Unit 1 - Sem 4 - 22MAT230

Mathematics for Computing 4

Dr Sunil Kumar S and Prof K P Soman

School of Artificial Intelligence

Amrita Vishwa Vidyapeetham

If you find any mistakes or have any comments to share,

I would be grateful to receive them at s_sunilkumar@cb.amrita.edu

https://github.com/mfcpjt/MFC4_22MAT230

```
clearvars
clear all
ready = true;
PUBLISH = ready;
```

Syllabus

Unit 1

Linear Algebra-4

Special Matrices: Fourier Transform, discrete and Continuous, Shift matrices and Circulant matrices, The Kronecker product, Toeplitz matrices and shift invariant filters, Graphs and Laplacians and Kirchhoff's laws, Clustering by spectral methods and K-means, Completing rank one matrices, The Orthogonal Procrustes Problem, Distance matrices.

Unit 2

Calculus-4

Optimization methods for sparsity: Split algorithm for L2+ L1, Split algorithm for L1 optimization, Augmented Lagrangian, ADMM, ADMM for LP and QP, Matrix splitting and Proximal algorithms, Compressed sensing, and Matrix Completion.

Optimization methods for Neural Networks: Gradient Descent, Stochastic gradient descent, and ADAM (adaptive methods), Loss function and learning function.

Unit 3

Probability and statistics - 4

Basics of statistical estimation theory and testing of hypothesis.

Component		**Weightage %**
Internal (70)	Weekly Tests	30
	Lab Experiments	10
	Mid Project Review	10
	Final Project Review	20
External (30)	End Semester Exam-Written	20
	End Semester - Coding	10

22MAT230 - MFC 4											
	1	2	3		4	5	6	7	8	9	10
Mon		D	D	break				Lunch		C	C
Tue		C	C			D	D		Evalify		
Wed											
Thu		D	D			C	C				
Fri											

Table of Contents

Eigen Value Problem

If

$A\mathbf{x} = \lambda\mathbf{x}$

we call $\mathbf{x}$ as the eigen vector of A corresponding to the eigen value λ.

Then we have

$A^2\mathbf{x} = AA\mathbf{x} = \lambda A\mathbf{x} = \lambda^2\mathbf{x}$

In general

$A^n\mathbf{x} = \lambda^n\mathbf{x}$

All powers of A have the same eigen vectors as A.

Consider the "linear" combinations of the powers of A.

$C = \sum_k c_k A^k$

where $c_k \in R$

Lets try to find the eigen vectors of C.

$C\mathbf{x} = \left(\sum_k c_k A^k\right)\mathbf{x} = \sum_k c_k A^k\mathbf{x} = \sum_k c_k \lambda^k\mathbf{x}$

$\therefore A,\ A^n$ and $\sum_k c_k A^k$

share the same eigen vectors.

Euler's Formula for nth root

$e^{i\theta} = \cos(\theta) + i\sin(\theta)$

When θ is an integral multiple of 2π ,ie., $\theta = 2\pi k$

$e^{i2\pi k} = \cos(2\pi k) + i\sin(2\pi k) = 1$, where $k = \cdots, -2, -1, 0, 1, 2, \cdots$

$\therefore n^{th}$ root of unity can be written as $e^{i2\pi k/n}$, where $k \in \{0, 1, 2, \cdots, (n-1)\}$

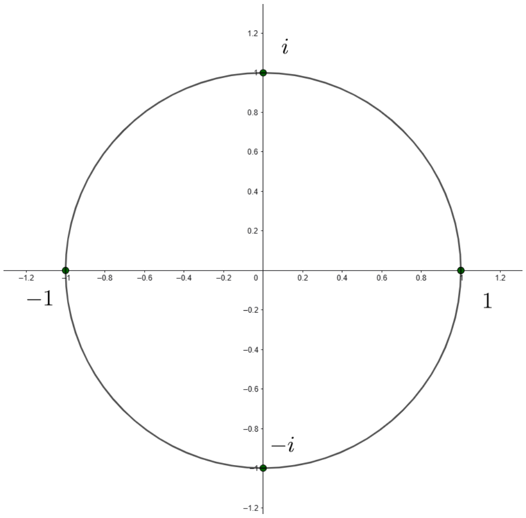
$e^{i2\pi k/n} = (e^{i2\pi/n})^k = \omega^k$

4^{th} root of unity

$e^{i2\pi k/4},\ k \in \{0, 1, 2, 3\}$

the roots are $\{\omega^0, \omega^1, \omega^2, \omega^3\} = \{1, \omega^1, \omega^2, \omega^3\}$

where $\omega = e^{i2\pi/n}$



The roots are points on the unit circle in the complex plane separated by angle $\frac{2\pi}{4} = \frac{\pi}{2}$

Extending this argument, nth roots are points on the unit circle separated by an angle $\frac{2\pi}{n}$

Solutions to the equation $z^n = 1$ are $\{\omega, \omega^2, \omega^3, \cdots, \omega^{n-1}, 1\}$ where $\omega = e^{2\pi i/n}$

(Cyclic) Shift Matrix P

$$P = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 \end{bmatrix},\ \mathbf{x} = \begin{bmatrix} 1 \\ 2 \\ 3 \\ 4 \end{bmatrix},\ P\mathbf{x} = \begin{bmatrix} 2 \\ 3 \\ 4 \\ 1 \end{bmatrix},\ P^2\mathbf{x} = \begin{bmatrix} 3 \\ 4 \\ 1 \\ 2 \end{bmatrix},\ P^3\mathbf{x} = \begin{bmatrix} 4 \\ 1 \\ 2 \\ 3 \end{bmatrix},\ P^4\mathbf{x} = \mathbf{x}$$

For an $n \times n$ matrix P, $P^n = I_n$

All columns vectors of P are unit norm and orthogonal to each other. So P is an orthonormal matrix.

