N x_0} - \mathbf{M^{-1}Nx} \\
&= \mathbf{M^{-1}N}(\mathbf{x_0} - \mathbf{x}) \\
&= \mathbf{M^{-1}N e_0}
\end{aligned}$$

And then for $\mathrm{k} = 2$, we get

$$\mathbf{e_2} = \mathbf{M^{-1}N}(\mathbf{M^{-1}N x_1} + \mathbf{M^{-1}b}) + \mathbf{M^{-1}b} - \mathbf{M^{-1}Nx} - \mathbf{M^{-1}b}$$

Now, we want to find what is $\mathbf{x_2}$, consider

$$\begin{aligned}
\mathbf{x} &= \mathbf{M^{-1}N x_1} + \mathbf{M^{-1}b} \\
&= \mathbf{M^{-1}N}(\mathbf{M^{-1}N x_0} + \mathbf{M^{-1}b}) + \mathbf{M^{-1}b} \\
&= (\mathbf{M^{-1}N})^{\mathbf{2}}\mathbf{x_0} - (\mathbf{M^{-1}N})^{\mathbf{2}}\mathbf{x} \\
&= (\mathbf{M^{-1}N})^{\mathbf{2}}(\mathbf{x_0} - \mathbf{x}) \\
&= (\mathbf{M^{-1}N})^{\mathbf{2}}\mathbf{e_0}
\end{aligned}$$

Inductively, assume that for all k, $\mathbf{e_k} = (\mathbf{M^{-1}N})^{\mathbf{k}}\mathbf{e_0}$. Then it suffices to show:

$$\mathbf{e_{k+1}} = (\mathbf{M^{-1}N})^{\mathbf{k+1}}\mathbf{e_0}$$

.

$$\begin{aligned}
\mathbf{e_{k+1}} &= \mathbf{x_{k+1}} - \mathbf{x} \\
&= \mathbf{M^{-1}N x_k} + \mathbf{M^{-1}b} - \mathbf{M^{-1}Nx} - \mathbf{M^{-1}b} \\
&= \mathbf{M^{-1}N}(\mathbf{x_k} - \mathbf{x}) \\
&= \mathbf{M^{-1}N e_k} \\
&= \mathbf{M^{-1}N}(\mathbf{M^{-1}N})^{\mathbf{k}}\mathbf{e_0} \\
&= (\mathbf{M^{-1}N})^{\mathbf{k+1}}\mathbf{e_0}
\end{aligned}$$

□

Problem 1.2

When $\mathbf{M^{-1}N}$ be diagonalizable, i.e., it has basis of eigenvectors $\{\mathbf{u}_1, \mathbf{u}_2, \mathbf{u}_3, \ldots, \mathbf{u}_n\}$, with associated eigenvalues $\{\lambda_1, \lambda_2, \lambda_3, \ldots \lambda_n\}$. We will show that the basic iteration

$$\mathbf{x_{k+1}} = \mathbf{M^{-1}N x_k} + \mathbf{M^{-1}b}, \quad k = 0, 1, 2, 3, \ldots$$

converges, i.e., $\mathbf{e}_k \to 0$, if and only if $\max_i |\lambda_i| < 1$.

if $\mathbf{e}_0$ were purely in the direction of the eigenvector associated with the largest magnitude eigenvalue, $\max_i |\lambda_i|$; and this slowest decay rate will satisfy

$$\|\mathbf{e}_k\| \approx \left(\max_i |\lambda_i|\right) \|\mathbf{e}_{k-1}\| \approx \left(\max_i |\lambda_i|\right)^k \|\mathbf{e}_0\|$$

Proof. Since $\mathbf{M^{-1}N}$ is diagonalizable, we get:

$$\mathbf{M^{-1}N} = \mathbf{V}\Lambda\mathbf{V^{-1}}$$

Where $\mathbf{V} \in \mathbb{R}^{n\times n}$ whose columns are the given eigenvectors and Λ is $\mathbb{R}^{n\times n}$ diagonal matrix, with corresponding eigenvalues on its diagonal. Then we obtain the following identity:

$$\mathbf{e_k} = (\mathbf{M^{-1}N})^{\mathbf{k}}\mathbf{e_0} = (\mathbf{V}\Lambda\mathbf{V^{-1}})\mathbf{e_0} = \mathbf{V}\Lambda\mathbf{V^{-1}}\mathbf{V}\Lambda\mathbf{V^{-1}}\ldots\mathbf{V}\Lambda\mathbf{V^{-1}}e_0$$

$\mathbf{V}\Lambda\mathbf{V^{-1}}$ would repeat k times. Notice that $\mathbf{V^{-1}V} = \mathbf{I}$, thus we will end up with

$$\mathbf{e_k} = \mathbf{V}\Lambda^k\mathbf{V^{-1}}e_0.$$

Notice that on the diagonal of Λ we have eigenvalues to the power of k. To keep the entries of e_k finite, we would need to ensure that $lim_{k\to\infty}(max_i\{\lambda_i\}_{i=1}^n) < \infty$, which can only happen if $max_i\{|\lambda_i|\} < 1$ □

To analyze when the slowest decay occurs, I will first check that the equality holds when the initial error is in the direction of the dominant eigenvector. Afterwards we will show the approximate equality, when the initial error vector is not in the direction of the dominant eigenvector.

Let $\lambda_1 > \lambda_2 \geq ... \geq \lambda_n > 0$ Case 1: $\mathbf{e_0}||\mathbf{v_1}$, where $\mathbf{v_1}$ is the dominant eigenvector of $\mathbf{M^{-1}N}$. Thus, $\mathbf{e_0} = c\mathbf{v_1}$ for some scalar c. Then:

Thus the exact equality holds, when the initial error is purely in the direction of the dominant eigenvector:

$$||e_k|| = max_i|\lambda_i||e_{k-1}|| = max_i|\lambda_i|^k||e_0||$$

Case 2: $e_0 \not\parallel v_1$. Then e_0 can be written as a linear combination of the basis of the eigenspace spanned by n linearly independent eigenvectors.

Proof.

$$e_0 = \alpha_1 v_1 + \alpha_2 v_2 + \cdots + \alpha_n v_n$$
$$e_1 = M^{-1}Ne_0 = M^{-1}N(\alpha_1 v_1 + \alpha_2 v_2 + \cdots + \alpha_n v_n) =$$
$$\alpha_1 M^{-1}Nv_1 + \cdots + \alpha_n M^{-1}Nv_n = \alpha_1\lambda_1 v_1 + \cdots + \alpha_n\lambda_n v_n$$
$$\vdots$$
$$e_{k-1} = \alpha_1(M^{-1}N)^{k-1}v_1 + \cdots + \alpha_n(M^{-1}N)^{k-1} = \alpha_1\lambda_1^{k-1} + \cdots + \alpha_n\lambda_n^{k-1}v_n =$$
$$\lambda_1^{k-1}(\alpha_1 v_1 + \alpha_2(\frac{\lambda_2}{\lambda_1})^{k-1}v_1 + \cdots + \alpha_n(\frac{\lambda_n}{\lambda_1})^{k-1}v_n$$
$$e_k = \alpha_1(M^{-1}N)^k v_1 + \cdots + \alpha_n(M^{-1}N)^k = \alpha_1\lambda_1^k v_1 + \cdots + \alpha_n\lambda_n^k v_n =$$
$$\lambda_1^k(\alpha_1 v_1 + \alpha_2(\frac{\lambda_2}{\lambda_1})^k v_1 + \cdots + \alpha_n(\frac{\lambda_n}{\lambda_1})^k v_n)$$

Since, λ_1 is the dominant eigenvalue, as $k \longrightarrow \infty$, the fraction $(\frac{\lambda_i}{\lambda_1})^{k-1} \longrightarrow 0$ $(i = 2, 3, ...n)$. This yields the conclusion that for sufficiently large k,

$$lim_{k\to\infty}||e_{k-1}|| = ||\lambda_1^{k-1}\alpha_1 v_1||.$$

Then [2]

$$lim_{k\to\infty}||e_k|| = lim_{k\to\infty}||M^{-1}Ne_{k-1}|| = ||M^{-1}N(\lambda_1^{k-1}\alpha_1 v_1)||$$
$$= ||(\lambda_1^{k-1}\alpha_1)M^{-1}Nv_1||$$
$$= ||(\lambda_1^{k-1}\alpha_1\lambda_1 v_1||$$
$$= |\lambda_1|||e_{k-1}||$$

Computationally, k can be sufficiently large number but it cannot go to infinity. Therefore, we can say that for sufficiently large k,

$$||e_k|| \approx |\lambda_1|||e_{k-1}||$$

To show that $||e_k|| \approx |\lambda_1|^k||e_0||$ from our result above:

$$||e_k|| = ||\Sigma_{i=1}^n \alpha_i(M^{-1}N)^i v_i|| = ||\Sigma_{i=1}^n \alpha_i\lambda_i^i v_i|| = ||\Sigma_{i=1}^n \alpha_i O(\lambda_i^i)v_i|| \approx |\lambda_1|||\Sigma_{i=1}^n \alpha_i v_i|| = |\lambda_1|^k||e_0||.$$

This shows that

$$||e_k|| \approx |\lambda_1^k||e_0||.$$

□

By definition, the quantity $(\max_i |\lambda_i|)$ is called the spectral radius of $\mathbf{M^{-1}N}$ and is denoted by $\rho\left(\mathbf{M^{-1}N}\right)$. And we have shown that the basic iteration (1.2) is convergent if and only if $\rho\left(\mathbf{M^{-1}N}\right) = \max_i \left|\lambda_i\left(\mathbf{M^{-1}N}\right)\right| < 1$.

Now, suppose we have chosen a splitting of $\mathbf{A} = \mathbf{M} - \mathbf{N}$ so that $\rho(\mathbf{M^{-1}N}) < 1$, then such a splitting is called a convergent splitting. And the iteration with this splitting should be converge to the exact solution $\mathbf{x}$ regardless of how good (or how bad) the initial approximation $\mathbf{x}_0$. In order to guarantee

[2]Remark: without loss of generality at the beginning of the problem we assumed that $\lambda_1 > \lambda_2 \geq \lambda_3 \geq ... \geq \lambda_n > 0$

that the error at the k th step is reduced to, say, $a\%$ of its initial value (i.e., $\|\mathbf{e}_k\| \leq 10^{-a} \|\mathbf{e}_0\|$), k must be large enough to satisfy $\rho\left(\mathbf{M}^{-1}\mathbf{N}\right)^k \leq 10^{-a}$ implying

$$k \geq \frac{-a}{\log_{10}\left(\rho\left(\mathbf{M}^{-1}\mathbf{N}\right)\right)}$$

that a is an integer. Based on the construction, Problem 1.3 in the following is a naive example and use what we have been introduced to predict how many iterations will be necessary to guarantee a reduction.

Problem 1.3

Let $\mathbf{A} = \begin{bmatrix} 2 & 1 \\ 1 & 3 \end{bmatrix}$ and consider the splitting

$$\mathbf{M} = \begin{bmatrix} 2 & 0 \\ 0 & 3 \end{bmatrix} \quad \text{and} \quad \mathbf{N} = \begin{bmatrix} 0 & -1 \\ -1 & 0 \end{bmatrix}$$

Predict with careful justification how many iterations will be necessary to guarantee a reduction in the error to 0.1% (i.e. 10^{-3}) of its initial value.

