

University of Vienna
Faculty of Mathematics

Applied Analysis Problems

Milutin Popovic

March 13, 2022

Contents

1	Sheet 2	1
1.1	Problem 1	1
1.1.1		1
1.1.2		2
1.1.3		2
1.2	Problem 2	3
1.2.1		3
1.2.2		3
1.3	Problem 3	4
1.3.1		4
1.3.2		4
1.3.3		4
1.4	Problem 4	5
1.4.1		5
1.4.2		5
1.4.3		5
1.5	Problem 5	6
1.5.1		6
1.5.2		6

1 Sheet 2

1.1 Problem 1

1.1.1

We let $\rho(A)$ be the spectral radius of a matrix $A \in \mathbb{R}^{n \times n}$. A matrix norm $\|\cdot\|_1$ is consistent with the vector norm $\|\cdot\|_2$ if

$$\|Ax\|_2 \leq \|A\|_1 \|x\|_2 \quad (1)$$

for all $x \in \mathbb{R}^n$ and all $A \in \mathbb{R}^{n \times n}$. Indeed every matrix norm induced by a vector norm is consistent. To show this let $\|\cdot\|_M$ be a matrix norm and $\|\cdot\|_v$ be a vector norm, defined as

$$\|x\|_v = \|xv^T\|_M \quad (2)$$

for all $x \in \mathbb{R}^n$ and some $v \neq 0$, of $\mathbb{R}^{n \times n}$ and \mathbb{R}^n respectively. Then we have

$$\|Ax\|_v = \|Axv^T\| \leq \|A\|_M \|xv^T\|_M = \|A\|_M \|v\|_v \quad \square \quad (3)$$

Note that for $\mathbb{C}^{n \times n}$ and \mathbb{C}^n use $v^* \neq 0$ the conjugate transpose.

1.1.2

Now we consider a splitting of $A = D - (L + U)$, where D, L and U are defined as

$$D = \text{diag}(a_{11}, \dots, a_{nn}), \quad (4)$$

$$(L)_{ij} = \begin{cases} -(A)_{ij} & i > j \\ 0 & i \leq j \end{cases}, \quad (U)_{ij} = \begin{cases} -(A)_{ij} & i < j \\ 0 & i \geq j \end{cases}, \quad (5)$$

Then the matrix of a single Jacobi iteration method is

$$B_J = D^{-1}(L + U) \quad (6)$$

We can show that if A is strictly diagonally dominant then

$$\rho(B_J) \leq \|B_J\|_\infty < 1. \quad (7)$$

If A is strictly diagonally dominant, this means that

$$|A_{ii}| > \sum_{j \neq i}^n |A_{ij}| \quad \forall i \in \{1, \dots, n\} \quad (8)$$

$$\Leftrightarrow \sum_{j \neq i} \frac{|A_{ij}|}{|A_{ii}|} < 1. \quad (9)$$

Now let (λ, v) be an eigen-pair of B_J , then

$$B_J v = D^{-1}(L + U)v = \lambda v \quad (10)$$

$$(L + U)v = \lambda Dv. \quad (11)$$

For a chosen i this means

$$|\lambda| |A_{ii}| |v_i| = \left| -\sum_{j>i} A_{ij} v_j - \sum_{j<i} A_{ij} v_j \right| \quad (12)$$

$$\leq \sum_{j>i} |A_{ij}| |v_j| + \sum_{j<i} |A_{ij}| |v_j| \quad (13)$$

$$= \sum_{j \neq i} |A_{ij}| |v_j|. \quad (14)$$

We can choose and i such that $|v_i| \leq \|v\|_\infty$, then

$$|\lambda| |A_{ii}| |v_i| \leq \sum_{j \neq i} |A_{ij}| \|v\|_\infty \quad (15)$$

$$\Rightarrow |\lambda| |A_{ii}| \leq \sum_{j \neq i} |A_{ij}| \quad (16)$$

$$\Rightarrow |\lambda| \leq \sum_{j \neq i} \frac{|A_{ij}|}{|A_{ii}|} < 1. \quad \square \quad (17)$$