$\therefore\ PP^T = P^TP = I_n$

```
P = [0 1 0 0;
     0 0 1 0;
     0 0 0 1;
```

```
     1 0 0 0]
```

```
P = 4×4
     0     1     0     0
     0     0     1     0
     0     0     0     1
     1     0     0     0
```



```
syms x1 x2 x3 x4;
```

```
Warning: Class 'sym' is defined in a class folder and takes precedence over a function with the
same name that is earlier on the MATLAB path. In a future release, class 'sym' will no longer be
given precedence.

Click here for the locations of the conflicting items.
Click here for guidelines to avoid this warning.
```

```
x = [x1; x2; x3; x4];
P*x
```

ans =

$$\begin{pmatrix} x_2 \\ x_3 \\ x_4 \\ x_1 \end{pmatrix}$$

```
P^2*x
```

ans =

$$\begin{pmatrix} x_3 \\ x_4 \\ x_1 \\ x_2 \end{pmatrix}$$

```
P^3*x
```

ans =

$$\begin{pmatrix} x_4 \\ x_1 \\ x_2 \\ x_3 \end{pmatrix}$$

Example

```
P = [ 0 1 0 0;
     0 0 1 0;
     0 0 0 1;
     1 0 0 0];

x = [5;6;7;8];
```

```
P1x = P*x
```

```
P1x = 4×1
      6
      7
      8
      5
```



```
P2x = P*P*x
```

```
P2x = 4×1
      7
      8
      5
      6
```

```
P3x = P*P*P*x
```

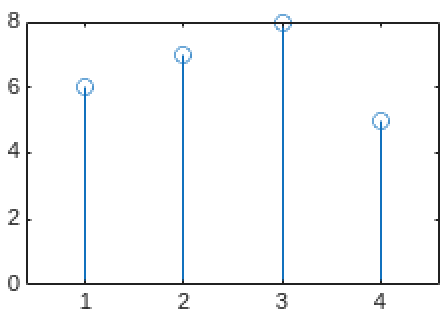
```
P3x = 4×1
      8
      5
      6
      7
```

```
P4x = P*P*P*P*x
```

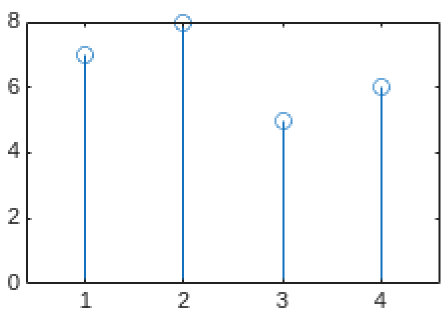
```
P4x = 4×1
      5
      6
      7
      8
```

Plot the shifted vectors

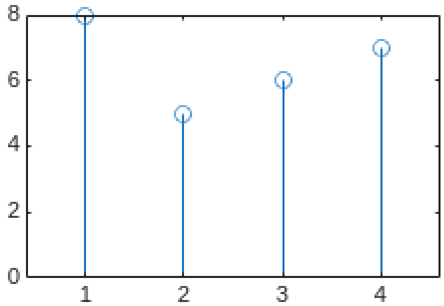
```
stem(P1x)
```



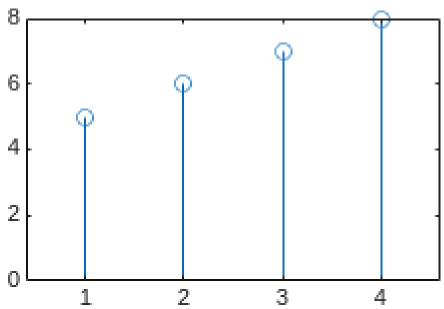

```
stem(P2x)
```



```
stem(P3x)
```




```
stem(P4x)
```



Trace of Shift Matrix P

```
clearvars -except PUBLISH ready
P = [ 0 1 0 0;
      0 0 1 0;
      0 0 0 1;
      1 0 0 0]
```

```
P = 4×4
     0     1     0     0
     0     0     1     0
     0     0     0     1
     1     0     0     0
```

```
trace(P)
```

```
ans =
0
```

```
trace(P^2)
```

```
ans =
0
```

```
trace(P^3)
```

```
ans =
0
```

```
trace(P^4)
```

```
ans =
4
```

```
trace(P^5)
```



```
ans =
0
```

Example 1

```
clearvars -except PUBLISH ready
for i=3:20
    trace(CreateShiftMat(i)^i)
end
```

```
ans =
3
ans =
4
ans =
5
ans =
6
ans =
7
ans =
8
ans =
9
ans =
10
ans =
11
ans =
12
ans =
13
ans =
14
ans =
15
ans =
16
ans =
17
ans =
18
ans =
19
ans =
20
```

Determinant of Shift Matrices

```
clearvars -except PUBLISH ready
P = [ 0 1 0 0;
     0 0 1 0;
```

```
     0 0 0 1;
     1 0 0 0]
```

```
P = 4×4
     0     1     0     0
     0     0     1     0
     0     0     0     1
     1     0     0     0
```



```
det(P)
```

```
ans =
-1
```

```
det(P^2)
```

```
ans =
1
```

```
det(P^3)
```

```
ans =
-1
```

```
det(P^4)
```

```
ans =
1
```

```
det(P^5)
```

```
ans =
-1
```

Example 1

```
P = CreateShiftMat(3)
```

```
P = 3×3
     0     1     0
     0     0     1
     1     0     0
```

```
det(P)
```

```
ans =
1
```

```
det(P^2)
```

```
ans =
1
```

```
det(P^3)
```

```
ans =
```

```
1
```

Example 2


```
P = CreateShiftMat(4)
```

```
P = 4×4
     0     1     0     0
     0     0     1     0
     0     0     0     1
     1     0     0     0
```

```
det(P)
```

```
ans =
-1
```

```
det(P^2)
```

```
ans =
1
```

```
det(P^3)
```

```
ans =
-1
```

```
det(P^4)
```

```
ans =
1
```