Solution. *By the definition of* $\mathbf{M}$*, we know the inverse of* $\mathbf{M}$ *is* $\mathbf{M^{-1}} = \begin{bmatrix} \frac{1}{2} & 0 \\ 0 & \frac{1}{3} \end{bmatrix}$.

$$M^{-1}N = \begin{bmatrix} \frac{1}{2} & 0 \\ 0 & \frac{1}{3} \end{bmatrix} \begin{bmatrix} 0 & -1 \\ -1 & 0 \end{bmatrix} \begin{bmatrix} 0 & -\frac{1}{2} \\ -\frac{1}{3} & 0 \end{bmatrix}$$

And so in order to find eigenvalues, we want to set up det$(\mathbf{M^{-1}N} - \lambda\mathbf{I}) = 0$*, that is*

$$det\left(\begin{bmatrix} -\lambda & -\frac{1}{2} \\ -\frac{1}{3} & -\lambda \end{bmatrix}\right) = 0$$

To calculate determinant of the above matrix, we got

$$(-\lambda)(-\lambda) - (\frac{-1}{2})(\frac{-1}{3}) = 0$$

Solve for λ*, which gives us* $-\sqrt{\frac{1}{6}}$ *and* $\sqrt{\frac{1}{6}}$*. By the result from Problem 1.2, we claim that* $\rho\left(\mathbf{M}^{-1}\mathbf{N}\right) = \max_i \left|\lambda_i\left(\mathbf{M}^{-1}\mathbf{N}\right)\right| = \sqrt{\frac{1}{6}} < 1$*. By using the relationship between* $\mathbf{M^{-1}N}$ *and the number of iteration, we want to guarantee a reduction in the error to* 0.1% *of its initial value, that is* $||e_k|| \leq 10^{-3}||e_0||$*. Thus,* k *must be large enough to justify* $\rho(\mathbf{M^{-1}N})^k \leq 10^{-3}$*. So,* $0 < \rho(\mathbf{M^{-1}N}) \leq 10^{\frac{-3}{k}}$*, then we have* $-\frac{3}{k} \geq log_{10}\rho(\mathbf{M^{-1}N})$*, and that is* $\frac{3}{k} \leq -log_{10}\rho(\mathbf{M^{-1}N})$*, which is* $k \geq \frac{-3}{log_{10}\rho(\mathbf{M^{-1}N})}$*. Thus, we have* $k \geq \frac{-3}{\log_{10}\left(\sqrt{\frac{1}{6}}\right)} = \frac{3570}{463} \approx 7.7106$*. Therefore,* $k = 8$ *and so 8 iterations will be necessary to guarantee a reduction in error to* 0.1% *of its initial value.*

After trying the naive example, we would like to explore the fact when the matrix $\mathbf{A}$ is singular.

Problem 1.4

Show that if $\mathbf{A}$ is singular then the iteration $\mathbf{x}_{k+1} = \mathbf{M}^{-1}\mathbf{N}\mathbf{x}_k + \mathbf{M}^{-1}\mathbf{b}$ will not be convergent in general, even if $\mathbf{M}$ is nonsingular.

Proof. Suppose $\mathbf{A}$ is singular and $\mathbf{M}$ is nonsingular. By splitting, we know $\mathbf{A} = \mathbf{M} - \mathbf{N}$ and so $\mathbf{N} = \mathbf{M} - \mathbf{A}$. Assume the iteration $\mathbf{x}_{k+1} = \mathbf{M}^{-1}\mathbf{N}\mathbf{x}_k + \mathbf{M}^{-1}\mathbf{b}$. If we want to show that $\mathbf{x_k}$ does not converge as k goes to infinity, it is sufficient to find one eigenvalue of $\mathbf{M^{-1}N}$ to be 1, which would give us the spectral radius $\rho(\mathbf{M^{-1}N})$ is at least 1. By the definition of eigenvalues and eigenvalues, we have $\mathbf{Av} = \lambda\mathbf{v}$ that λ is eigenvalue of $\mathbf{A}$ and $\mathbf{v}$ is the eigenvector of $\mathbf{A}$. Since $\mathbf{A}$ is

singular, then there exists some λ of $\mathbf{A}$ such that $\lambda = 0$ and there exist some vector $\mathbf{v} \in \ker(A - \lambda I)$ such that $\mathbf{Av} = \lambda\mathbf{v} = \mathbf{0}$. Then consider

$$\begin{aligned} M^{-1}N &= M^{-1}(M - A)v \\ &= M^{-1}Mv - M^{-1}Av \\ &= Iv - M^{-1}Av \\ &= v - 0 \\ &= v. \end{aligned}$$

Thus, we obtained $\mathbf{M^{-1}Nv} = \mathbf{v}$. By considering the definition of eigenvalue and eigenvector of $\mathbf{M^{-1}N}$, we know eigenvalue λ for $\mathbf{M^{-1}N}$ is 1. Thus, the spectral radius $\rho(\mathbf{M^{-1}N}) \geq 1$, which is what we want to show. □

Problem 1.5

Now we will consider the matrix $\mathbf{A}(\tau) = \begin{bmatrix} 1+\tau & 1-\tau \\ 1-\tau & 1+\tau \end{bmatrix}$ with the splitting

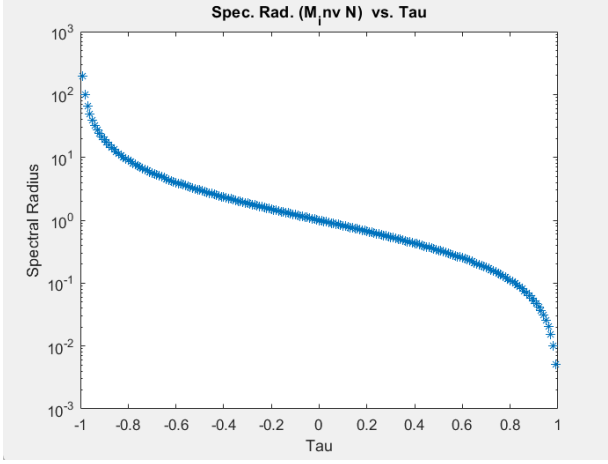
$$\mathbf{M}(\tau) = \begin{bmatrix} 1+\tau & 0 \\ 0 & 1+\tau \end{bmatrix} \quad \text{and} \quad \mathbf{N}(\tau) = \begin{bmatrix} 0 & -(1-\tau) \\ -(1-\tau) & 0 \end{bmatrix}.$$

Notice that at $\tau = 0, \mathbf{A}(\tau)$ is singular. Plot and discuss the values of $\rho\left(\mathbf{M}(\tau)^{-1}\mathbf{N}(\tau)\right)$ vs. τ for $\tau \in [-1, 1]$

Solution.
```
clear;
clc;
tau = -1:0.01:1;
n = size(tau);
n = n(2);
spectral_radius = [];
for i = 1: n
    A_tau = [1+tau(:,i) 1-tau(:,i); 1-tau(:,i) 1+tau(:,i)];
    M_tau = [1+tau(:,i) 0; 0 1+ tau(:,i)];
    N_tau = [0 -(1-tau(:,i)); -(1-tau(:,i)) 0];
    M_inv = [1/(1+tau(:,i)) 0; 0 1/(1+tau(:,i))];
    M_inv_N =  M_inv * N_tau
    spectral_radius = [spectral_radius,max(abs(eig(M_inv_N)))];
end
semilogy(tau, spectral_radius, '*');
title('Spec. Rad. (M_inv N)  vs. Tau');
xlabel('Tau');
ylabel('Iteration Index');
```

Keeping in mind the log scale of the plot below, we can notice that, on $\tau \in [-1, 1]$, the larger τ gets, the larger the spectral radius. For example, when tau is 0.19, the corresponding spectral radius is ≈ 0.680672.

As we realized, sometimes it is not possible to calculate the eigenvalues of $\mathbf{M^{-1}N}$ in order to find $\rho\left(\mathbf{M^{-1}N}\right)$. So, it might be easier for us to calculate the matrix norm $\left\|\mathbf{M^{-1}N}\right\|$ as calculating ∞-norm or calculating the 1-norm. For any eigenvalue λ of $\mathbf{M^{-1}N}$ (with associated eigenvector $\mathbf{u}$ having $\|\mathbf{u}\| = 1$), we have $|\lambda| \leq \left\|\mathbf{M^{-1}N}\right\|$ as a result. And in particular, we have $\rho\left(\mathbf{M^{-1}N}\right) \leq \left\|\mathbf{M^{-1}N}\right\|$. But we also want to point out that $\left\|\mathbf{M^{-1}N}\right\|$ will always provide cheap but sadly pessimistic estimate to the iteration convergent rate. Additionally, we cannot guarantee monotone convergence of the error based on the condition of $\rho\left(\mathbf{M^{-1}N}\right) < 1$. In the following, we consider an iteration matrix with the condition on basic iteration convergence and find the relationship between multiple quantities by plotting them out.



Problem 1.6

Consider an iteration matrix $\mathbf{B}(\alpha) = \mathbf{M}^{-1}\mathbf{N} = \begin{bmatrix} \alpha & 4 \\ 0 & \alpha \end{bmatrix}$, $\rho(\mathbf{B}(\alpha)) = |\alpha|$ and the basic iteration is convergent for $|\alpha| < 1$. Determine how many iterations are needed to get an iteration error $\|\mathbf{e}_k\|_\infty$ smaller than 10% of the initial error $\|\mathbf{e}_0\|_\infty$, if $\alpha = 0.99$ and $\mathbf{e}_0 = [1, 1]^T$? How does this compare with an estimate based on $\rho(\mathbf{B}(\alpha))^k$? Plot on a semilog scale both $\|\mathbf{e}_k\|_\infty$ vs. k and $\rho(\mathbf{B}(\alpha))^k$ vs k on the same plot. (use the matlab function semilogy)

Solution Notice that

$$\mathbf{B}^k(\alpha) = (\mathbf{M}^{-1}\mathbf{N})^k = \begin{bmatrix} \alpha & 4 \\ 0 & \alpha \end{bmatrix}^k$$

$$= \mathbf{B}(\alpha) = \mathbf{M}^{-1}\mathbf{N} = \begin{bmatrix} \alpha^k & k + 4\alpha \\ 0 & \alpha^k \end{bmatrix}$$

$$B(\alpha)^k = \begin{bmatrix} \alpha^k & (k-1)4\alpha \\ 0 & \alpha^k \end{bmatrix} \begin{bmatrix} 1 \\ 1 \end{bmatrix}$$

Therefore, $||B^k \begin{bmatrix} 1 \\ 1 \end{bmatrix}||_\infty = || \begin{bmatrix} \alpha^k + (k-1)4\alpha \\ \alpha^k \end{bmatrix}||_\infty = \alpha^k + (k-1)4\alpha$ [3]

Implementation:

```
clc;
clear;
alpha = 0.99;
e0 = [1;1];
B = [alpha 4; 0 alpha];
rho_B = abs(alpha);

e_k_list = [];
rho_list = [];
for i = 1: 10000
    e_k_norm_f = norm(B^i*e0, inf);
    e_k_list = [e_k_list, e_k_norm_f];
    rho_i = max(abs(eig(B^i)));
    rho_list = [rho_list, rho_i];
```

[3] Since alpha is positive, we will ignore the absolute value brackets

```
end


figure(1)
semilogy(rho_list, '-');
hold on;
semilogy(e_k_list, '--');
legend('Spectral Radius', 'Inf. Norm of k-th error')
xlabel('iteration index');
```

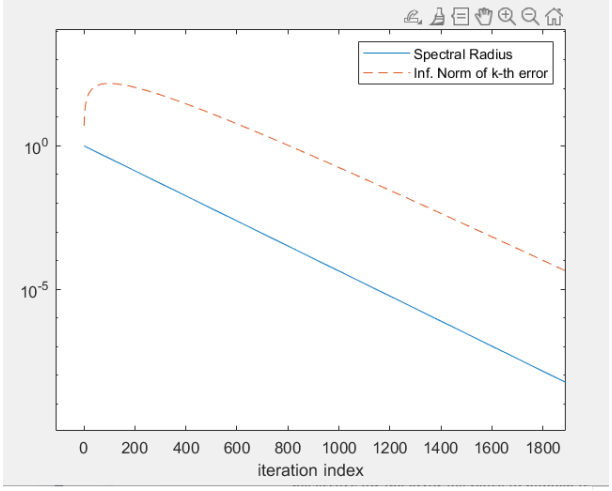


Figure 1: Spectral Radius, Infinity Norm of k-th error vs. Iteration Index

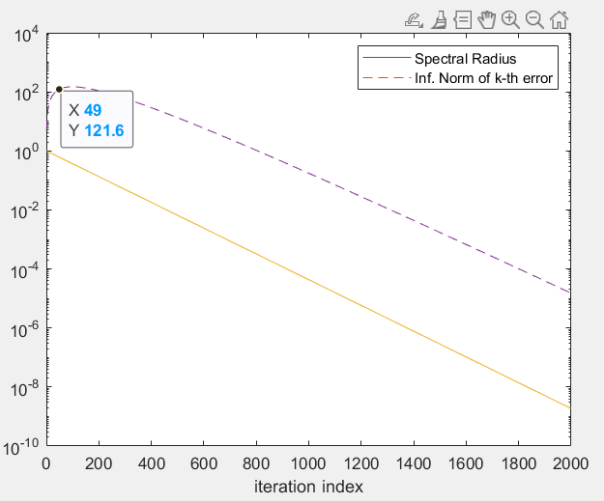


Figure 2: Spectral Radius, Infinity Norm of k-th error vs. Iteration Index

Even though the spectral radius continues decreasing, from the very beginning of the iteration, for approximately first 49 iterations, the normed bound, $||M^{-1}N||$ increases, then it decays along the spectral radius. Further, our results confirm that $||M^{-1}N||$ indeed dominates the spectral radius $\rho(M^{-1}N)$. Our conclusion is that, $\rho(M^{-1}N) < 1$ is both, necessary and sufficient for eventual convergence of the error. However, $\rho(M^{-1}N) < 1$ it is not sufficient to guarantee monotone convergence of error. Indeed, our plot indicates that the error from B^k initially increases then it decreases. Thus the iteration error does not converge monotonically. Moreover, number of iterations necessary to guarantee 10 percent decrease in the initial error, using the formula from problem 1.2, is 230.

