# Eigenvalues and Eigenvectors

## NDSU ECE 463/663

## Lecture #4


## Inst: Jake Glower


Please visit Bison Academy for corresponding lecture notes, homework sets, and solutions

# Eigenvalues and Eigenvectors

In Linear Algebra, you cover eigenvalues ($\lambda$) and eigenvectors ($\Lambda$)

- Eigenvalues: $|\lambda I - A| = 0$
- Eigenvectors: $A\Lambda = \lambda\Lambda$

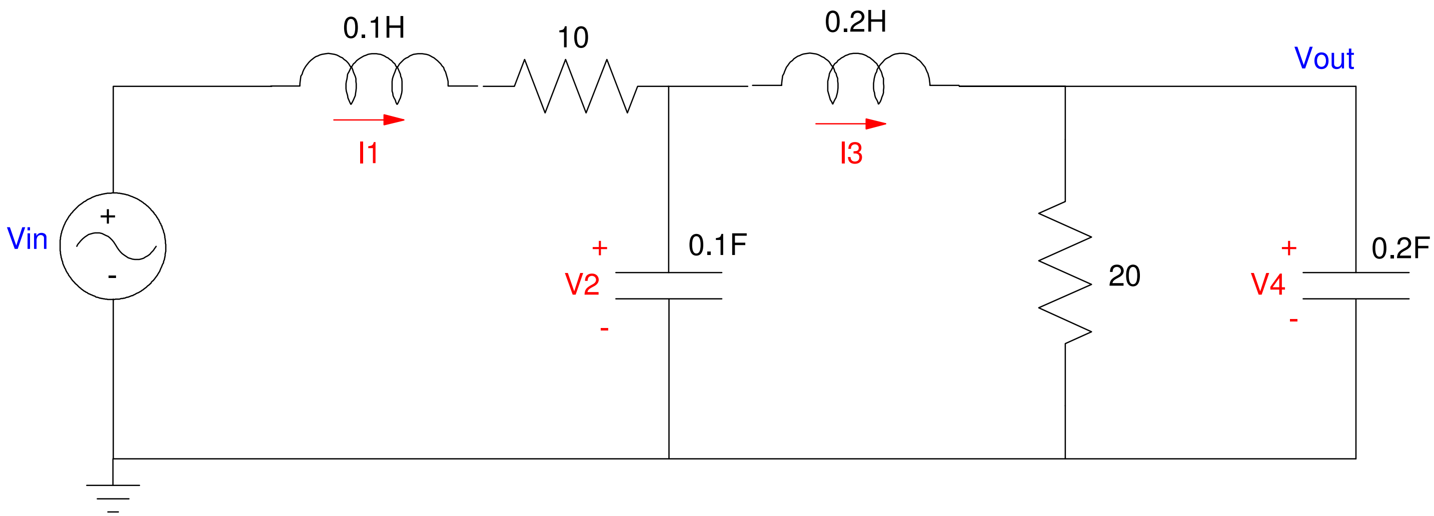
What does this mean?

- Eigenvalues tell you *how* a system behaves
- Eigenvectors tell you *what* behaves that way

# RLC Circuit Example: (Lecture #3)

$$sX = AX + BU$$

$$\begin{bmatrix} sI_1 \\ sV_2 \\ sI_3 \\ sV_4 \end{bmatrix} = \begin{bmatrix} -100 & -10 & 0 & 0 \\ 10 & 0 & -10 & 0 \\ 0 & 5 & 0 & -5 \\ 0 & 0 & 5 & -0.25 \end{bmatrix} \begin{bmatrix} I_1 \\ V_2 \\ I_3 \\ V_4 \end{bmatrix} + \begin{bmatrix} 10 \\ 0 \\ 0 \\ 0 \end{bmatrix} V_{in}$$



## Eigenvalues:

Eigenvalues and poles are the same thing

The transfer function to V4 is

$$Y = V_4 = \left(\frac{2500}{(s+98.50)(s+0.5025)(s^2+0.7525s+75.38)}\right)V_{in}$$

The eigenvalues of the A matrix are

```
A4 = [-100,-10,0,0;10,0,-10,0;0,5,0,-5;0,0,5,-0.25]

 -100.0000  -10.0000         0         0
   10.0000         0  -10.0000         0
         0    5.0000         0   -5.0000
         0         0    5.0000   -0.2500

eig(A4)

 -98.9950
  -0.3763 + 8.6740i
  -0.3763 - 8.6740i
  -0.5025
```