Example 3

```
P = CreateShiftMat(6)
```

```
P = 6×6
     0     1     0     0     0     0
     0     0     1     0     0     0
     0     0     0     1     0     0
     0     0     0     0     1     0
     0     0     0     0     0     1
     1     0     0     0     0     0
```

```
det(P)
```

```
ans =
-1
```

```
det(P^2)
```

```
ans =
1
```

```
det(P^3)
```

```
ans =
-1
```

```
det(P^4)
```



```
ans =
1
```

```
det(P^5)
```

```
ans =
-1
```

Check the determinant of Shift Matrices

```
for i=4:20
    det(CreateShiftMat(i)^i)
end
```

```
ans =
1
ans =
1
ans =
1
ans =
1
ans =
1
ans =
1
ans =
1
ans =
1
ans =
1
ans =
1
ans =
1
ans =
1
ans =
1
ans =
1
ans =
1
ans =
1
ans =
1
```

Check the determinant of **odd** sized Shift Matrices

```
for i=3:2:21
    det(CreateShiftMat(i))
```

```
end
```

```
ans =
1
ans =
1
ans =
1
ans =
1
ans =
1
ans =
1
ans =
1
ans =
1
ans =
1
ans =
1
```


Check the determinant of **even** sized Shift Matrices

```
for i=4:2:20
    det(CreateShiftMat(i))
end
```

```
ans =
-1
ans =
-1
ans =
-1
ans =
-1
ans =
-1
ans =
-1
ans =
-1
ans =
-1
ans =
-1
```

Eigen values of Shift matrices

$$P\mathbf{x} = P\begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} = \begin{bmatrix} x_2 \\ x_3 \\ x_4 \\ x_1 \end{bmatrix} = \lambda\begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} \Rightarrow x_1 = \lambda x_4 = \lambda^2 x_3 = \lambda^3 x_2 = \lambda^4 x_1 \ \Rightarrow 1 = \lambda^4 = e^{2\pi ki}$$

$$\lambda = 1^{1/4} = e^{2i\pi k/4} = \{\omega^k\}, k \in \{0, 1, 2, 3\}$$

$\lambda \in \{i, i^2, i^3, i^4\} = \{i, -1, -i, 1\}$

If P is $n \times n$, then $\lambda = e^{2\pi i/N}$

N^{th} roots of unity are the Eigen values of the shift matrix of size $N \times N$.

Eigen vectors of Shift matrices

$(P - \lambda I)\mathbf{x} = \mathbf{0} \quad \mathbf{x} \in RNS(P - \lambda I)$ and $\mathbf{x} \neq \mathbf{0}$

$$\left(\begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 \end{bmatrix} - \lambda \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}\right)\begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} = \begin{bmatrix} -\lambda & 1 & 0 & 0 \\ 0 & -\lambda & 1 & 0 \\ 0 & 0 & -\lambda & 1 \\ 1 & 0 & 0 & -\lambda \end{bmatrix}\begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

4 equations in 4 variables,

$\lambda x_1 - x_2 = 0$

$\lambda x_2 - x_3 = 0$

$\lambda x_3 - x_4 = 0$

$\lambda x_4 - x_1 = 0$

One free variable must be there. Assume x_1 to be the free variable.

$x_1 = 1$

$x_2 = \lambda$

$x_3 = \lambda^2$

$x_4 = \lambda^3$

The eigen vector, $\mathbf{x} = \begin{bmatrix} 1 \\ \lambda \\ \lambda^2 \\ \lambda^3 \end{bmatrix} = \begin{bmatrix} 1 \\ \omega \\ \omega^{2k} \\ \omega^{3k} \end{bmatrix}$ where $\lambda_k = \omega^k$ and $k \in \{0, 1, 2, 3\}$

The eigen vectors as columns will be the matrix $\begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & \omega & \omega^2 & \omega^3 \\ 1 & \omega^2 & \omega^4 & \omega^6 \\ 1 & \omega^3 & \omega^6 & \omega^9 \end{bmatrix}$

Eigen values of P are $\{\omega^k\}; k = \{0, 1, \cdots, n-1\}$

Eigen vectors of P are the column vectors of the matrix $\begin{bmatrix} 1 & 1 & 1 & 1 & \cdots & 1 \\ 1 & \omega & \omega^2 & \omega^3 & \cdots & \omega^{(n-1)} \\ 1 & \omega^2 & \omega^4 & \omega^6 & \cdots & \omega^{2(n-1)} \\ \vdots & & & & & \vdots \\ 1 & \omega^{(n-1)} & \omega^{2(n-1)} & \omega^{3(n-1)} & \cdots & \omega^{(n-1)(n-1)} \end{bmatrix}$

This is the Fourier matrix denoted as F_n

$\overline{F}^T F = NI$ Verify this relation.

Circulant Matrices

is a square matrix.

It can be written as a polynomial of the cyclic shift matrix P.

Circulant matrix as a polynomial of P

$C = c_o I + c_1 P + c_2 P^2 + c_3 P^3$

$D = d_o I + d_1 P + d_2 P^2 + d_3 P^3$

A matrix C of order n with entries c_{ij} is called a circulant matrix if $a_{i_1 j_1} = a_{i_2 j_2}$ whenever $i_1 - j_1 = i_2 - j_2 (\mathrm{mod}\ n)$

A circulant matrix is thus a special case of a Toeplitz matrix.

```
syms c_0 c_1 c_2 c_3 d_0 d_1 d_2 d_3

c = [c_0 c_1 c_2 c_3];
d = [d_0 d_1 d_2 d_3];

P = CreateShiftMat(4);
C = c(1)*eye(4) + c(2)*P + c(3)*P*P + c(4)*P*P*P
```