To find the minimum numeber of iterations for getting an iteration error $||e_0||$ smaller than 10 percent of the initial error $||e_0||$, we would need to solve

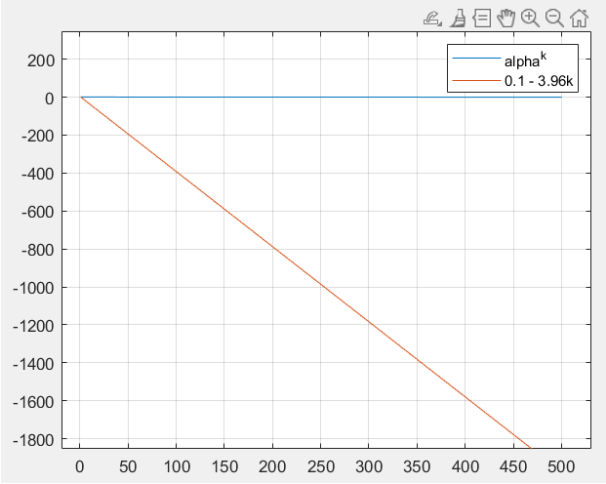
$$\alpha^k + 4(k-1)\alpha < 0.1$$

This means that we are interested in finding k, such that $\alpha^k < 1 - 4(k-1)\alpha$, where $\alpha = 0.99$. According to our results, k such that 0.99^k is dominated by $0.1 - 3.96(k-1)$ is 500.

```
clc;
clear;
k = 1: 1: 500;
alpha = 0.99;
exp_funct = [];
lin_funct = [];
termination_criteria = 0;
min_num_iter = 0;
for i = 1: 500
    exp_funct = [exp_funct, alpha^i];
    lin_funct = [lin_funct, 0.1 - 3.96*(i-1)];
    if exp_funct(i) > lin_funct(i)
        termination_criteria = 1;
    end
    if termination_criteria == 0
        break;
    end
    min_num_iter = i;
end

display('Minimum Number of Iterations to ensure that an
   iteration error is smaller than 10% of the initial error is
   : ')
min_num_iter

figure(1)
plot(k, exp_funct);
legend('alpha^k');
hold on;
plot(k, lin_funct);
legend('0.1 - 3.96k');
grid on;
```



2 Two Basic Iterations

Based on understanding of convergence criterion for the basis iteration from last section, we would like to discover and develop efficient good splitting strategies. That is choosing a splitting $\mathbf{A} = \mathbf{M} - \mathbf{N}$ so that linear systems having M as a coefficient matrix are easy to solve. Now, we know there are two possibilities that occur to us. One is when $\mathbf{M}$ is diagonal, and another one is when $\mathbf{M}$ is triangular.

So first, let's consider the situation when $\mathbf{M}$ is diagonal, that is also called the Jacobi iteration. Then, the iteration $\mathbf{x}_{k+1} = \mathbf{M}^{-1}\mathbf{N}\mathbf{x}_k + \mathbf{M}^{-1}\mathbf{b}$ means that at each step we solve the linear system $\mathbf{M}\mathbf{x}_{k+1} = \mathbf{N}\mathbf{x}_k + \mathbf{b}$. However, since $\mathbf{M}$ is a diagonal matrix, this is a very easy system to solve.

Problem 2.1

Consider the matrix given by

$$\mathbf{A} = \begin{bmatrix} 3 & 1 & -1 \\ 2 & 4 & 1 \\ 1 & -1 & 3 \end{bmatrix}$$

Without computing any eigenvalues, show the Jacobi method is convergent and give the smallest number of iterations that will be sufficient (independent of starting point) to reduce the error to 1% of its initial value.

Solution. *By the definition that we mentioned previously, we know* $\mathbf{M} = diag(\mathbf{A})$*, so we get* $\mathbf{M} = \begin{bmatrix} 3 & 0 & 0 \\ 0 & 4 & 0 \\ 0 & 0 & 3 \end{bmatrix}$*. And so pick matrix* $\mathbf{N} = \begin{bmatrix} 0 & 1 & -1 \\ 2 & 0 & 1 \\ 1 & -1 & 0 \end{bmatrix}$ *to check* $\mathbf{A} = \mathbf{M} - \mathbf{N}$*. By the matrix* $\mathbf{M}$*, we also know the inverse of* $\mathbf{M}$ *to be* $\mathbf{M^{-1}} = \begin{bmatrix} \frac{1}{3} & 0 & 0 \\ 0 & \frac{1}{4} & 0 \\ 0 & 0 & \frac{1}{3} \end{bmatrix}$.

Now consider

$$||\mathbf{M^{-1}N}|| = \left\| \begin{bmatrix} \frac{1}{3} & 0 & 0 \\ 0 & \frac{1}{4} & 0 \\ 0 & 0 & \frac{1}{3} \end{bmatrix} \begin{bmatrix} 0 & 1 & -1 \\ 2 & 0 & 1 \\ 1 & -1 & 0 \end{bmatrix} \right\| = \left\| \begin{bmatrix} 0 & \frac{1}{3} & -\frac{1}{3} \\ \frac{1}{2} & 0 & -\frac{1}{4} \\ \frac{1}{3} & -\frac{1}{3} & 0 \end{bmatrix} \right\| = \frac{5}{6} < 1.$$

Because of the relationship between $\rho(\mathbf{M^{-1}N})$ *and* $||\mathbf{M^{-1}N}||$*, we know*

$$\rho(\mathbf{M^{-1}N}) \leq ||\mathbf{M^{-1}N}|| < 1.$$

So, by transitivity, $\rho(\mathbf{M^{-1}N}) < 1$*. Thus, we can conclude the Jacobi method is convergent using the given* $\mathbf{A}$ *without computing any eigenvalues. By using the definition of iteration number under a specific percentage to reduce error from the initial value, we have*

$$\mathbf{k} \geq \frac{-2}{\mathbf{log_{10}}(\frac{\mathbf{5}}{\mathbf{6}})} \approx 25.2585$$

Therefore, iteration needs to be at least 26.

Problem 2.2

Suppose $\mathbf{A}$ is an $n \times n$ matrix that is diagonally dominant (which means that $\sum_{j \neq i} |a_{ij}| < |a_{ii}|$ for each row index i). Show that Jacobi's method is convergent. Furthermore, if $\xi_i = \sum_{j \neq i} \frac{|a_{ij}|}{|a_{ii}|}$ and $\mathbf{x}$ is the exact solution to $\mathbf{Ax} = \mathbf{b}$, show that the Jacobi iterates $\mathbf{x_k}$ satisfy

$$\|\mathbf{x} - \mathbf{x}_k\|_\infty \leq \max_i \frac{\xi_i}{1 - \xi_i} \|\mathbf{x}_k - \mathbf{x}_{k-1}\|_\infty$$

Notice that the right-hand side contains explicitly computable quantities.

Proof. First, we want to show Jacobi's method is convergent. Since $\sum_{j \neq i} |a_{ij}| < |a_{ii}|$, then we know $\sum_{j \neq i} \frac{|a_{ij}|}{|a_{ii}|} < 1$.

So

$$||\mathbf{M^{-1}N}||_\infty = \left\| \frac{1}{|a_{ii}|} \sum_{j \neq i} |a_{ij}| \right\|_\infty = \max \sum_{j \neq i} \frac{|a_{ij}|}{|a_{ii}|} < 1$$

Thus, $\rho(\mathbf{M^{-1}N}) \leq ||\mathbf{M^{-1}N}|| < 1$. Therefore, we can conclude that Jacobi's method is convergent.
We know that $\mathbf{x - x_k} = \mathbf{M^{-1}N}(\mathbf{x - x_k + x_k - x_{k-1}})$. Then we have

$$\begin{aligned} ||\mathbf{x - x_k}||_\infty &= ||\mathbf{M^{-1}N}(\mathbf{x - x_k + x_k - x_{k-1}})||_\infty \\ &\leq ||\mathbf{M^{-1}N}||_\infty ||\mathbf{x - x_k}|| + ||\mathbf{M^{-1}N}||_\infty ||\mathbf{x_k - x_{k-1}}||_\infty. \end{aligned}$$

Grouping terms containing $||\mathbf{M^{-1}N}||_\infty$ on one side of the inequality, yields:

$$||\mathbf{x - x_k}||_\infty (1 - ||\mathbf{M^{-1}N}||_\infty) \leq ||\mathbf{M^{-1}N}||_\infty ||\mathbf{x_k - x_{k-1}}||_\infty$$

Therefore, we have:

$$||\mathbf{x - x_k}||_\infty \leq \frac{||\mathbf{M^{-1}N}||_\infty}{(1 - ||\mathbf{M^{-1}N}||_\infty)} ||\mathbf{x_k - x_{k-1}}||_\infty$$

Then consider

$$\begin{aligned} ||\mathbf{M^{-1}N}|| &= \frac{1}{a_{ii}} \sum_{j \neq i} n_{ij} && n_{ij} \text{ as the entries in the matrix } \mathbf{N} \\ &= \frac{1}{a_{ii}} \sum_{j \neq i} a_{ij} && \text{since } \mathbf{M} = diag(\mathbf{A}) \text{ and } \mathbf{A} = \mathbf{M} - \mathbf{N} \\ &\leq \frac{1}{a_{ii}} \sum_{j \neq i} |a_{ij}| \\ &= \xi_i. \end{aligned}$$

Thus, we have $||\mathbf{M^{-1}N}||_\infty \leq \xi_i$. Now, consider $\frac{||\mathbf{M^{-1}N}||_\infty||}{1-||\mathbf{M^{-1}N}||_\infty} \leq \frac{\xi_i}{1-\xi_i} \leq \max_i \frac{\xi_i}{1-\xi_i}$. Therefore, we can conclude we got as desired. □

Now, let's consider the situation when $\mathbf{M}$ is triangular, that is to choose $\mathbf{M} = \text{tril}(\mathbf{A})$, which we called it Gauss-Seidel iteration. Then we solve the (lower) triangular system $\mathbf{Mx}_{k+1} = \mathbf{Nx}_k + \mathbf{b}$ at each step. In next problem, we write a MatLab file which applies both methods that we mentioned to predict the number of iterations to reduce error to some specific percentage of the initial magnitude.

Problem 2.3

Write a Matlab n-file that applies both the Jacobi and Gauss-Seidel methods to the linear system $\mathbf{Ax} = \mathbf{b}$ where

$$\mathbf{A} = \begin{bmatrix} \mathbf{T} & \mathbf{I} \\ \mathbf{I} & \mathbf{T} \end{bmatrix}, \mathbf{T} = \begin{bmatrix} -4 & 1 & 0 & 0 & 0 \\ 1 & -5 & 1 & 0 & 0 \\ 0 & 1 & -4 & 1 & 0 \\ 0 & 0 & 1 & -5 & 1 \\ 0 & 0 & 0 & 1 & -4 \end{bmatrix}$$

and $\mathbf{I}$ is the 5×5 identity matrix; the right-hand side is

Use as an initial approximation $\mathbf{x}_0 = \mathbf{b}$ (i.e., take the right-hand side itself as a crude approximation to the solution) and terminate the iterations when the relative change in each component is less than 0.05% from one iteration to the next. Additionally, your m-file should calculate and record the (true) error at each iteration and plot error vs iteration index on semilog axes at the end (use the matlab

function semilogy). For the purposes of calculating the error, you can use the fact that the exact solution for the given right hand side is

$$\mathbf{x} = [-1 - 2 - 2 - 1 - 1 - 2 - 1 - 2 - 1 - 2]^T.$$

Calculate the spectral radius of both the Jacobi iteration matrix and the Gauss-Seidel iteration matrix and use this to predict how many iterations should be necessary to reduce the error to 1% of its initial magnitude. Comment on your plot and computational results and compare your predictions with the performance of the m-file you've written.