# Eigenvalues tell you *how* the system behaves

- The real part tells you the 2% settling time
- The complex part tells you the frequency of oscillation
- The damping ratio tells you the overshoot for a step input

```
eig(A4)

 -98.9950
  -0.3763 + 8.6740i
  -0.3763 - 8.6740i
  -0.5025
```

- *Something* decays as exp(-98.995t)
- *Something* decays as exp(-0.3763t) cos(8.6740t)
- *Something* decays as exp(-0.5025t)

# Eigenvalues specify what the transient response looks like

Example: Impulse Response

$$Y = \left(\frac{2500}{(s+98.50)(s+0.5025)(s^2+0.7525s+75.38)}\right)$$

$$= \left(\frac{a}{s+98.50}\right) + \left(\frac{b\angle -\phi}{s+0.3763+j8.674}\right) + \left(\frac{b\angle \phi}{s+0.3763-j8.674}\right) + \left(\frac{c}{s+0.5025}\right)$$

Take the inverse LaPlace transform

$$y(t) = ae^{-98.995t} + 2be^{-0.3763t}\cos(8.6740t + \phi) + ce^{-0.5025t}$$

## Eigenvectors

The eigenvectors tell you *what* behaves that way.

The transient response will be

$$X(t) = a_1\Lambda_1 e^{\lambda_1 t} + a_2\Lambda_2 e^{\lambda_2 t} + a_3\Lambda_3 e^{\lambda_3 t} + a_4\Lambda_4 e^{\lambda_4 t}$$

where

$a_i$ are constants determined by the initial condition,

$\Lambda_i$ are the eigenvectors of A, and

$\lambda_i$ are the eigenvalues of A.

## Eigenvectors (cont'd)

At t=0, you get

$$X(0) = a_1\Lambda_1 + a_2\Lambda_2 + a_3\Lambda_3 + a_4\Lambda_4$$

or

$$X_0 = \Lambda A$$

$$A = \Lambda^{-1} X_0$$

In short,

- The eigenvalues tell you how the system behaves,
- The eigenvectors tell you what behaves that way,
- The initial condition tells you how much you excite each eigenmode.

## Circuit Example:

Each eigenvalue has a corresponding eigenvector

- denoted by color

Eigenvalues

```
V = eig(A)

 -98.9950  -0.3763 + 8.6740i  -0.3763 - 8.6740i  -0.5025
```

Eigenvectors

```
[M,V] = eig(A)

M =  (eigenvector matrix)

-0.9950    -0.0705 + 0.0061i  -0.0705 - 0.0061i  -0.0710
 0.1000     0.7075             0.7075              0.7067
-0.0050    -0.0439 - 0.6076i  -0.0439 + 0.6076i  -0.0355
 0.0003    -0.3498 + 0.0304i  -0.3498 - 0.0304i   0.7031
```

## Translation:

- If you make the initial condition proportional to the first eigenvector, only the first eigenvalue appears (all other terms are zero)

Let

$$X_0 = \begin{bmatrix} I_1(0) \\ V_2(0) \\ I_3(0) \\ V_4(0) \end{bmatrix} = 2\begin{bmatrix} -0.9950 \\ 0.1000 \\ -0.0050 \\ 0.0003 \end{bmatrix} = 2\Lambda_1$$

then

$$X(t) = X_0 \cdot e^{-98.995t}$$

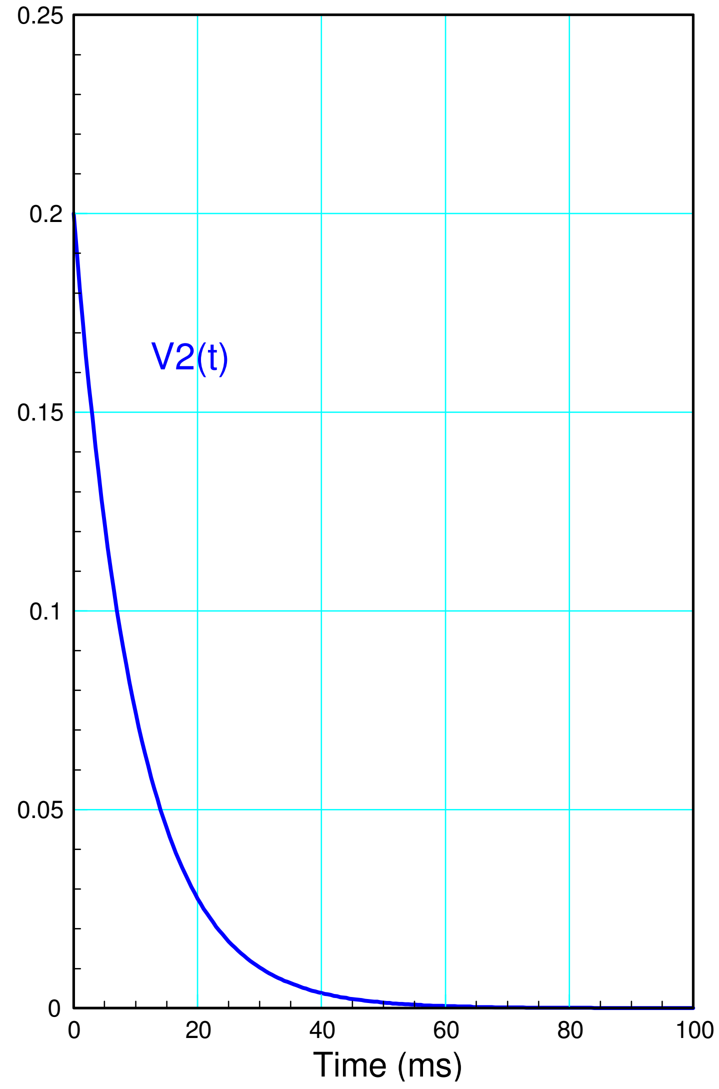
$$X(t) = 2\Lambda_1 \cdot e^{\lambda_1 t}$$