C =

$$\begin{pmatrix} c_0 & c_1 & c_2 & c_3 \\ c_3 & c_0 & c_1 & c_2 \\ c_2 & c_3 & c_0 & c_1 \\ c_1 & c_2 & c_3 & c_0 \end{pmatrix}$$

```
D = d(1)*eye(4) + d(2)*P + d(3)*P*P + d(4)*P*P*P
```

D =

$$\begin{pmatrix} d_0 & d_1 & d_2 & d_3 \\ d_3 & d_0 & d_1 & d_2 \\ d_2 & d_3 & d_0 & d_1 \\ d_1 & d_2 & d_3 & d_0 \end{pmatrix}$$

$C = c_o I + c_1 P + c_2 P^2 + c_3 P^3$

$D = d_o I + d_1 P + d_2 P^2 + d_3 P^3$


```
clearvars -except PUBLISH ready
P = CreateShiftMat(4)
```

```
P = 4×4
     0     1     0     0
     0     0     1     0
     0     0     0     1
     1     0     0     0
```

```
c = [2 1 3 5]
```

```
c = 1×4
     2     1     3     5
```

```
d = [1 -1 -2 2]
```

```
d = 1×4
     1    -1    -2     2
```

```
C = c(1)*eye(4) + c(2)*P + c(3)*P*P + c(4)*P*P*P
```

```
C = 4×4
     2     1     3     5
     5     2     1     3
     3     5     2     1
     1     3     5     2
```

```
D = d(1)*eye(4) + d(2)*P + d(3)*P*P + d(4)*P*P*P
```

```
D = 4×4
     1    -1    -2     2
     2     1    -1    -2
    -2     2     1    -1
    -1    -2     2     1
```

Properties of circulant matrices

1. $CD = DC$
2. CD is circulant.
3. First row of the matrix CD is the cyclic convolution of c and d

```
C*D
```

```
ans = 4×4
    -7    -5     8     4
     4    -7    -5     8
     8     4    -7    -5
    -5     8     4    -7
```

```
D*C
```

```
ans = 4×4
    -7    -5     8     4
     4    -7    -5     8
     8     4    -7    -5
    -5     8     4    -7
```


Creating Circulant Matrix using Toeplitz

```
clearvars -except PUBLISH ready
v = [1 2 3 0 -1]
```

```
v = 1×5
     1     2     3     0    -1
```

```
C = toeplitz([v(1) fliplr(v(2:end))], v)
```

```
C = 5×5
     1     2     3     0    -1
    -1     1     2     3     0
     0    -1     1     2     3
     3     0    -1     1     2
     2     3     0    -1     1
```

```
C1 = CreateCirculantMat(v)
```

```
C1 = 5×5
     1     2     3     0    -1
    -1     1     2     3     0
     0    -1     1     2     3
     3     0    -1     1     2
     2     3     0    -1     1
```

This should result in a zero matrix

```
C - C1
```

```
ans = 5×5
     0     0     0     0     0
     0     0     0     0     0
     0     0     0     0     0
     0     0     0     0     0
     0     0     0     0     0
```

Eigen values and Eigen vectors of Circulant Matrices

$$C = \sum_{i=0}^{n} c_k P^k$$

$$C\mathbf{q}_k = \left(\sum_{i=0}^{n-1} c_i P^i\right)\mathbf{q}_k = \sum_{i=0}^{n-1} c_i P^i \mathbf{q}_k = \sum_{i=0}^{n-1} c_i \omega^{ik} \mathbf{q}_k$$

$$k = 0 \quad C\mathbf{q}_0 = \lambda_0 \mathbf{q}_0 \quad \lambda_0 = \sum_{i=0}^{n-1} c_i = c_0 + c_1 + c_2 \cdots + c_{n-1}$$

$$k = 1 \quad C\mathbf{q}_1 = \lambda_1 \mathbf{q}_1 \quad \lambda_1 = \sum_{i=0}^{n-1} c_i \omega^i = c_0 + c_1 \omega + c_2 \omega^2 + \cdots + c_{n-1}\omega^{(n-1)}$$

$$k = 2 \quad C\mathbf{q}_2 = \lambda_2 \mathbf{q}_2 \quad \lambda_2 = \sum_{i=0}^{n-1} c_i \omega^{2i} = c_0 + c_1 \omega^2 + c_2 \omega^4 + \cdots + c_{n-1}\omega^{(n-1)}$$

$$\vdots$$

$$k = n-1 \quad C\mathbf{q}_{n-1} = \lambda_{n-1} \mathbf{q}_{n-1} \quad \lambda_{n-1} = \sum_{i=0}^{n-1} c_i \omega^{i(n-1)} = c_0 + c_1 \omega^{2(n-1)} + c_2 \omega^{4(n-1)} + \cdots + c_{n-1}\omega^{(n-1)(n-1)}$$

$$\begin{bmatrix} \lambda_0 \\ \lambda_1 \\ \lambda_2 \\ \vdots \\ \lambda_{n-1} \end{bmatrix} = \begin{bmatrix} 1 & 1 & 1 & 1 & \cdots & 1 \\ 1 & \omega & \omega^2 & \omega^3 & \cdots & \omega^{(n-1)} \\ 1 & \omega^2 & \omega^4 & \omega^6 & \cdots & \omega^{2(n-1)} \\ \vdots & & & & & \vdots \\ 1 & \omega^{(n-1)} & \omega^{2(n-1)} & \omega^{3(n-1)} & \cdots & \omega^{(n-1)(n-1)} \end{bmatrix} \begin{bmatrix} c_0 \\ c_1 \\ c_2 \\ \vdots \\ c_{n-1} \end{bmatrix}$$