Implementation:

```
clear;
clc;
T = [-4 1 0 0 0 ;
     1 -5 1 0 0;
     0 1 -4 1 0;
     0 0 1 -5 1;
     0 0 0 1 -4];
I = eye(5);
A = [T I; I T];
b = [0; 6; 3; 1; 1; 6; -1; 4; 0; 6];
x = [-1; -2; -2; -1; -1; -2; -1; -2; -1; -2];

x0 = b;

% Applying Jacobi iteration
M_jac = diag(A);
M_jac = diag(M_jac);
N_jac = M_jac-A;

%M diagonal, thus M^-1 would be reciprocal of diagonals inv(M)
   is cheap
%operation
M_inv_jac = M_jac\eye(10);
%finding the spectral radius of M_inv*N
spec_radius_M_inv_N_jac = max(abs(eig(M_inv_jac*N_jac)));
x_list_jac = [];
errors_jac = [];
two_norms_of_errors_jac= [];
true_error_approx_jac = [];
e0 = x - x0;
x_current_jac = x0;

k = 20;
for i = 1: k
    M_times_x_next = N_jac*x_current_jac + b;
    x_next_jac = M_jac \ M_times_x_next;
    %collecting the errors at each iteration
    e_i_jac = x_next_jac - x_current_jac;

    %collecting error vectors
    errors_jac = [errors_jac, e_i_jac];
    x_current_jac = x_next_jac;

    %errors_2_norms
    two_norms_of_errors_jac = [two_norms_of_errors_jac, norm(
       errors_jac(:, i), 2)];

    %from the
```

```
    true_error_approx_jac = [true_error_approx_jac,
       spec_radius_M_inv_N_jac^i*norm(e0,2)];

    termination_criteria = 0; %termination criteria is same as
       "have_we_found_one_over_point_5" and if we have not the
        first
    %break will stop the first inner loop the second will stop
       the outer loop
    p = size(e_i_jac);
    for j = 1:p(:,1)
        if  abs(e_i_jac(j)/x_current_jac(j)) >=0.0005
            termination_criteria = 1;
           break;
        end
    end
    if termination_criteria == 0
       break
    end
    %errors is a 10 by "i" matrix whose column space is the
       span of all error vectors at each iteration.

end



M_Gaus_Seidel = tril(A);
M_Gaus_Seidel_inv = M_Gaus_Seidel\eye(10);
N_Gaus_Seidel = M_Gaus_Seidel - A;
x_current_GS = x0;
errors_gs = [];
two_norms_of_errors_gs = [];
true_error_approx_gs = [];
spec_radius_M_inv_N_GS = max(abs(eig(M_Gaus_Seidel_inv*
   N_Gaus_Seidel)));
x_current_GS = x0;
h = 20;
for i = 1: h

    M_Gaus_Seidel_x_next_GS = N_Gaus_Seidel*x_current_GS + b;
    x_next_GS = M_Gaus_Seidel \ M_Gaus_Seidel_x_next_GS;
    %collecting the errors at each iteration
    e_i_gs = x_next_GS - x_current_GS;
    errors_gs = [errors_gs, e_i_gs];
    x_current_GS = x_next_GS;

    %errors_2_norms
    two_norms_of_errors_gs = [two_norms_of_errors_gs, norm(
       errors_gs(:, i), 2)];
    true_error_approx_gs = [true_error_approx_gs,
       spec_radius_M_inv_N_GS^i*norm(e0,2)];

    termination_criteria_gs = 0; %termination criteria is same
       as "have_we_found_one_over_point_5" and if we have not
       the first
    %break will stop the first inner loop the second will stop
       the outer loop
   p = size(e_i_gs);
   for j = 1:p(:,1)
```

```
        if  abs(e_i_gs(j)/x_current_GS(j)) >=0.0005
            termination_criteria = 1;
            break;
        end
    end
    if termination_criteria == 0
       break
    end
    %errors is a 10 by "i" matrix whose column space is the span
        of all error vectors at each iteration.

end

figure;
semilogy(two_norms_of_errors_jac, '*');
hold on;
semilogy(two_norms_of_errors_gs, '-');
legend('Jacobi Method','Gaus Seidel Mehtod');
ylabel('Error');
xlabel('iteration index');
grid on;


%Number of iterations necessary for the error in each component
    to decrease
%in Jacobi vs. Gaus-Seidel Method:

min_num_iterations_Jac = ceil(-2/log10(spec_radius_M_inv_N_jac)
    )
min_num_iterations_GS = ceil(-2/log10(spec_radius_M_inv_N_GS))
```

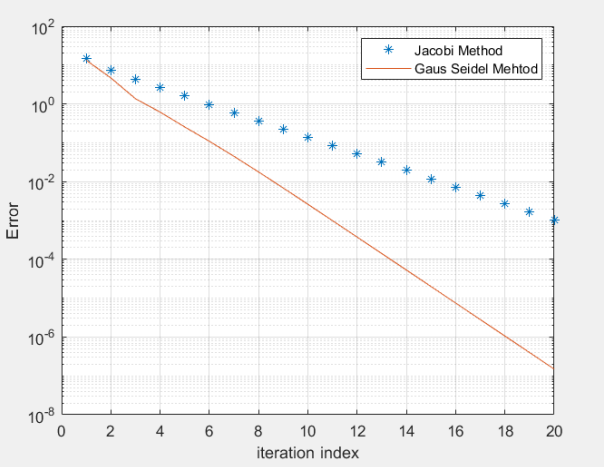


Figure 3: Jacobi vs. Gaus Seidel

Our results indicate that at every iteration, Jacobi Method dominates Gaus Seidel Method. Further, number of iterations that ensures decrease of the relative change in each component below 0.0005 in Jacobi and Gaus Seidel Method is respectively 10 and 5.

3 SOR Methods

Gaus - Seidel iterations will often give us monotone convergence of the Gauss-Seidel in the components of $\mathbf{x}$. This incentivizes us to accelerate the convergence process by combining current and previous iterates of the components of $\mathbf{x}$

Assume that we know the first $i-1$ components of the current iterate x_{k+1}, and all components of the previous iterate x_k for some index $i, 1 \leq i \leq n$. To determine the i th component, $x_i^{(k+1)}$, we can compute the Gauss-Seidel value $\hat{x}_i^{(k+1)}$ from the following equation:

$$a_{ii}\hat{x}_i^{(k+1)} + \sum_{j<i} a_{ij}x_j^{(k+1)} + \sum_{j>i} a_{ij}x_j^{(k)} = b_i$$

Then we can combine an improved value from the previous value $x_i^{(k)}$ and the current Gauss-Seidel value $\hat{x}_i^{(k+1)}$:

$$x_i^{(k+1)} = (1-\omega)x_i^{(k)} + \omega\hat{x}_i^{(k+1)}$$

for an optimal choice of ω. We say that the strategy is underrelaxation, if $\omega < 1$. Underrelaxation corresponds to interpolating a value for $x_i^{(k+1)}$ between $x_i^{(k)}$ and $\hat{x}_i^{(k+1)}$; The strategy is called overrelaxation, if $\omega > 1$. Overrelaxation corresponds to extrapolating a value for $x_i^{(k+1)}$ starting from $x_i^{(k)}$ and and finding entries beyond $\hat{x}_i^{(k+1)}$. Extrapolation is often used because it frequently accelerates convergence of the iteration process. These methods are called Successive OverRelaxation methods and are often referred as **SOR**. When $\omega = 1$, we are dealing with just Gauss-Seidel method.

Our main goal is to choose ω - the extrapolation parameter so that convergance of our iterative method is the fastest. Choosing a particular ω, would define an iterative method, but is there a way to express this iteration in terms of a splitting of A? This is possible by multiplying $a_{ii}\hat{x}_i^{(k+1)} + \sum_{j<i} a_{ij}x_j^{(k+1)} + \sum_{j>i} a_{ij}x_j^{(k)} = b_i$ by ω, and solving $x_i^{(k+1)} = (1-\omega)x_i^{(k)} + \omega\hat{x}_i^{(k+1)}$ for $\omega\hat{x}_i^{(k+1)}$. We will obtain:

$$a_{ii}\left(x_i^{(k+1)} - (1-\omega)x_i^{(k)}\right) + \omega\sum_{j<i} a_{ij}x_j^{(k+1)} + \omega\sum_{j>i} a_{ij}x_j^{(k)} = \omega b_i$$

We can divide the both sides of this equality by ω and rearrange its terms to obtain:

$$\frac{1}{\omega}a_{ii}x_i^{(k+1)} + \sum_{j=1}^{i-1} a_{ij}x_j^{(k+1)} = \frac{1}{\omega}\left[(1-\omega)a_{ii}x_i^{(k)}\right) - \omega\sum_{j=i+1}^{n} a_{ij}x_j^{(k)}\Bigg] + b_i$$

As we can see, we have a component iteration method. We can rewrite this method as already damiliar vector iteration $\mathbf{M}\mathbf{x_{k+1}} = \mathbf{N}\mathbf{x_k} + \mathbf{b}$ by defining a splitting of $\mathbf{A}$. Assume $\mathbf{D}$ is diagonal of $\mathbf{A} and \mathbf{L}$ is the strict lower triangle of $\mathbf{A}$; Further, assume that $\mathbf{U}$ be the strict upper triangle of $\mathbf{A}$[4]. Then for every $\omega \neq 0$, we get:

$$\mathbf{A} = \frac{1}{\omega}(\omega\mathbf{L} + \mathbf{D}) - \frac{1}{\omega}[(1-\omega)\mathbf{D} - \omega\mathbf{U}]$$

Now we can define the splitting of $\mathbf{A} = \mathbf{M} - \mathbf{N}$, where $\mathbf{M} = \frac{1}{\omega}(\omega\mathbf{L} + \mathbf{D})$ and $\mathbf{N} = \frac{1}{\omega}[(1-\omega)\mathbf{D} - \omega\mathbf{U}]$.

As we can see, the convergence rate of this iteration is determined by the size of the iteration matrix. $\mathbf{H}(\omega) = \mathbf{M}^{-1}\mathbf{N} = \left(\mathbf{I} + \omega\mathbf{D}^{-1}\mathbf{L}\right)^{-1}\left[(1-\omega)\mathbf{I} - \omega\mathbf{D}^{-1}\mathbf{U}\right]$.

[4]Strict upper or lower diagonal matrices have zeros on the main diagonals

Problem 3.1

For the matrix A given in Problem 2.3, graph the spectral radius of $\mathbf{H}(\omega)$ vs. ω for all $\omega \in [0, 2]$. With the help of this plot, determine the optimal relaxation parameter (the ω giving the smallest spectral radius). Predict how many iteration steps would be necessary to solve the linear system $\mathbf{Ax} = \mathbf{b}$ of Problem 2.3 to the tolerances stated in that problem. Modify your m-file of Problem 2.3 to perform an SOR iteration with the optimal relaxation parameter that you've obtained, and plot the error vs. iteration index as in Problem 2.3 together with the Jacobi and Gauss-Seidel error plots (all on one graph). Comment on your results.