# Eigenvalue at -98.995

```
X0 = 2*M(:,1)

I1   -1.9899
V2    0.2000
I3   -0.0101
V4    0.0005

G = ss(A4, X0, C4, D4);
t = [0:200]' / 200 * 0.1;
X = impulse(G, t);
plot(t,X(:,2))
```

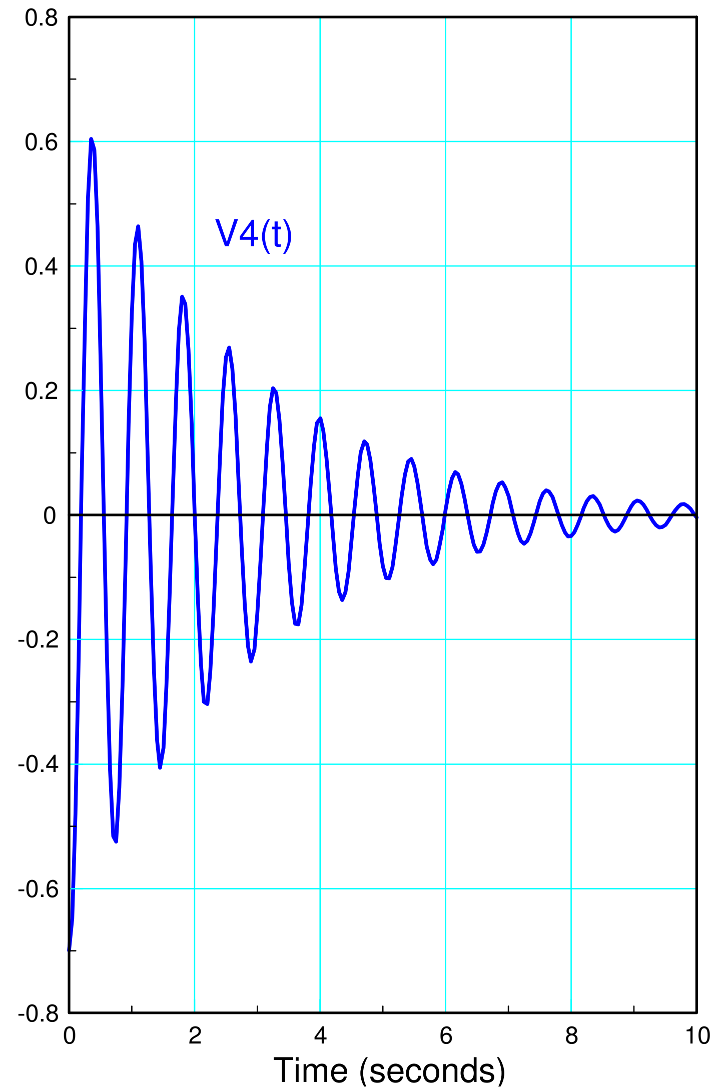


# λ = -0.3763 + 8.6740i

```
X0 = 2*real(M(:,2))

I1    -0.1410
V2     1.4151
I3    -0.0877
V4    -0.6996

G = ss(A4, X0, C4, D4);
t = [0:200]' / 200 * 0.1;
X = impulse(G, t);
plot(t,X)
```