Trace of Circulant Matrices

Example 2

x1 = [1 2 1 2]

x2 = [1 2 3 4]

```
x1 = [1 2 1 2]
```

```
x1 = 1×4
     1     2     1     2
```

```
x2 = 1:4
```

```
x2 = 1×4
     1     2     3     4
```

```
cconv(x1,x2,4)
```

```
ans = 1×4
    16    14    16    14
```

```
conv(x1,x2)
```

```
ans = 1×7
```

```
     1     4     8    14    15    10     8
```



```
P = [ 0 1 0 0;
     0 0 1 0;
     0 0 0 1;
     1 0 0 0]
```

```
P = 4×4
     0     1     0     0
     0     0     1     0
     0     0     0     1
     1     0     0     0
```

```
c = [1 2 -1 -2];
C = c(1)*eye(4) + c(2)*P + c(3)*P^2 + c(4)*P^3
```

```
C = 4×4
     1     2    -1    -2
    -2     1     2    -1
    -1    -2     1     2
     2    -1    -2     1
```

```
clearvars -except PUBLISH ready
P = [ 0 1 0 0;
     0 0 1 0;
     0 0 0 1;
     1 0 0 0];

s1 = [2 1 3 5];

C = s1(1)*eye(4) + s1(2)*P + s1(3)*P*P + s1(4)*P*P*P
```

```
C = 4×4
     2     1     3     5
     5     2     1     3
     3     5     2     1
     1     3     5     2
```

```
TC = toeplitz([s1(1) fliplr(s1(2:end))],s1);
```

Toeplitz Matrix

is a constant diagonal matrix. ie, it satisfies the following condition

$T_{i,j} = T_{i+1,j+1} = a_{i-j}$

A matrix T with entries t_{ij} is called a Toeplitz matrix if

$t_{i_1 j_1} = t_{i_2 j_2}$ whenever $i_1 - j_1 = i_2 - j_2$. This means t_{ij} is a function of $i - j$.

This matrix is also known as isodiagonal matrices.

The numbers in the first row and first column of the Toeplitz martrix are called the $generators$ of T.

A Toeplitz matrix has constant diagonals. The first row and column tell you the rest of the matrix, because they contain the first entry of every diagonal. **Circulant matrices** are Toeplitz matrices that satisfy the extra "wraparound" condition that makes them periodic. Effectively c_{-3} is the same as c_1 (for 4×4 circulants) :

$$\textbf{Toeplitz matrix} \quad A = \begin{bmatrix} a_0 & a_{-1} & a_{-2} & a_{-3} \\ \boldsymbol{a_1} & a_0 & a_{-1} & a_{-2} \\ a_2 & \boldsymbol{a_1} & a_0 & a_{-1} \\ a_3 & a_2 & \boldsymbol{a_1} & a_0 \end{bmatrix} \qquad \textbf{Circulant matrix} \quad C = \begin{bmatrix} c_0 & c_3 & c_2 & \boldsymbol{c_1} \\ \boldsymbol{c_1} & c_0 & c_3 & c_2 \\ c_2 & \boldsymbol{c_1} & c_0 & c_3 \\ c_3 & c_2 & \boldsymbol{c_1} & c_0 \end{bmatrix}$$

Toeplitz matrix need not be a square matrix unlike the circulant or the cyclic shift matrix.

$$T = \begin{bmatrix} a_0 & a_{-1} & a_{-2} & \cdots & \cdots & a_{n-1} \\ a_1 & a_0 & a_{-1} & \cdots & & a_{n-2} \\ a_2 & a_1 & \ddots & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & a_{-1} & a_{-2} \end{bmatrix}$$

is a toeplitz matrix.

Toeplitz matrix is a filter in Signal Processing.

Linear finite difference equations with constant coefficients produce Toeplitz matrices.

The inverse of a Toeplitz matrix is usually not a toeplitz matrix.

Creating Toeplitz matrix using toeplitz function

toeplitz(first col vector, first row vector)

```
toeplitz(1:3,1:4)
```

```
ans = 3×4
     1     2     3     4
     2     1     2     3
     3     2     1     2
```

Convolution of 2 sequences $x[n]$ and $h[n]$ is defined as

$$y[k] = h[n] * x[n] = \sum_{n=0}^{k} x[n]h[k-n]$$

To demonstrate this lets assume 2 small simple sequences, assuming indexing beginning from 0.

$x = [1 \quad 2 \quad 3] = x[0], x[1], x[2]$

$h = [4 \quad 5] = h[0], h[1]$

The convolution will have $3 + 2 - 1 = 4$ samples. They are

$y[0] = x[0]h[0]$

$y[1] = x[0]h[1] + x[1]h[0]$

$y[2] = x[1]h[1] + x[2]h[0]$

$y[3] = x[2]h[1]$

This can be written as the matrix equation below

$$\mathbf{y} = \begin{bmatrix} h[0] & 0 & 0 \\ h[1] & h[0] & 0 \\ 0 & h[1] & h[0] \\ 0 & 0 & h[1] \end{bmatrix} \begin{bmatrix} x[0] \\ x[1] \\ x[2] \end{bmatrix} = T\mathbf{x}$$

T can be created using $convmtx$ function.

For High pass filter

```
T = convmtx([1 -1],4);
```

For low pass filter - moving average

```
T = convmtx(ones(4,1)/4,4);
```

running/moving average = low pass filter

filter co-efficients = $[1 \quad 1 \quad 1 \quad 1] \times 0.25$

Averaging over nearby samples removes the rapidly changing high frequency part of the signal. This is equivalent to a Low pass filter.

The differences of consecutive samples will result in cancelling out slowly varying or low frequencies. The output sequence will be predominantly having high frequency components. Or, this is equivalent to a high pass filter.