Implementation:

```
clc;
clear;
T = [-4 1 0 0 0 ;
    1 -5 1 0 0;
    0 1 -4 1 0;
    0 0 1 -5 1;
    0 0 0 1 -4];
I = eye(5);
A = [T I; I T];
%constructing strict lower triangle of A
L = tril(A,-1);
D = diag(A);
D = diag(D);
U = triu(A,1);
b = [0; 6; 3; 1; 1; 6; -1; 4; 0; 6];
x_true = [-1; -2; -2; -1; -1; -2; -1; -2; -1; -2];
tol = 0.0005;

%constructing H(w)
w = 1: 0.01: 2;
n = size(w);
n = n(2);

spectral_radius_H_w = [];
%norm_x_true_minus_x_comp_each_entry = [];
for i = 1: n
    M = (1/w(:,i))*(w(:,i)*L+D);
    N = (1/w(:,i))*((1-w(:,i))*D - w(:,i)*U);
    M_inv = M\eye(10);
    H_w = M_inv*N;                                    %inv(eye(n)
        + w(:,i)*inv(D)*L) * ((1-w(:,i))*eye(n) - w*inv(D)*U);
    %spectral radius rho at every iteration
    spectral_radius_H_w = [spectral_radius_H_w, max(abs(eig(H_w
       )))];
end

%After the above loop, we will obtain the optimal
   overrelaxation parameter
%w = 1.12

w_opt = 1.12;
L = tril(A,-1);
D = diag(A);
D = diag(D);
U = triu(A,1);
M_opt = (1/w_opt)*(w_opt*L+D);
N_opt = (1/w_opt)*((1-w_opt)*D - w_opt*U);
```

```
inv_M_opt = M_opt\eye(10);
H_w_opt = inv_M_opt*N_opt;
spectral_radius_H_w_opt = max(abs(eig(H_w_opt)));

%Minimum number of iterations necessary to solve Ax = b
min_num_iterations = ceil(-2/log10(spectral_radius_H_w_opt));


%SOR Method
x0 = b;
error_SOR = 2;
x_previous = x0;

i = 1;
while error_SOR > 0.0005
      x_current = M_opt\(N_opt*x_previous + b);
      true_error_SOR(i) = norm((x_current - x_true),2);
      i = i+1;
      error_SOR = norm((x_current - x_previous), 2)/norm(
         x_previous,2);
      x_previous = x_current;

end


%Implementing Gaus Seidel Iteration Method:
x0 = b;
e0 = x_previous - x0;
M_Gaus_Seidel = tril(A);
M_Gaus_Seidel_inv = M_Gaus_Seidel\eye(10);
N_Gaus_Seidel = M_Gaus_Seidel - A;
x_current_GS = x0;
errors_gs = [];
two_norms_of_errors_gs = [];
true_error_approx_gs = [];
spec_radius_M_inv_N_GS = max(abs(eig(M_Gaus_Seidel_inv*
   N_Gaus_Seidel)));
x_current_GS = x0;
h = 20;
for i = 1: h

    M_Gaus_Seidel_x_next_GS = N_Gaus_Seidel*x_current_GS + b;
    x_next_GS = M_Gaus_Seidel \ M_Gaus_Seidel_x_next_GS;
    %collecting the errors at each iteration
    e_i_gs = x_next_GS - x_current_GS;
    errors_gs = [errors_gs, e_i_gs];
    x_current_GS = x_next_GS;

    %errors_2_norms
    two_norms_of_errors_gs = [two_norms_of_errors_gs, norm(
       errors_gs(:, i), 2)];
    true_error_approx_gs = [true_error_approx_gs,
       spec_radius_M_inv_N_GS^i*norm(e0,2)];

    termination_criteria_gs = 0; %termination criteria is same
       as "have_we_found_one_over_point_5" and if we have not
       the first
    %break will stop the first inner loop the second will stop
       the outer loop
```

```
    p = size(e_i_gs);
    for j = 1:p(:,1)
        if  abs(e_i_gs(j)/x_current_GS(j)) >=0.0005
           termination_criteria = 1;
           break;
        end
    end
    if termination_criteria == 0
       break
    end
    %errors is a 10 by "i" matrix whose column space is the span
        of all error vectors at each iteration.

end

%implementing Jacobi Method:
% Applying Jacobi iteration
M_jac = diag(A);
M_jac = diag(M_jac);
N_jac = M_jac-A;

%M diagonal, thus M^-1 would be reciprocal of diagonals inv(M)
    is cheap
%operation
M_inv_jac = M_jac\eye(10);
%finding the spectral radius of M_inv*N
spec_radius_M_inv_N_jac = max(abs(eig(M_inv_jac*N_jac)));
x_list_jac = [];
errors_jac = [];
two_norms_of_errors_jac= [];
true_error_approx_jac = [];
e0 = x_true - x0;
x_current_jac = x0;

k = 20;
for i = 1: k
    M_times_x_next = N_jac*x_current_jac + b;
    x_next_jac = M_jac \ M_times_x_next;
    %collecting the errors at each iteration
    e_i_jac = x_next_jac - x_current_jac;

    %collecting error vectors
    errors_jac = [errors_jac, e_i_jac];
    x_current_jac = x_next_jac;

    %errors_2_norms
    two_norms_of_errors_jac = [two_norms_of_errors_jac, norm(
       errors_jac(:, i), 2)];

    %from the
    true_error_approx_jac = [true_error_approx_jac,
       spec_radius_M_inv_N_jac^i*norm(e0,2)];

    termination_criteria = 0; %termination criteria is same as
       "have_we_found_one_over_point_5" and if we have not the
        first
    %break will stop the first inner loop the second will stop
       the outer loop
    p = size(e_i_jac);
```

```
    for j = 1:p(:,1)
        if  abs(e_i_jac(j)/x_current_jac(j)) >=0.0005
            termination_criteria = 1;
           break;
        end
    end
    if termination_criteria == 0
       break
    end
    %errors is a 10 by "i" matrix whose column space is the
       span of all error vectors at each iteration.

end

figure;
semilogy(w, spectral_radius_H_w, '*');
title('Spectral Radius of H(w) vs. w');
grid on;

figure;
semilogy(two_norms_of_errors_jac);
hold on;
semilogy(two_norms_of_errors_gs);
semilogy(true_error_SOR);
legend('Jacobi Iteration Method', 'Gaus Seidel Iteration', 'SOR
    Method');
ylabel('Error');
xlabel('iteration index');
grid on;
```

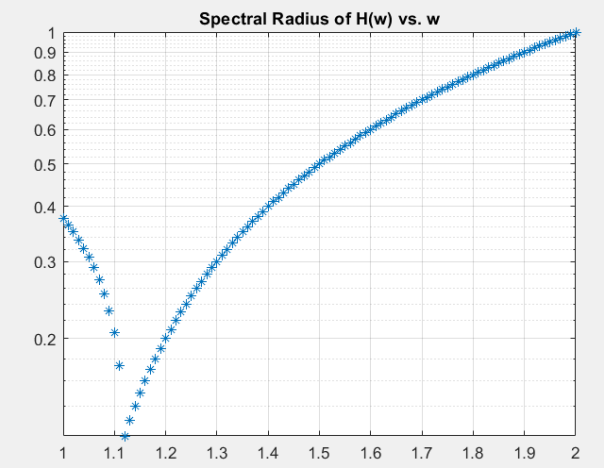


Figure 4: Spectral Radius of H(w) vs. w

According to our results plots, the optimal relaxation parameter is $w = 1.12$. Further, if we observe error plots, at any iteration index, two norm of error in error in Jacobi Method dominates the error in both Gaus-Seidel and SOR Methods. Further, at every iteration, two norm of the Gaus-Seidel error also dominates SOR method. Therefore we consider SOR Method to be more optimal as it gives us the least error out of the three iteration methods.

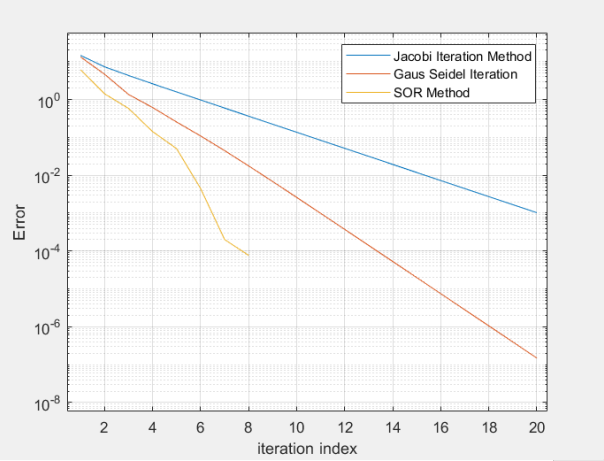


Figure 5: Error Plots: Jacobi vs. Gaus Seidel vs. SOR

4 Vector Acceleration

One can attempt to accelerate convergence of the basic iteration in a variety of ways. One approach is to rewrite the basic iteration:

$$\begin{aligned}\mathbf{x}_{k+1} &= \mathbf{M}^{-1}\mathbf{N}\mathbf{x}_k + \mathbf{M}^{-1}\mathbf{b} \\ &= \mathbf{x}_k + \mathbf{M}^{-1}\left(\mathbf{b} - \mathbf{A}\mathbf{x}_k\right) \\ &= \mathbf{x}_k + \mathbf{M}^{-1}\mathbf{r}_k,\end{aligned}$$

where $\mathbf{r}_k = \mathbf{b} - \mathbf{A}\mathbf{x}_k$ is the residual associated with $\mathbf{x}_k$. We will try to accelerate convergence by inserting a parameter " τ " and then choosing a value for τ so as to get the fastest rate. Consider then the modified iteration

$$\mathbf{x}_{k+1} = \mathbf{x}_k + \tau\mathbf{M}^{-1}\mathbf{r}_k.$$

Problem 4.1

Assuming that the basic iteration matrix $\mathbf{M}^{-1}\ \mathbf{N}$ has real eigenvalues, find the optimum value of τ that produces the fastest rate of convergence in the above iteration and express it in terms of the largest and smallest eigenvalues of $\mathbf{M}^{-1}\mathbf{N}$. The original unaccelerated iteration (using $\tau = 1$) need not be convergent even (!), but what restrictions do need to be put on the eigenvalues of $\mathbf{M}^{-1}\mathbf{N}$ (assuming that they're real) for the optimal " τ " iteration to be convergent?

Solution:

Let $\lambda_1, \lambda_2, ..., \lambda_n$ be eigenvalues of $\mathbf{M^{-1}N}$. Assuming $\{\lambda_i\}_{i=1}^n \in \mathcal{R}$ [5]

$$\begin{aligned}\mathbf{e_k} &= \mathbf{x_k} - \mathbf{x} \\ &= \mathbf{x_{k-1}} + \tau\mathbf{M^{-1}r_{k-1}} - \mathbf{x} \\ &= \mathbf{x_{k-1}} + \tau\mathbf{M^{-1}}(\mathbf{b} - \mathbf{Ax_{k-1}}) - \mathbf{x} \\ &= \mathbf{x_{k-1}} + \tau\mathbf{M^{-1}}(\mathbf{b} - \mathbf{Ax_{k-1}}) - (\mathbf{x} + \tau\mathbf{M^{-1}}(\mathbf{b} - \mathbf{Ax})) \\ &= \mathbf{x_{k-1}} + \tau\mathbf{M^{-1}b} - \tau\mathbf{M^{-1}Ax_{k-1}} - \mathbf{x} - \tau\mathbf{M^{-1}b} + \tau\mathbf{M^{-1}b} + \tau\mathbf{M^{-1}bAx} \\ &= \mathbf{x_{k-1}} - \mathbf{x} - \tau\mathbf{M^{-1}Ax_{k-1}} + \tau\mathbf{M^{-1}bAx} \\ &= \mathbf{e_{k-1}} - \tau\mathbf{M^{-1}Ae_{k-1}} \\ &= \mathbf{e_{k-1}}(\mathbf{I} - \tau\mathbf{M^{-1}A}) \\ &= (\mathbf{I} - \tau\mathbf{M^{-1}A})^{\mathbf{k}}\mathbf{e_0}\end{aligned}$$

[5] $\mathcal{R}$ - real numbers

Using $\mathbf{A} = \mathbf{M} - \mathbf{N}$ in the above result yields:

$$\begin{aligned}\mathbf{e_k} &= (\mathbf{I} - \tau\mathbf{M^{-1}}(\mathbf{M} - \mathbf{N}))^{\mathbf{k}}\mathbf{e_0} \\ &= (\mathbf{I} - \tau\mathbf{M^{-1}}\mathbf{M} + \tau\mathbf{M^{-1}}\mathbf{N})^{\mathbf{k}}\mathbf{e_0} \\ &= (\mathbf{I} - \tau(\mathbf{I} - \mathbf{M^{-1}}\mathbf{N}))^{\mathbf{k}}\mathbf{e_0}\end{aligned}$$

According to our previous results, the spectral radius of $\mathbf{I} - \tau(\mathbf{I} - \mathbf{M^{-1}}\mathbf{N})^{\mathbf{k}}$ must be less than one, for our iteration to converge. Let $\rho(\mathbf{I} - \tau(\mathbf{I} - \mathbf{M^{-1}}\mathbf{N})^{\mathbf{k}})$ be the spectral radius of $\mathbf{I} - \tau(\mathbf{I} - \mathbf{M^{-1}}\mathbf{N})^{\mathbf{k}}$. Then the eigenvalues of $\mathbf{I} - \tau(\mathbf{I} - \mathbf{M^{-1}}\mathbf{N})^{\mathbf{k}}$ can be expressed in terms of the eigenvalues of $\mathbf{M^{-1}}\mathbf{N}$.

$$\rho(\mathbf{I} - \tau(\mathbf{I} - \mathbf{M^{-1}}\mathbf{N})^{\mathbf{k}}) = \max|1 - \tau(1 - \lambda_i)| < 1$$

Let $\mathbf{B} := \mathbf{I} - \tau(\mathbf{I} - \mathbf{M^{-1}}\mathbf{N})^{\mathbf{k}}$. Let us denote the eigenvalues of $\mathbf{B}$ by $\mu_1, ...\mu_n$. As we established $\mu_i = 1 - \tau(1 - \lambda_i)$.