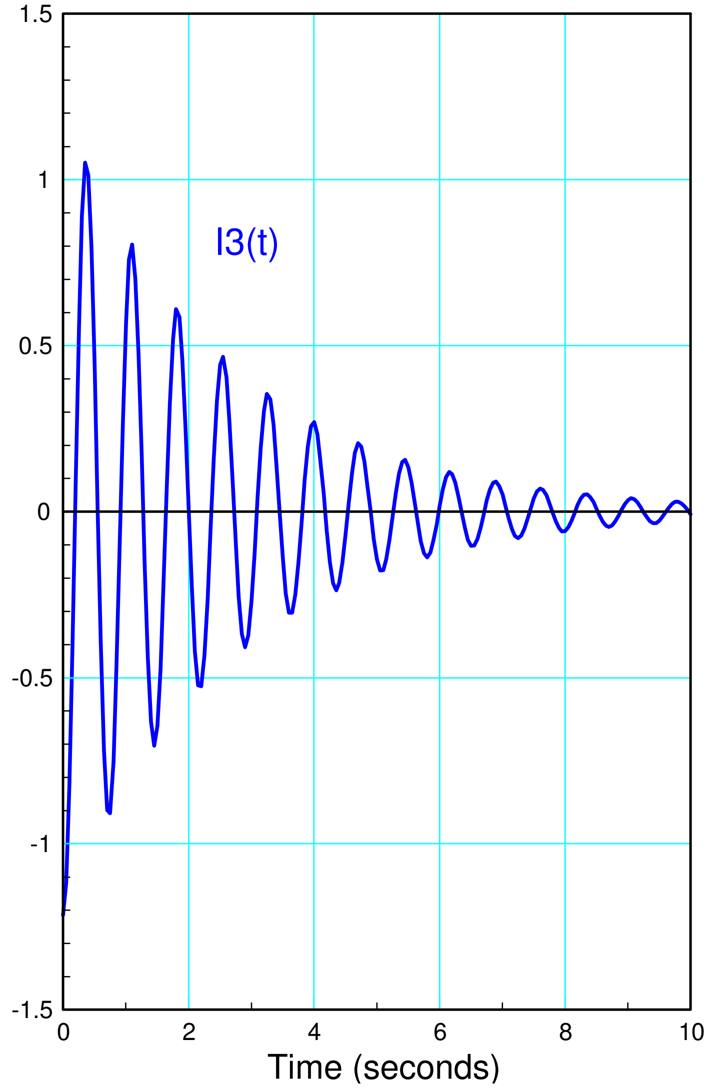


# λ = -0.3763 + 8.6740i

```
X0 = imag(2*M(:,2))

I1     0.0123
V2          0
I3    -1.2152
V4     0.0608

G = ss(A4, X0, C4, D4);
X = impulse(G, t);
plot(t,X(:,[3]))
```