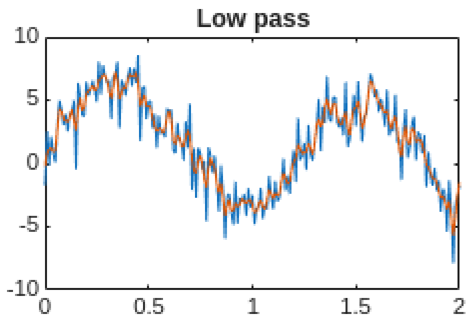
filter co-efficients = $[1 \quad -1]$

Toeplitz matrix as low pass filter

```
clearvars -except PUBLISH ready
t = 0:0.01:2;
n = length(t);
y = 2*sin(2*t) +  5*sin(5*t) + 1.5*randn(1,n);

T = diag(0.25*ones(n-1,1),1) + diag(0.5*ones(n,1)) +
diag(0.25*ones(n-1,1),-1);
y1 = T*y';
plot(t,y);hold on
plot(t,y1);hold off
```

```
title("Low pass")
```

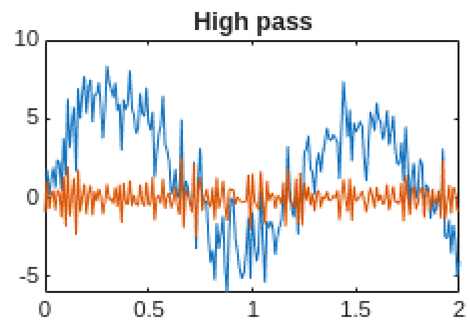


School of AI, AVV

Toeplitz matrix as high pass filter

Check the vector (rows / column vectors)

```
clearvars -except PUBLISH ready
t = 0:0.01:2;
n = length(t);
y = 2*sin(2*t) +  5*sin(5*t) + 1.5*randn(1,n);
% plot(t,y)
T = diag(-0.25*ones(n-1,1),1) + diag(0.5*ones(n,1)) +
diag(-0.25*ones(n-1,1),-1);
y1 = T*y';
plot(t,y);hold on
plot(t,y1);hold off
title("High pass")
```



Kronecker product

of 2 matrices $A_{m\times n}$ and $B_{p\times q}$ is defined as the following $pm \times qn$ sized block matrix

$$A \otimes B = \begin{bmatrix} a_{11} & \cdots & a_{n1} \\ \vdots & \ddots & \vdots \\ a_{m1} & \cdots & a_{mn} \end{bmatrix} \otimes \begin{bmatrix} b_{11} & \cdots & b_{q1} \\ \vdots & \ddots & \vdots \\ b_{p1} & \cdots & b_{pq} \end{bmatrix} = \begin{bmatrix} a_{11}B & \cdots & a_{n1}B \\ \vdots & \ddots & \vdots \\ a_{m1}B & \cdots & a_{mn}B \end{bmatrix}$$

A simple case using Identity matrices of different sizes

$A = I_2$ and $B = I_3$

$A \otimes B = I_6$

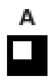
```
kron(eye(2),eye(3))
```



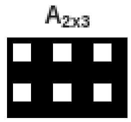
```
ans = 6×6
     1     0     0     0     0     0
     0     1     0     0     0     0
     0     0     1     0     0     0
     0     0     0     1     0     0
     0     0     0     0     1     0
     0     0     0     0     0     1
```

Visualizing the Kronecker product

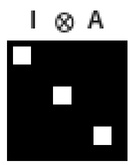
```
A = zeros(20,20);
A(4:12,4:12)=1;
imshow(single(A))
title("A")
```




```
imshow(kron(ones(2,3),A))
title("A_{2x3}")
```

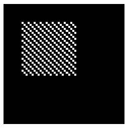



```
imshow(kron(eye(3),A))
title("I \otimes A")
```




```
imshow(kron(A,eye(3)))
title("A \otimes I")
```

$A \otimes I$



$$\text{if } B \text{ and } C \text{ are} \left\{ \begin{array}{l} \text{nonsingular} \\ \text{lower(upper) triangular} \\ \text{banded} \\ \text{symmetric} \\ \text{positive definite} \\ \text{stochastic} \\ \text{Toeplitz} \\ \text{permutations} \\ \text{orthogonal} \end{array} \right\}, \text{ then } B \otimes C \text{ is} \left\{ \begin{array}{l} \text{nonsingular} \\ \text{lower(upper) triangular} \\ \text{block banded} \\ \text{symmetric} \\ \text{positive definite} \\ \text{stochastic} \\ \text{block Toeplitz} \\ \text{a permutation} \\ \text{orthogonal} \end{array} \right\}.$$

Properties

Bilinear and Associative

$A \otimes (B + C) = A \otimes B + A \otimes B)$

$(B + C) \otimes A = B \otimes A + C \otimes A$

$(kA) \otimes B = A \otimes (kB) = k(A \otimes B)$

$(A \otimes B) \otimes C = A \otimes (B \otimes C)$

$A \otimes 0 = 0 \otimes A = 0$

where A, B, C are matrices, 0 is a zero matrix, k is a scalar.

Not Commutative

In general

$A \otimes B \neq B \otimes A$

```
clearvars -except PUBLISH ready
rng(100)
A = randi([-1,3],2);
B = randi([-1,5],2);
kron(A,B)
```

```
ans = 4×4
    -1     3    -1     3
```

```
    -1     4    -1     4
     0     0    -3     9
     0     0    -3    12
```



```
kron(B,A)
```

```
ans = 4×4
    -1    -1     3     3
     0    -3     0     9
    -1    -1     4     4
     0    -3     0    12
```

Mixed Product property

For rectangular matrices $A, B, C \& D$

$(A \otimes B)(C \otimes D) = (AC) \otimes (BD)$

```
clearvars -except PUBLISH ready
rng(100)
A = randi([-1,3],2);
B = randi([-1,5],2);
C = randi([-1,3],2);
D = randi([-1,5],2);

S1 = kron(A,B)*kron(C,D);
S2 = kron(A*C,B*D);
sum(S1-S2,"all")
```

```
ans =
0
```

Hadamard property

$(A \otimes B) \odot (C \otimes D) = (A \odot C) \otimes (B \odot D)$

RHS implies

size(A) = size (C) = $m \times n$

and

size(B) = size(D) = $p \times q$

Using this information to evaluate the LHS

```
clearvars -except PUBLISH ready
rng(12882)
A = randi([-1,3],2);
B = randi([-1,5],2);
C = randi([-1,3],2);
D = randi([-1,5],2);

S1 = kron(A,B).*kron(C,D);
```

```
S2 = kron(A.*C,B.*D);
sum(S1-S2,"all")
```

```
ans =
0
```


Inverse

For invertible square matrices

$(A \otimes B)^{-1} = A^{-1} \otimes B^{-1}$

or $(A \otimes B) \times (A^{-1} \otimes B^{-1}) = I \otimes I$

Holds for pseudo inverse also.