Notice that, for each τ we have n eigenvalues of $\mathbf{B}$. So each τ is associated with its spectral radius $\max|1 - \tau(1 - \lambda_i)|$

Without loss of generality, assume that $\lambda_1 \geq \lambda_2 \geq ... \geq \lambda_n > 0$ are in the spectrum of $\mathbf{M^{-1}}\mathbf{N}$ ($\sigma(\mathbf{M^{-1}}\mathbf{N})$). The function $1 - \tau(1 - \lambda_i)$ is monotone, (linear affine) function. [6], meaning the absolute value would bring the negative part of this function above the spectrum axis - $\sigma(\mathbf{M^{-1}}\mathbf{N})$, which is where $\{\lambda_i\}_i^n$ live. This boils down to the fact that, we are exploring our monotone function $1-\tau(1-\lambda_i)$ on a compact interval $[\lambda_1, \lambda_n]$, where λ_1, λ_n are the smallest and the largest eigenvalues of $\mathbf{M^{-1}}\mathbf{N}$. Of course, our underlying assumption is that the largest and the smallest eigenvalues in the spectrum of $\mathbf{M^{-1}}\mathbf{N}$ are finite.

This claim can be solidified also from the fact that, $1 - \tau(1 - \lambda_i)$ is a monotone, continuous function on a closed and bounded (which in euclidean setting means compact) interval $[\lambda_1, \lambda_n]$. This means that the max and min can be achieved only at the endpoints. Therefore the only elements in the spectrum of $\mathbf{M^{-1}}\mathbf{N}$ that matter are the largest and smallest elements.

Why do we care about the lagest and the smallest value of $1 - \tau(1 - \lambda_1)$ **?**:

According to our previous results, we care that the spectral radius of $||\mathbf{I} - \tau(\mathbf{I} - \mathbf{M^{-1}}\mathbf{N})||$, which is $|1 - \tau(1 - \lambda_i)|$, is less than one.[7] Therefore, we need to make sure that the maximum possible value of $|1 - \tau(1 - \lambda_i)|$ is less than 1. But since $1 - \tau(1 - \lambda_i)$ is monotone and continuous on a compact support $[\lambda_1, \lambda_n]$, the function $1 - \tau(1 - \lambda_i)$ will reach its max and min at the end points of $[\lambda_1, \lambda_n]$, which implies that the function $|1 - \tau(1 - \lambda_i)|$ will reach maximum at the end points of $[\lambda_1, \lambda_n]$.

Then the relevant question becomes, what does $max\{|1 - \tau(1 - \lambda_1|, |1 - \tau(1 - \lambda_1|\}$ equal to?

According to our previous assumption, λ_1 is the dominant eigenvalue of $\mathbf{M^{-1}}\mathbf{N}$, and the minimum eigenvalue in the spectrum of $\mathbf{M^{-1}}\mathbf{N}$ is λ_n. This yields the following:

Conclusion (*): The only points that matter in the compact domain[8] $[\lambda_1, \lambda_n]$ **are** λ_1 **and** λ_n**. The other points in the spectrum of** $M^{-1}N$ **do not matter, since they do not correspond to the global extrema points of our subject of interest** $1 - \tau(1 - \lambda_i)$ **on** $[\lambda_1, \lambda_n]$**.**

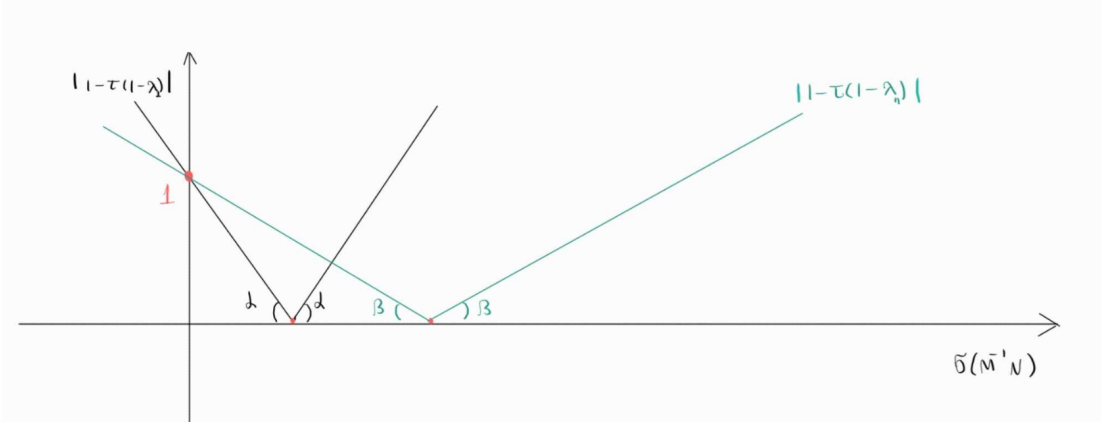
Now that we have established the conclusion above, the quantity that we would like to explore is $\min_\tau(\max\{|1 - \tau(1 - \lambda_1)|, |1 - \tau(1 - \lambda_n)|\})$.

To gain better visual intuition, we can take a look at the following plots:

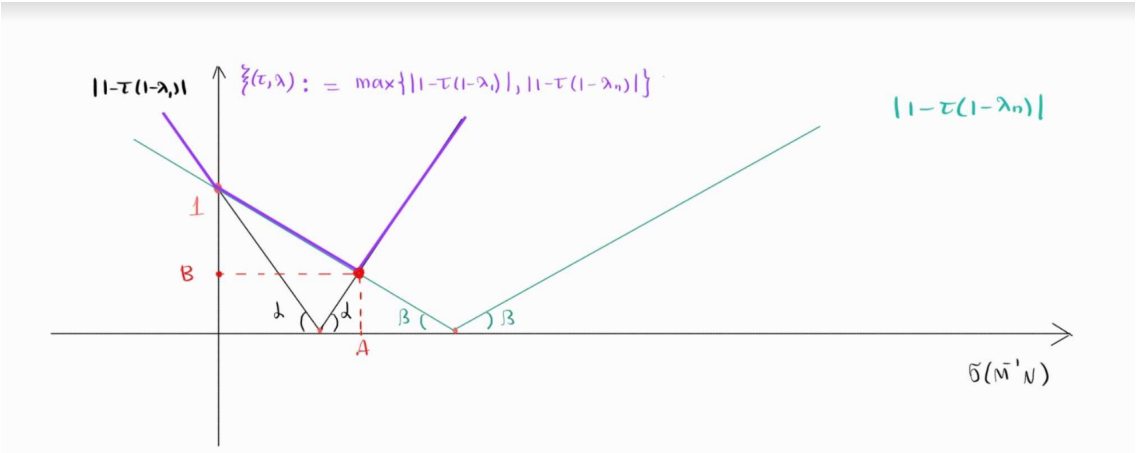
[6] This is because we assume we have information about $\{\lambda_i\}_{i=1}^{i=n}$ and the only parameter in this equation is τ. In other words, each eigenvalue of $\mathbf{M^{-1}}\mathbf{N}$, uniquely determines τ, at which point we are dealing with a linear function whose only varying parameter is τ

[7] Otherwise, as previously showed, our method will not converge.

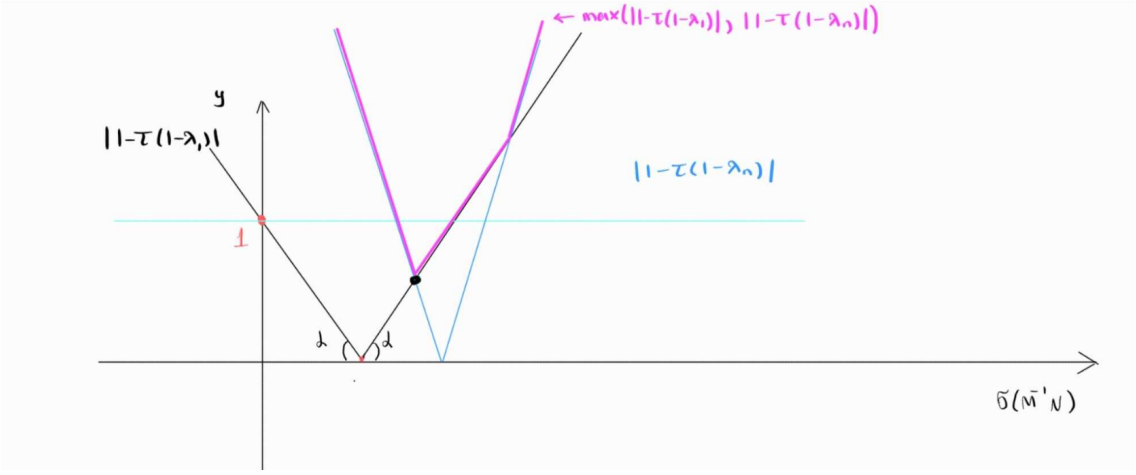
[8] The assumption is that $[\lambda_1, \lambda_n]$ is closed and bounded, in other words, the largest and the smallest eigenvalues of $M^{-1}N$ are finite



The following plot will give us a better understanding of how $max\{|1-\tau(1-\lambda_1)|, |1-\tau(1-\lambda_1)|\}$ looks like:



The minima can be achieved at (A,B) or at (0,1) when the two graphs intersect.
We cannot have the following scenario:



This is due to the fact that the slope of $|1-\tau(1-\lambda_n)|$ cannot be greater than 1, since this would mean that $|1-\tau(1-\lambda_n)|$ would intersect y axis above 1.

These arguments allow us to deduce that to obtain the optimal tau, we need to solve the following equation:

$$\begin{aligned}
|1-\tau(1-\lambda_1)| = |1-\tau(1-\lambda_n)| &\iff \\
(1-\tau(1-\lambda_1))^2 - (1-\tau(1-\lambda_n))^2 = 0 &\iff \\
\big[(1-\tau(1-\lambda_1)) - (1-\tau(1-\lambda_n))\big]\big[(1-\tau(1-\lambda_1)) + (1-\tau(1-\lambda_n))\big] = 0 &\iff \\
\big[1-\tau+\tau\lambda_1 - 1 + \tau - \tau\lambda_n\big]\big[1-\tau+\tau\lambda_1 + 1 - \tau + \tau\lambda_n\big] = 0 &\iff \\
\text{Case 1: } \tau(\lambda_1-\lambda_n) = 0 \iff \lambda_1 = \lambda_n & \\
\text{Case 2: } \tau(\lambda_1+\lambda_n-2) = 2 \iff \tau = \frac{2}{\lambda_1+\lambda_n-2} &
\end{aligned}$$

However, we have already established that middle points of $[\lambda_1, \lambda_n]$ do not matter due to the fact that the spectral radius will be solely determined by $max\{|1-\tau(1-\lambda_1)|, |1-\tau(1-\lambda_n)|\}$, from already proven **Conclusion ***, and $\lambda_1 = \lambda_2$ at the corner of $|1-\tau(1-\lambda_{\lambda_i})|$, which, due to the monotonicity of $\lambda_1 = \lambda_2$ at the corner of $|1-\tau(1-\lambda_{\lambda_i})$ is always less than or equal to the $\min\{|1-\tau(1-\lambda_1)|, |1-\tau(1-\lambda_n)|\}$, and even more so to the $\max\{|1-\tau(1-\lambda_1)|, |1-\tau(1-\lambda_n)|\}$.

Thus, the only case that is valid, is **case 2**, uniquely determining the optimal

$$\tau = \frac{2}{\lambda_1+\lambda_n-2}$$

.