1.1.3

Next we show that the Jacobi method converges for every initial guess x^0 to the solution of the equation $Ax = b$, given that A is strictly diagonally dominant. So with any initial guess x^0 at the k -th iteration we have

$$Dx^{(k)} = (L + U)x^{(k-1)} + b \quad (18)$$

$$\Leftrightarrow x^{(k)} = D^{-1}(L + U)x^{(k-1)} + D^{-1}b \quad (19)$$

$$= B_J x^{(k-1)} + D^{-1}b \quad (20)$$

Now let x be the exact solution, then the error at the k -th iteration is

$$e^{(k)} = x - x^{(k)} = B_J (x - x^{(k-1)}) = \dots \quad (21)$$

$$= B_J^k e^{(0)} \quad (22)$$

Assume that $e^{(0)} \neq 0$, then we need $\lim_{n \rightarrow \infty} B_J^k = 0$. If A is **diagonally dominant** we have $\rho(B_J) < 1$ this means for an eigen-pair (λ, v) of B_J we have

$$\lim_{n \rightarrow \infty} B_J^k v = \lim_{n \rightarrow \infty} \lambda^k v \quad (23)$$

$$\Rightarrow \lim_{n \rightarrow \infty} \lambda^k = 0, \quad (24)$$

for all λ because $\rho(B_J) < 1$. □

1.2 Problem 2

Now consider a $A \in \mathbb{R}^{n \times n}$,

$$A = \begin{pmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{pmatrix}. \quad (25)$$

Let $A = D - (L + U)$ like in the above problem. The Gauss-Siedel method has the iteration matrix $B_G = (D - L)^{-1}U$ and the Jacobi method has the iteration matrix $B_J = D^{-1}(L + U)$.

1.2.1

We show that the spectral radii of B_J and B_G satisfy

$$\rho(B_J) = \sqrt{|\rho(B_G)|}, \quad (26)$$

by directly calculating the eigenvalues of B_J and B_G respectively. We start with B_J ,

$$\det(B_J - \lambda I) = \det \begin{pmatrix} -\lambda & -\frac{a_{12}}{a_{11}} \\ -\frac{a_{21}}{a_{22}} & -\lambda \end{pmatrix} \quad (27)$$

$$= \lambda^2 - \frac{a_{12}a_{21}}{a_{11}a_{22}} = 0 \quad (28)$$

$$\Rightarrow \lambda^2 = \frac{a_{12}a_{21}}{a_{11}a_{22}} \quad (29)$$

For B_G we have

$$\det(B_G - \lambda I) = \det \begin{pmatrix} -\lambda & -\frac{a_{12}}{a_{11}} \\ 0 & -\lambda - \frac{a_{21}a_{12}}{a_{11}a_{22}} \end{pmatrix} \quad (30)$$

$$= -\lambda \left(-\lambda - \frac{a_{21}a_{12}}{a_{11}a_{22}} \right) = 0 \quad (31)$$

$$\Rightarrow \lambda_1 = 0, \quad \lambda_2 = -\frac{a_{21}a_{12}}{a_{11}a_{22}}. \quad (32)$$

Which satisfies the above condition, remember

$$\rho(A) = \max\{|\lambda| : \lambda \text{ is eigenvalue } A\}, \quad (33)$$

especially note the absolute value. □

1.2.2

Indeed the error of the Gauss-Siedel iteration converges to 0 if $\rho(B_G) < 1$. Which is the case, because exactly then $\rho(B_G) = \rho(B_J)^2 < 1$. Where we can also conclude that the Gauss-Siedel method converges twines as fast as the Jacobi method for 2×2 matrices.