$(A \otimes B)^{+} = A^{+} \otimes B^{+}$

```
clearvars -except PUBLISH ready
rng(100)
A = randi([-1,3],2);
B = randi([-1,5],2);

S1 = inv(kron(A,B));
S2 = kron(inv(A),inv(B));
sum(S1-S2,"all")
```

```
ans =
0
```

```
S3 = pinv(kron(A,B));
S4 = kron(pinv(A),pinv(B));
sum(S3-S4,"all")
```

```
ans =
3.9135e-15
```

Transpose

$(A \otimes B)^{T} = A^{T} \otimes B^{T}$

$(A \otimes B)^{*} = A^{*} \otimes B^{*}$ Here $*$ stands for conjugate transposition

```
clearvars -except PUBLISH ready
rng(100)
A = randi([-1,3],2)
```

```
A = 2×2
     1     1
     0     3
```

```
B = randi([-1,5],2)
```

```
B = 2×2
    -1     3
    -1     4
```



```
transpose(kron(A,B))
```

```
ans = 4×4
    -1    -1     0     0
     3     4     0     0
    -1    -1    -3    -3
     3     4     9    12
```

```
kron(transpose(A),transpose(B))
```

```
ans = 4×4
    -1    -1     0     0
     3     4     0     0
    -1    -1    -3    -3
     3     4     9    12
```

Determinant

If A is $n \times n$ and B is $m \times m$, then

$$|A \otimes B| = |A|^m |B|^n$$

```
clearvars -except PUBLISH ready
rng(1001)
A = randi([1,3],2)
```

```
A = 2×2
     1     1
     1     2
```

```
B = randi([1,5],3)
```

```
B = 3×3
     1     2     5
     1     4     1
     2     3     3
```

```
det(kron(A,B))
```

```
ans =
324
```

```
det(A)^3*det(B)^2
```

```
ans =
324
```

Trace

$tr(A \otimes B) = tr(A) \times tr(B)$

```
clearvars -except PUBLISH ready
rng(12001)
A = randi([1,3],2)
```



```
A = 2×2
     2     3
     1     2
```

```
B = randi([1,5],3)
```

```
B = 3×3
     4     3     1
     2     3     3
     3     3     1
```

```
trace(kron(A,B))
```

```
ans = 
32
```

```
trace(A)
```

```
ans = 
4
```

```
trace(B)
```

```
ans = 
8
```

Singular values

If A and B have r_A and r_B non zero singular values respectively , then

$A \otimes B$ has $r_A \times r_B$ non zero singular values

$\therefore\ rank(A \otimes B) = rank(A) \times rank(B)$

```
clearvars -except PUBLISH ready
rng(12001)
A = randi([1,3],3,2)*randi([1,3],2,4);
B = randi([1,5],3,2)*randi([1,3],2,5);
[~,S1,~]=svd(kron(A,B));
sum(diag(S1)>1e-5)
```

```
ans = 
4
```

Eigen Values and Eigen Vectors

If $\{\lambda_1, \cdots, \lambda_n\}$ are the eigen values of $A_{n\times n}$ and $\{\mu_1, \cdots, \mu_m\}$ are the eigen values of $B_{m\times m}$ then

the eigen values of $A \otimes B$ are $\lambda_i \mu_j$ where $i = 1, \cdots, n$ and $j = 1, \cdots, m$

$A\mathbf{x} = \lambda\mathbf{x}$

$B\mathbf{y} = \lambda\mathbf{y}$

$(A \otimes B)(\mathbf{x} \otimes \mathbf{y}) = (A\mathbf{x}) \otimes (B\mathbf{y}) = \lambda\mathbf{x} \otimes \mu\mathbf{y} = \lambda\mu(\mathbf{x} \otimes \mathbf{y})$

Linear Convolution

is Teoplitz multiplication

Example 1

```
clearvars -except PUBLISH ready
x = [1 2 3 2];
y = [1 0 2];
toeplitz(x,y)
```

```
ans = 4×3
     1     0     2
     2     1     0
     3     2     1
     2     3     2
```

```
conv(x,y)
```

```
ans = 1×6
     1     2     5     6     6     4
```

Example 2

```
clearvars -except PUBLISH ready
x = [1 2 3]
```

```
x = 1×3
     1     2     3
```

```
y = [1 1 1]
```

```
y = 1×3
     1     1     1
```

```
conv(x,y)
```

```
ans = 1×5
     1     3     6     5     3
```

Example 3

```
clearvars -except PUBLISH ready
x = [1 4 7 3]
```

```
x = 1×4
     1     4     7     3
```

```
y = [1 2]
```

```
y = 1×2
     1     2
```



```
conv(x,y)
```

```
ans = 1×5
     1     6    15    17     6
```

Cyclic/Circular Convolution

is Circulant multiplication

Example 1

x1 = [1 2 3]

x2 = [5 0 4]

```
clearvars -except PUBLISH ready
x1 = 1:3
```

```
x1 = 1×3
     1     2     3
```

```
x2 = [5 0 4]
```

```
x2 = 1×3
     5     0     4
```

```
cconv(x1,x2,3)
```

```
ans = 1×3
    13    22    19
```