The idea from earlier this section can be extended by using a different parameter at each iteration step, which is

$$\mathbf{x}_{k+1} = \mathbf{x}_k + \tau_{k+1}\mathbf{M}^{-1}\mathbf{r}_k$$

This is a nonstationary iteration since the mapping from $\mathbf{x}_k$ to $\mathbf{x}_{k+1}$ changes from step to step, which is often called Richardson iteration in general or more particularly as Chebyshev iteration if the $\{\tau_k\}$ are chosen as reciprocals of Chebyshev points.

Problem 4.2

If $\mathbf{x}_0 = 0$, show that Richardson iteration produces a sequence of iterates that satisfy

$$\mathbf{x}_k \in \operatorname{span}\left\{\mathbf{M}^{-1}\mathbf{r}_0, \mathbf{M}^{-1}\mathbf{r}_1, \ldots, \mathbf{M}^{-1}\mathbf{r}_{k-1}\right\} \tag{1}$$

and

$$\mathbf{x}_k \in \operatorname{span}\left\{\widetilde{\mathbf{b}}, \widetilde{\mathbf{A}}\widetilde{\mathbf{b}}, \widetilde{\mathbf{A}}^2\widetilde{\mathbf{b}}, \ldots, \widetilde{\mathbf{A}}^{k-1}\widetilde{\mathbf{b}}\right\} \tag{2}$$

where $\widetilde{\mathbf{A}} = \mathbf{M}^{-1}\mathbf{A}$ is the preconditioned coefficient matrix and $\widetilde{\mathbf{b}} = \mathbf{M}^{-1}\mathbf{b}$.

Proof. Assume the Richardson iteration is $\mathbf{x}_{k+1} = \mathbf{x}_k + \tau_{k+1}\mathbf{M}^{-1}\mathbf{r}_k$ where $\mathbf{r}_k = \mathbf{b} - \mathbf{A}\mathbf{x}_k$ is the residual associated with $\mathbf{x}_k$ and suppose $\mathbf{x_0} = 0$. We first want to show (1) first by induction.
Base step:
when $\mathbf{k} = 0$, $\mathbf{x_1} = \mathbf{x_0} + \tau_1\mathbf{M^{-1}r_0} = \tau_1\mathbf{M^{-1}r_0}$. Since τ is a scalar, so $\mathbf{x_1} \in \operatorname{span}\left\{\mathbf{M^{-1}r_0}\right\}$.
Inductive Step:
Let $\mathbf{k} \geq 0$ be an arbitrary integer and suppose $\mathbf{x}_k \in \operatorname{span}\left\{\mathbf{M}^{-1}\mathbf{r}_0, \mathbf{M}^{-1}\mathbf{r}_1, \ldots, \mathbf{M}^{-1}\mathbf{r}_{k-1}\right\}$. Then by the definition of span, we know that $\mathbf{x_k} = \tau_1\mathbf{M^{-1}r_0} + \tau_2\mathbf{M^{-1}r_1} + \ldots + \tau_k\mathbf{M^{-1}r_{k-1}}$. By Richardson iteration, we have

$$\begin{aligned}
x_{k+1} &= \mathbf{x}_k + \tau_{k+1}\mathbf{M}^{-1}\mathbf{r}_k \\
&= \tau_1\mathbf{M^{-1}r_0} + \tau_2\mathbf{M^{-1}r_1} + \ldots + \tau_k\mathbf{M^{-1}r_{k-1}} + \tau_{k+1}\mathbf{M}^{-1}\mathbf{r}_k \quad \text{by inductive hypothesis}
\end{aligned}$$

Thus, since τ with different index is just scalar, so we got

$$\mathbf{x}_{k+1} \in \operatorname{span}\left\{\mathbf{M}^{-1}\mathbf{r}_0, \mathbf{M}^{-1}\mathbf{r}_1, \ldots, \mathbf{M}^{-1}\mathbf{r}_{k-1}, \mathbf{M^{-1}r_k}\right\}$$

by induction. So, we can conclude that $\mathbf{x}_k \in \operatorname{span}\left\{\mathbf{M}^{-1}\mathbf{r}_0, \mathbf{M}^{-1}\mathbf{r}_1, \ldots, \mathbf{M}^{-1}\mathbf{r}_{k-1}\right\}$ is true for all $\mathbf{k} \geq 0$. Now, we want to show (2), also by induction. Additionally, assume $\widetilde{\mathbf{A}} = \mathbf{M}^{-1}\mathbf{A}$ is the

preconditioned coefficient matrix and $\widetilde{\mathbf{b}} = \mathbf{M}^{-1}\mathbf{b}$.
Base step:
when $\mathbf{k} = 0$, $\mathbf{x_1} = \mathbf{x_0} + \tau_1\mathbf{M^{-1}r_0} = \tau_1\mathbf{M^{-1}}(\mathbf{b} - \mathbf{Ax_0}) = \tau_1\mathbf{M^{-1}b} = \tau_1\widetilde{\mathbf{b}}$. Since τ is a scalar, so $\mathbf{x_1} \in \text{span}\{\widetilde{\mathbf{b}}\}$.
Inductive step:
Let $\mathbf{k} \geq 0$ be an arbitrary integer and suppose $\mathbf{x}_k \in \text{span}\left\{\widetilde{\mathbf{b}}, \widetilde{\mathbf{A}}\widetilde{\mathbf{b}}, \widetilde{\mathbf{A}}^2\widetilde{\mathbf{b}}, \ldots, \widetilde{\mathbf{A}}^{k-1}\widetilde{\mathbf{b}}\right\}$. Then by the definition of span, we know that $\mathbf{x_k} = \tau_1\widetilde{\mathbf{b}} + \tau_2\widetilde{\mathbf{A}}\widetilde{\mathbf{b}} + \tau_3\widetilde{\mathbf{A}}^2\widetilde{\mathbf{b}} + \ldots + \tau_k\widetilde{\mathbf{A}}^{k-1}\widetilde{\mathbf{b}}$. By Richardson iteration, we have that

$$\begin{aligned}
\mathbf{x_{k+1}} &= \mathbf{x_k} + \tau_{k+1}\mathbf{M^{-1}r_k} \\
&= \tau_1\widetilde{\mathbf{b}} + \tau_2\widetilde{\mathbf{A}}\widetilde{\mathbf{b}} + \tau_3\widetilde{\mathbf{A}}^2\widetilde{\mathbf{b}} + \ldots + \tau_k\widetilde{\mathbf{A}}^{k-1}\widetilde{\mathbf{b}} + \tau_{k+1}\mathbf{M^{-1}r_k} \\
&= \tau_1\widetilde{\mathbf{b}} + \tau_2\widetilde{\mathbf{A}}\widetilde{\mathbf{b}} + \tau_3\widetilde{\mathbf{A}}^2\widetilde{\mathbf{b}} + \ldots + \tau_k\widetilde{\mathbf{A}}^{k-1}\widetilde{\mathbf{b}} + \tau_{k+1}\mathbf{M^{-1}}(\mathbf{b} - \mathbf{Ax_k}) \\
&= \mathbf{L} + \tau_{k+1}\mathbf{M^{-1}}(\mathbf{b} - \mathbf{A}(\tau_1\widetilde{\mathbf{b}} + \tau_2\widetilde{\mathbf{A}}\widetilde{\mathbf{b}} + \tau_3\widetilde{\mathbf{A}}^2\widetilde{\mathbf{b}} + \ldots + \tau_k\widetilde{\mathbf{A}}^{k-1}\widetilde{\mathbf{b}})) \\
&= \mathbf{L} + \tau_{k+1}\mathbf{M^{-1}b} - \tau_{k+1}\mathbf{M^{-1}A}\tau_1\widetilde{\mathbf{b}} - \ldots - \tau_{k+1}\mathbf{M^{-1}A}\tau_k\widetilde{\mathbf{A}}^{k-1}\widetilde{\mathbf{b}} \\
&= \mathbf{L} + \tau_{k+1}\widetilde{\mathbf{b}} - \tau_{k+1}\tau_1\widetilde{\mathbf{A}}\widetilde{\mathbf{b}} - \ldots - \tau_{k+1}\tau_k\widetilde{\mathbf{A}}\widetilde{\mathbf{A}}^{k-1}\widetilde{\mathbf{b}} \\
&= \mathbf{L} + \tau_{k+1}\widetilde{\mathbf{b}} - \tau_{k+1}\tau_1\widetilde{\mathbf{A}}\widetilde{\mathbf{b}} - \ldots - \tau_{k+1}\tau_k\widetilde{\mathbf{A}}^{k}\widetilde{\mathbf{b}} \\
&= \tau_1\widetilde{\mathbf{b}} + \tau_2\widetilde{\mathbf{A}}\widetilde{\mathbf{b}} + \tau_3\widetilde{\mathbf{A}}^2\widetilde{\mathbf{b}} + \ldots + \tau_k\widetilde{\mathbf{A}}^{k-1}\widetilde{\mathbf{b}} + +\tau_{k+1}\widetilde{\mathbf{b}} - \tau_{k+1}\tau_1\widetilde{\mathbf{A}}\widetilde{\mathbf{b}} - \ldots - \tau_{k+1}\tau_k\widetilde{\mathbf{A}}^{k}\widetilde{\mathbf{b}} \\
&= (\tau_1 + \tau_{k+1})\widetilde{\mathbf{b}} + (\tau_2 - \tau_{k+1}\tau_1)\widetilde{\mathbf{A}}\widetilde{\mathbf{b}} + \ldots + (-\tau_{k+1}\tau_k)\widetilde{\mathbf{A}}^{k}\widetilde{\mathbf{b}}
\end{aligned}$$

Since τ with any index is scalar, so scalars are closed under multiplication and addition, we see $\mathbf{x}_{k+1} \in \text{span}\left\{\widetilde{\mathbf{b}}, \widetilde{\mathbf{A}}\widetilde{\mathbf{b}}, \widetilde{\mathbf{A}}^2\widetilde{\mathbf{b}}, \ldots, \widetilde{\mathbf{A}}^{k}\widetilde{\mathbf{b}}\right\}$. Thus, we can conclude that $\mathbf{x}_k$ is in the span as we want. □

Problem 4.3

Show that Richardson iteration produces a sequence of residuals $\{\mathbf{r}_0, \mathbf{r}_1, \ldots, \mathbf{r}_k, \ldots\}$ that satisfies

$$\mathbf{r}_{k+1} = \left(\mathbf{I} - \tau_{k+1}\mathbf{AM}^{-1}\right)\mathbf{r}_k \tag{3}$$

Find the value of the parameter τ_{k+1} that minimizes the 2-norm of residual vector in the next step; that is, so that

$$\|\mathbf{r}_{k+1}\|_2 = \min_{\tau}\left\|\left(\mathbf{I} - \tau\mathbf{AM}^{-1}\right)\mathbf{r}_k\right\|_2 \tag{4}$$

Proof. Suppose the Richardson iteration is $\mathbf{x}_{k+1} = \mathbf{x}_k + \tau_{k+1}\mathbf{M}^{-1}\mathbf{r}_k$ where $\mathbf{r}_k = \mathbf{b} - \mathbf{Ax}_k$ is the residual associated with $\mathbf{x}_k$.
First, we will prove (3). By assumption, we also have $\mathbf{r}_{k+1} = \mathbf{b} - \mathbf{Ax}_{k+1}$. Then consider

$$\begin{aligned}
\mathbf{r_{k+1}} &= \mathbf{b} - \mathbf{A}(\mathbf{x_k} + \tau_{\mathbf{k+1}}\mathbf{M^{-1}r_k}) \\
&= \mathbf{b} - \mathbf{Ax_k} - \mathbf{A}\tau_{\mathbf{k+1}}\mathbf{M^{-1}r_k} \\
&= (\mathbf{b} - \mathbf{Ax_k}) - \mathbf{A}\tau_{\mathbf{k+1}}\mathbf{M^{-1}}(\mathbf{b} - \mathbf{Ax_k}) \\
&= (\mathbf{I} - \mathbf{A}\tau_{\mathbf{k+1}}\mathbf{M^{-1}})(\mathbf{b} - \mathbf{Ax_k}) \\
&= (\mathbf{I} - \tau_{\mathbf{k+1}}\mathbf{AM^{-1}})(\mathbf{b} - \mathbf{Ax_k}) && \text{because } \tau_{k+1} \text{ is a scalar} \\
&= (\mathbf{I} - \tau_{\mathbf{k+1}}\mathbf{AM^{-1}})\mathbf{r_k}
\end{aligned}$$