1.3 Problem 3

Let $Q \in \mathbb{R}^{n \times n}$ be the Poisson matrix, the matrix of the finite difference method on a $n \times n$ grid,

$$Q = \begin{pmatrix} 2 & -1 & & & \\ -1 & 2 & -1 & & \\ & \ddots & \ddots & \ddots & \\ & & -1 & 2 & -1 \\ & & & -1 & 2 \end{pmatrix} \quad (34)$$

1.3.1

The eigenvalues of Q lie in the interval $[0, 4]$. To show this let (λ, v) be an eigen-pair of Q , they satisfy the equation

$$Qv = \lambda v \quad (35)$$

At the k -th ($k \in \{1, \dots, n\}$) step we have $v^{(0)} = 0$, $v^{(1)} = 1$ and $v^{(n+1)} = 0$

$$-v^{(k+1)} + 2v^{(k)} - v^{(k-1)} = \lambda v^{(k)} \quad (36)$$

$$\Rightarrow v^{(k+1)} = (2 - \lambda)v^{(k)} - v^{(k-1)}, \quad (37)$$

which are the Chebyshev polynomials of the second kind, where $(2 - \lambda_k)$ satisfies

$$(2 - \lambda_k) = 2 \cdot \cos\left(\frac{k\pi}{n+1}\right) \quad (38)$$

$$\Rightarrow \lambda_k = 2 \left(1 - \cos\left(\frac{k\pi}{n+1}\right)\right) \quad (39)$$

$$= 4 \cdot \sin^2\left(\frac{k\pi}{n+1}\right), \quad (40)$$

thereby we can conclude that $\lambda_k \in [0, 4] \quad \forall k \in \{1, \dots, n\}$. □

1.3.2

1.3.3

In this section we write a Python script that returns the matrix Q given n and for $b = (1, \dots, 1)^T$ $n = 20$ implements the Gauss-Siedel method for to solve $Qx = b$ for x .

```
[53]: def gauss_siedel(A, b, k):
    n, m = Q.shape
    D = np.reshape([Q[i][j] if i==j else 0 for i in range(n) for j in
    range(n)], (n, n))
    L = np.reshape([Q[i][j] if i>j else 0 for i in range(n) for j in range(n)],
    (n, n))
    U = np.reshape([Q[i][j] if i<j else 0 for i in range(n) for j in range(n)],
    (n, n))

    x = np.random.rand(n)
    for i in range(k):
        x = np.linalg.inv(D)@(b - (L + U)@x)
    return x

def poisson_mat(n, m=None):
    return 2 * np.eye(n, m) + (-1) * np.eye(n, m, k=1) + (-1) * np.eye(n, m,
    k=-1)

# test
for n in range(5, 20):
    Q = poisson_mat(n)
    b = np.ones(n)
    x = gauss_siedel(Q, b, k=1000)
    np.testing.assert_allclose(Q@x, b, rtol=1e-5, err_msg=f'GS failed at dim {n}')
```

1.4 Problem 4

Now let $P_n \in \mathbb{R}^{n \times n}$ be the matrix

$$P_n = \begin{pmatrix} 2 & -1 & & & -1 \\ -1 & 2 & -1 & & \\ & \ddots & \ddots & \ddots & \\ & & -1 & 2 & -1 \\ -1 & & & -1 & 2 \end{pmatrix} \quad (41)$$

1.4.1

We can show that all the eigenvalues of P_n are in $[0, 4]$ because P_n is the finite difference/laplacian matrix for periodic boundary conditions and can be diagonalized by a DFT. So let (λ, v) be the eigen-pair of P_n , which satisfy the equation

$$P_n v = \lambda v \quad (42)$$

This is the standard Poisson with periodic boundary conditions, with the eigenvector $v_j = \omega^{jk} = e^{2\pi i \frac{jk}{n}}$ for a $j \in \{1, \dots, n\}$.

$$(P_n v)_j = 2\omega^{jk} - \omega^{(j-1)k} - \omega^{(j+1)k} \quad (43)$$

$$= \omega^{jk}(2 - \omega^{-k} - \omega^k) \quad (44)$$

$$= \omega^{jk}(2 - 2\cos\left(\frac{2\pi k}{n}\right)) \quad (45)$$

$$= 4\sin^2\left(\frac{2\pi k}{n}\right)\omega^{jk} = \lambda_k v_j^k, \quad (46)$$

thereby we can conclude that $\lambda_k \in [0, 4]$ for all k .