Example 2

```
clearvars -except PUBLISH ready
x = [1 2 3 2];
y = [1 0 2];
cconv(x,y,4)
```

```
ans = 1×4
     7     6     5     6
```

```
clearvars -except PUBLISH ready
x = 1:4
```

```
x = 1×4
     1     2     3     4
```

```
C = toeplitz([x(1) x(2:end)],x)
```

```
C = 4×4
     1     2     3     4
     2     1     2     3
     3     2     1     2
     4     3     2     1
```



```
eig(C)
```

```
ans = 4×1
   -3.4142
   -1.0990
   -0.5858
    9.0990
```

```
C1 = toeplitz(x)
```

```
C1 = 4×4
     1     2     3     4
     2     1     2     3
     3     2     1     2
     4     3     2     1
```

```
clearvars -except PUBLISH ready
P = [ 0 1 0 0;
      0 0 1 0;
      0 0 0 1;
      1 0 0 0]
```

```
P = 4×4
     0     1     0     0
     0     0     1     0
     0     0     0     1
     1     0     0     0
```

```
C1 = eye(4) + 2*P + 3*P^2 + 4*P^3
```

```
C1 = 4×4
     1     2     3     4
     4     1     2     3
     3     4     1     2
     2     3     4     1
```

```
C2 = toeplitz([1 4:-1:2],1:4)
```

```
C2 = 4×4
     1     2     3     4
     4     1     2     3
     3     4     1     2
     2     3     4     1
```

```
eig(C1)
```

```
ans = 4×1
  10.0000 + 0.0000i
  -2.0000 + 2.0000i
```

```
-2.0000 - 2.0000i
-2.0000 + 0.0000i
```


Symbols

linear convolution

\ast $\ast$

circular convolution

yet to find out--------!?!?! circle around the convolution/asterisk symbol

kronecker product

\otimes $\otimes$

Hadamard product

\odot $\odot$

Coding exercises

1 Create Shift Matrix of size n

```
function Pn = CreateShiftMat(n)
    Pn = zeros(n);
    Pn(n,1) = 1;
    Pn(1:n-1,2:n)=eye(n-1);
end
```

2 Create Circulant matrix

```
function Cn = CreateCirculantMat(c)
    n = length(c);
    Pn = CreateShiftMat(n);
    Cn = zeros(n);
    for i = 1:n
        Cn = Cn + c(i)*Pn^(i-1);
    end
end
```

3 Check if the given matrix is a cyclic shift matrix or not.

4 Check if the given matrix is a circulant matrix or not.

5 Create the fourier matrix of size n.

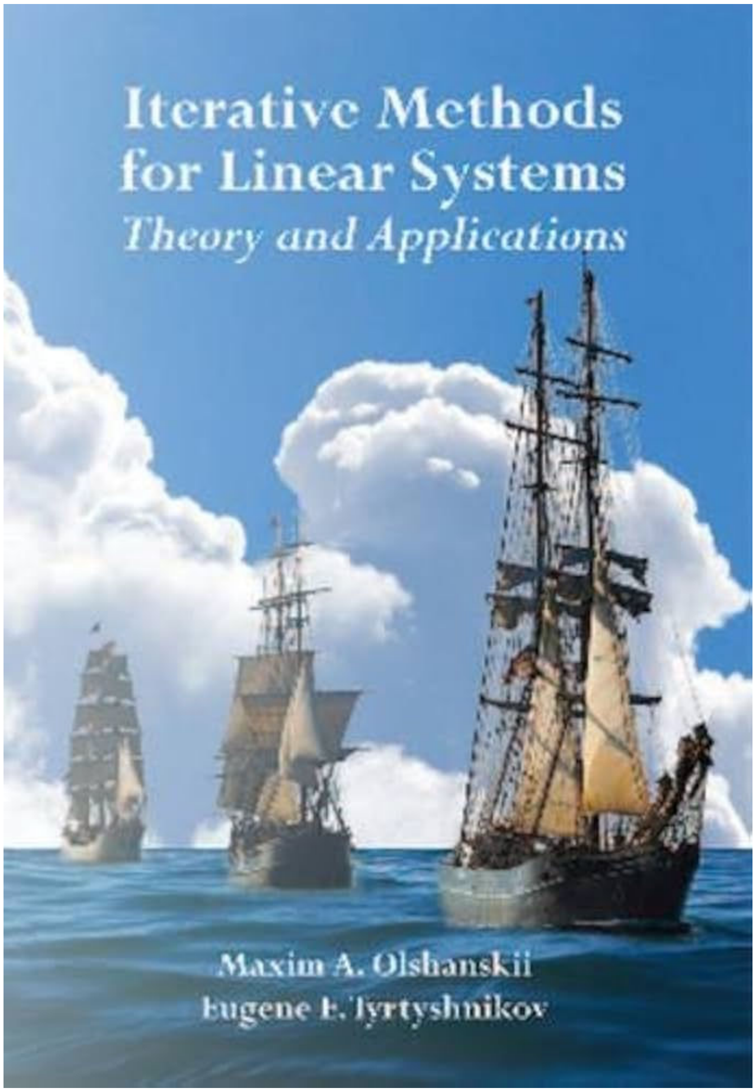
6. Show that the eigen vectors (as columns)of the P or the C matrix are orthogonal to each other.

7. Check if the given matrix is toeplitz or not.

Read more

Tensor product

Hadamard product




```
if(PUBLISH == ready)
    path = '/media/user/DATA4LINUX/new1/Repos/Mine/MFC4_22MAT230/';
    mlxfile = matlab.desktop.editor.getActive().Filename;
    [~, name, ext] = fileparts(mlxfile);
    outfile = [path, name, ext,'.pdf']
    export(matlab.desktop.editor.getActive().Filename, outfile);
    if ispc
        winopen(outfile);
    elseif ismac
        system(['open ' char(outfile)]);
    else
        system("env -u LD_LIBRARY_PATH xdg-open '" + outfile + "' &");
    end
```

```
end
```

```
Unrecognized function or variable 'ready'.
```