Thus, we have proven (3) as desired. Now, we want to show (4). □

To find the value of the parameter τ_{k+1} that minimizes the 2-norm of the residual vector in the next step, so that

$$||\mathbf{r_{k+1}}||_2 = \min_{\tau}||(\mathbf{I} - \tau\mathbf{AM^{-1}})\mathbf{r_k}||_2,$$

we can treat this as a function of τ and instead of minimizind a two norm, for simplicity we can solve

$$||\mathbf{r_{k+1}}||_2^2 = \min_{\tau}||(\mathbf{I} - \tau\mathbf{AM^{-1}})\mathbf{r_k}||_2^2$$

Simplifying steps for the right hand side involves following calculations:

$$\begin{aligned}\min_{\tau} ||(\mathbf{I}-\tau\mathbf{A}\mathbf{M^{-1}})\mathbf{r_k}||_2^2 &= [\mathbf{r_k}-\tau\mathbf{A}\mathbf{M^{-1}}\mathbf{r_k}]^T[\mathbf{r_k}-\tau\mathbf{A}\mathbf{M^{-1}}\mathbf{r_k}] \\ &= [\mathbf{r_k^T}\mathbf{r_k}-\mathbf{r_k^T}\tau(\mathbf{A}\mathbf{M^{-1}})^\mathbf{T}][\mathbf{r_k}-\tau\mathbf{A}\mathbf{M^{-1}}\mathbf{r_k}] \\ &= \mathbf{r_k^T}\mathbf{r_k}-\tau\mathbf{r_k^T}\mathbf{A}\mathbf{M^{-1}}\mathbf{r_k}-\tau\mathbf{r_k^T}(\mathbf{A}\mathbf{M^{-1}})^\mathbf{T}\mathbf{r_k}+\tau^2\mathbf{r_k^T}(\mathbf{A}\mathbf{M^{-1}})^\mathbf{T}\mathbf{A}\mathbf{M^{-1}}\mathbf{r_k}\end{aligned}$$

Now we will minimize this function with respect to τ, which can be done by taking a derivative of this function and finding its local minima (if any). Let us define

$$f(\tau)=\mathbf{r_k^T}\mathbf{r_k}-\tau\mathbf{r_k^T}\mathbf{A}\mathbf{M^{-1}}\mathbf{r_k}-\tau\mathbf{r_k^T}(\mathbf{A}\mathbf{M^{-1}})^\mathbf{T}\mathbf{r_k}+\tau^2\mathbf{r_k^T}(\mathbf{A}\mathbf{M^{-1}})^\mathbf{T}\mathbf{A}\mathbf{M^{-1}}\mathbf{r_k}$$

. Then

$$f'(\tau)=-\mathbf{r_k^T}\mathbf{A}\mathbf{M^{-1}}\mathbf{r_k}-\mathbf{r_k^T}\mathbf{A}\mathbf{M^{-1}}\mathbf{r_k}+2\tau\mathbf{r_k^T}(\mathbf{A}\mathbf{M^{-1}})^\mathbf{T}(\mathbf{A}\mathbf{M^{-1}})\mathbf{r_k}$$

Setting this expression to gain knowledge about the critical points of $f(\tau)$ yields:

$$-\mathbf{r_k^T}\mathbf{A}\mathbf{M^{-1}}\mathbf{r_k}-\mathbf{r_k^T}\mathbf{A}\mathbf{M^{-1}}\mathbf{r_k}+2\tau\mathbf{r_k^T}(\mathbf{A}\mathbf{M^{-1}})^\mathbf{T}(\mathbf{A}\mathbf{M^{-1}})\mathbf{r_k}=0$$

From here we can express τ to obtain:

$$\begin{aligned}2\tau\mathbf{r_k^T}(\mathbf{A}\mathbf{M^{-1}})^\mathbf{T}(\mathbf{A}\mathbf{M^{-1}})\mathbf{r_k}=\mathbf{r_k^T}\mathbf{A}\mathbf{M^{-1}}\mathbf{r_k}+\mathbf{r_k^T}\mathbf{A}\mathbf{M^{-1}}\mathbf{r_k} &\iff \\ \tau=\frac{\mathbf{r_k^T}(\mathbf{A}\mathbf{M^{-1}})^\mathbf{T}\mathbf{r_k}+\mathbf{r_k^T}\mathbf{A}\mathbf{M^{-1}}\mathbf{r_k}}{2\mathbf{r_k^T}(\mathbf{A}\mathbf{M^{-1}})^\mathbf{T}\mathbf{A}\mathbf{M^{-1}}\mathbf{r_k}} &\end{aligned}$$

Since the function $f(\tau)$ is convex and it is continuous with rerpect to the parameter τ, this critical point (τ) that we have found gives us the minimizer of it.
Conclusion: The best choice of τ that minimizes the residuals that are produced at every iteration index of Richardson's iteration, is:

$$\tau=\frac{\mathbf{r_k^T}(\mathbf{A}\mathbf{M^{-1}})^\mathbf{T}\mathbf{r_k}+\mathbf{r_k^T}\mathbf{A}\mathbf{M^{-1}}\mathbf{r_k}}{2\mathbf{r_k^T}(\mathbf{A}\mathbf{M^{-1}})^\mathbf{T}\mathbf{A}\mathbf{M^{-1}}\mathbf{r_k}}.$$

Further, defining the k^{th} degree polynomial, $\phi_k(z)=\prod_{i=1}^{k}(1-\tau_i z\ )$, one can show that,

$$\mathbf{r}_k=\prod_{i=1}^{k}\left(\mathbf{I}-\tau_i\mathbf{A}\mathbf{M}^{-1}\right)\mathbf{r}_0=\phi_k\left(\mathbf{A}\mathbf{M}^{-1}\right)\mathbf{r}_0.$$

If we look at the eigenvalues of $\phi_k\left(\mathbf{A}\mathbf{M}^{-1}\right)$, we can notice that they are exactly $\phi_k\left(\gamma_j\right)$ for each eigenvalue γ_j of $\mathbf{A}\mathbf{M}^{-1}$. Thus, if $\mathbf{A}\mathbf{M}^{-1}$ is diagonalizable and $\phi_k\left(\gamma_j\right)\to 0$ as $k\to\infty$ for each eigenvalue γ_j of $\mathbf{A}\mathbf{M}^{-1}$ then $\mathbf{x}_k\to\mathbf{x}$, the exact solution.

A different plan to get a one-step resisual minimization of problem 4.2 is to instead choose the parameters $\tau_1,\tau_2,\ldots\tau_k,\ldots$ so that the polynomial $\phi_k(z)$ gets progressively smaller on (or at least near) the eigenvalues of $\mathbf{A}\mathbf{M}^{-1}$ as k increases. We need to have some knowledge about the spectrum $\mathbf{A}\mathbf{M}^{-1}$ to optimally choose parameters $\tau_1,\ldots\tau_n$. We know that

$$\mathbf{A}\mathbf{M}^{-1}=\mathbf{M}\tilde{A}\mathbf{M}^{-1}=\mathbf{M}\left(\mathbf{I}-\mathbf{M}^{-1}\mathbf{N}\right)\mathbf{M}^{-1}$$

This means that $\mathbf{A}\mathbf{M}^{-1}$ has the same eigenvalues as the preconditioned coefficient matrix $\tilde{\mathbf{A}}$, which equal to $1-\gamma_j$ for each eigenvalue γ_j of the original iteration matrix $\mathbf{M}^{-1}\mathbf{N}$. More precisely, all eigenvalues of $\mathbf{A}\mathbf{M}^{-1}$ all must be in the interval $\left[1-\rho\left(\mathbf{M}^{-1}\mathbf{N}\right),1+\rho\left(\mathbf{M}^{-1}\mathbf{N}\right)\right]$, and might be sufficient to find polynomials $\phi_k(z)$ that are "small" on this interval. Chebyshev polynomials are well-suited to this task: Chebyshev Polynomials: $T_0(x)=1,\quad T_1(x)=x,\quad T_2(x)=x^2-\frac{1}{2},\quad T_3(x)=x^3-\frac{3}{4}x\quad\ldots$

Recursive definition of Chebyshev polynomials for for k $= 4, 5, \ldots$ is $T_k(x) = xT_{k-1}(x) - T_{k-2}(x)/4$. What distinguishes Chebyshev polynomials from other polynomials is that Chebyshev polynomials have the smallest uniform size throughout the interval $[-1, 1]$ out of all monic polynomials of the same degree.

In fact, $|T_k(x)| \leq 2^{-(k-1)}$ for all $x \in [-1, 1]$ and no other monic polynomial of degree k or less satisfies this inequality. One of the observation we can make is that, if we possess knowledge about where all the eigenvalues of a matrix $\mathbf{B}$ lie in the interval $[-1, 1]$, then $\rho\left(T_k(\mathbf{B})\right) \leq 2^{-(k-1)} \to 0$ as $k \to \infty$. It will be easier for us to define the Chebyshev polynomials directly in terms of their roots:

$$T_k\left(\xi_j\right) = 0 \quad \text{for} \quad \xi_j = \cos\left(\frac{(2j+1)\pi}{2k}\right) \quad j = 0, 1 \ldots, k-1$$

and then $T_k(x) = \prod_{j=0}^{k-1}\left(x - \xi_j\right)$. Using these facts, if we want to define a polynomial $\phi_k(z)$ that is small for $z \in \left[1 - \rho\left(\mathbf{M}^{-1}\mathbf{N}\right), 1 + \rho\left(\mathbf{M}^{-1}\mathbf{N}\right)\right]$, we could define $\phi_k(z) = T_k\left(\frac{z-1}{\rho(\mathbf{M}^{-1}\mathbf{N})}\right).\phi_k(z)$ would have zeros at $1 + \rho\left(\mathbf{M}^{-1}\mathbf{N}\right)\xi_j$. Originally we definied $\phi_k(z)$ in terms of our acceleration parameters $\tau_1, \tau_2, \ldots \tau_k$, therefore we could proceed by concatenating the zero lists of consecutive $\phi_k(z)$, consequently arriving at an aggregate sequence of acceleration parameters:

$$\tau_\ell = \left(1 + \rho\left(\mathbf{M}^{-1}\mathbf{N}\right)\cos\left(\frac{(2j+1)\pi}{2k}\right)\right)^{-1}$$
$$\text{with } k = 1, 2 \ldots; \quad j = 0, 1, \ldots, k-1$$
$$\text{and } \ell = \frac{k(k-1)}{2} + j + 1$$

Problem 4.4

Apply ten steps of Richardson Iteration with $\tau_1, \ldots, \tau_{10}$ defined as above (with $k = 1$, 2, 3, 4 and $j = 0, \ldots, k-1$) on the system defined in Problem 2.3. As in Problem 2.3, plot error vs iteration index on semilog axes and discuss the results. Compare with results obtained from ten steps using for τ a sequence of one-step minimizers that you derived in Problem 4.3.

Solution. *In the following script, we are applying ten steps of Richardson Iteration with $\tau_1, \ldots, \tau_{10}$ defined*

5 Conclusion

In this term project, we discussed multiple methods to solve linear system equations by distinct iteration methods. We see that each method provides its advantages and disadvantages based on a particular set up. Moreover, understanding the properties of matrix splitting and the convergence behavior of each method discussed above is essential for selecting an efficient iterative solver for a given linear system. Additionally, tuning parameters, such as relaxation factors in SOR, can further optimize the convergence process.

While we could not dive deep in Chebyshev Iterations, we understand that it converges more rapidly than Richardson or basic iterative methods for specific types of problems. One of the major issues in general for Chebyshev iteration is the complexity in its implementation. It may not always outperform other methods for all types of systems.

Our future research ideas are to relate these concepts to the data uncertainty. Nowdays it is easy to deal with methods that introduce matrices with sufficiently big dimensions, so that computations are no longer feasible. The methods we discussed are at the core of rendering dynamical systems in a way that would allow us to take advantage of either sparcity, splitting of matrices, vector accelerations, or coming up with methods that would give us rapid convergence.

Lastly, we would like to discover other methods other than direct method and iterative method, which may be able to minimize the residual or minimize the error more to solve linear system of equations, while keeping in mind, the variance-bias tradeoff.