1.4.2

Because P_n is a real, symmetric, circulant matrix the orthogonal components of the eigenvalues are also eigenvectors, i.e. $\text{Re}(v)$ and $\text{Im}(v)$. We may conclude this by pure calculation. In the j -th component we have k eigenvalues

$$(P_n \text{Re}(v^k))_j = 2\text{Re}(\omega^{jk}) - \text{Re}(\omega^{(j-1)k}) - \text{Re}(\omega^{(j+1)k}) \quad (47)$$

$$= \omega^{jk} + \omega^{-jk} - \frac{1}{2}\omega^{(j-1)k} - \frac{1}{2}\omega^{-(j-1)k} - \frac{1}{2}\omega^{(j+1)k} - \frac{1}{2}\omega^{-(j+1)k} \quad (48)$$

$$= (\omega^{jk} + \omega^{-jk}) - \frac{1}{2}(\omega^{jk}(\omega^k + \omega^{-k}) + \omega^{-jk}(\omega^k + \omega^{-k})) \quad (49)$$

$$= \frac{1}{2}(\omega^{jk} + \omega^{-jk})(2 - (\omega^k + \omega^{-k})) \quad (50)$$

$$= \text{Re}(\omega^{jk})\left(2 - 2\cos\left(\frac{2\pi k}{n}\right)\right) \quad (51)$$

$$= 4\sin^2\left(2\pi \frac{k}{n}\right)\text{Re}(\omega^{jk}) = \lambda_k \text{Re}(v_j^k). \quad (52)$$

1.4.3

We define the quantity $m(n) = \min\{|\lambda| : \lambda \text{ eigenvalue of } P_n\}$. We can show that the quantity converges to 0 as n goes to infinity by calculating for a k that minimises λ_k which is for a $k \neq 0$.

$$\lim_{n \rightarrow \infty} m(n) = \lim_{n \rightarrow \infty} \min_{k \in \{1, \dots, n\}} \{|\lambda_k(n)|\} = \lim_{n \rightarrow \infty} 4\sin^2\left(2\pi \frac{k}{n}\right) \quad (53)$$

$$= \lim_{n \rightarrow \infty} 4 \cdot \left(\frac{x^2}{n^2} + \frac{x^4}{n^4} + O\left(\frac{1}{n^6}\right)\right) = 0, \quad (54)$$

where $x = 2\pi k$.

1.5 Problem 5

Let Q be like in Problem 3. And split Q as

$$Q = D - N, \quad (55)$$

where D consists of diagonal entries of Q . For $p \in \mathbb{N}$ let C_p be the Neumann polynomial preconditioner, defined as

$$C_p = D^{-1} \sum_{k=0}^p (ND^{-1})^k \quad (56)$$

1.5.1

1.5.2

The following is a Python script, that takes n, p as an Input and returns C_p furthermore calculates the spectral condition number of the matrix $C_p Q$

```
[60]: def neumann_polynomial_preconditioner(n, p):
    Q = poisson_mat(n)
    D = np.reshape([Q[i][j] if i==j else 0 for i in range(n) for j in
range(n)], (n, n))
    N = Q - D
    C_p = np.zeros([n, n])
    for k in range(p+1):
        C_p += np.linalg.matrix_power(N @ np.linalg.inv(D), k)
    return np.linalg.inv(D) @ C_p

n = 20
Q = poisson_mat(n)
P = np.arange(1, 10)
cond_2 = []
for p in P:
    C_p = neumann_polynomial_preconditioner(n, p)
    cond_2.append(np.linalg.cond(C_p @ Q, p=2))
    print(p, cond_2[p-1], sep='\t')

plt.figure(figsize=[7, 4])
plt.scatter(P, cond_2)
print("Max. and Min. Singular value are far apart from each other for uneven
p")
```

```
1      31706.88589269506
2      534.147900485917
3      31706.885892666996
4      890.0968016459952
5      31706.885892699214
6      1245.8213126008322
7      31706.885892666185
8      1601.2319940002324
9      31706.88589270954
```

Max. and Min. Singular value are far apart from each other for uneven p