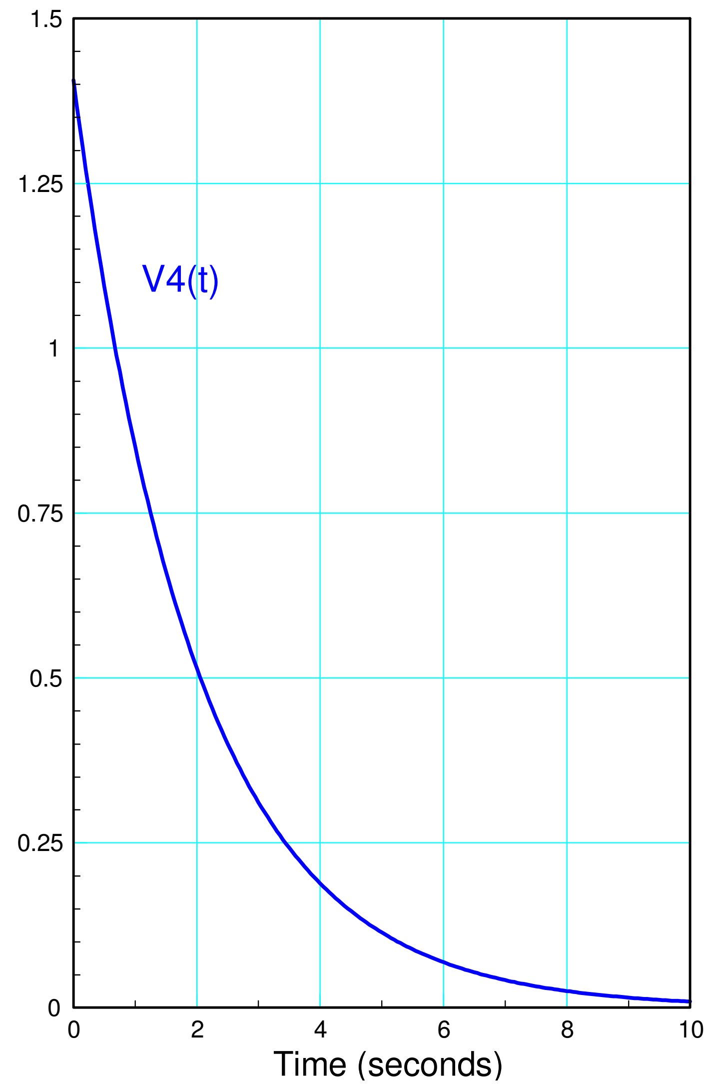


# Eigenvalue = -0.5025

```
X0 = 2*M(:,4)

I1   -0.1420
V2    1.4133
I3   -0.0710
V4    1.4062

G = ss(A4, X0, C4, D4);
X = impulse(G, t);
plot(t,X(:,[4]))
[t,X(:,4)]
```

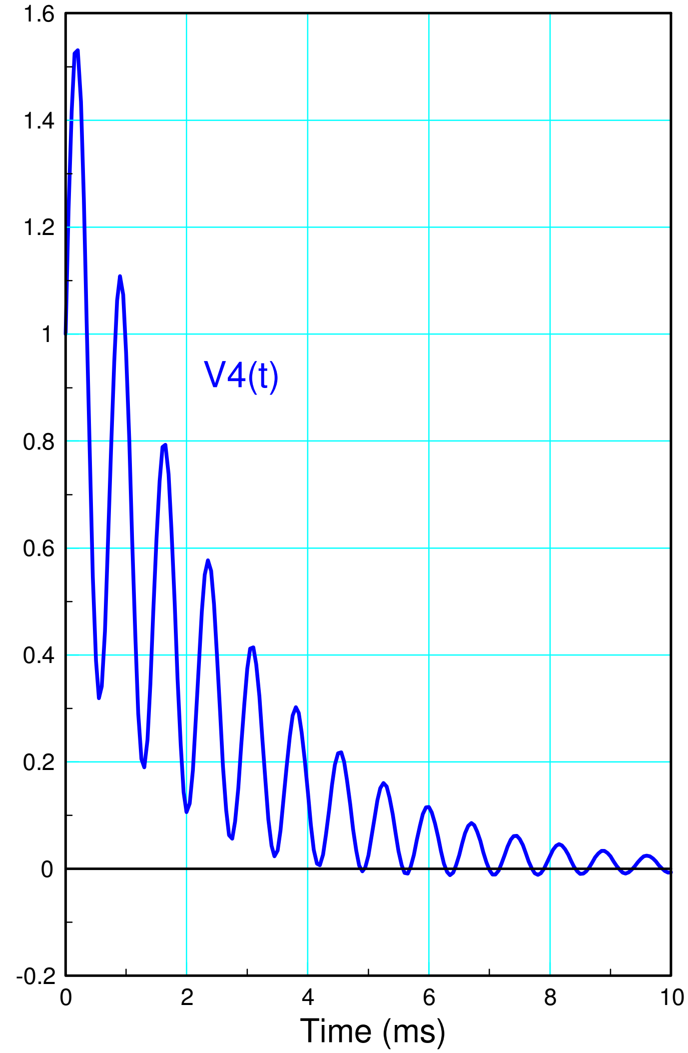


# Random Initial Condition

- All four eigenvectors are excited
- All four modes show up in the output
- The fast poles decay quickly
- The slow (dominant) pole eventually wins

```
X0 = ones(4,1);
G = ss(A4, X0, C4, D4);

t = [0:200]' / 200 * 10;
X = impulse(G, t);
plot(t,X(:,4))
```



# Example 2: 10-Stage RC Filter

From lecture #3

```
                                                10000000000
-----------------------------------------------------------------------------------------------------
(s+39.31) (s+36.72) (s+32.67) (s+27.51) (s+21.69) (s+15.75) (s+10.2) (s+5.539) (s+2.181) (s+0.4234)
```

A is a 10x10 matrix:

A10

| | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| -20.2000 | 10.0000 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 10.0000 | -20.2000 | 10.0000 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 10.0000 | -20.2000 | 10.0000 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 10.0000 | -20.2000 | 10.0000 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 10.0000 | -20.2000 | 10.0000 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 10.0000 | -20.2000 | 10.0000 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 10.0000 | -20.2000 | 10.0000 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 10.0000 | -20.2000 | 10.0000 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 10.0000 | -20.2000 | 10.0000 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 10.0000 | -10.2000 |

# Eigenvalues:

- Eigenvalues are Poles
- They tell you *how* the system behaves

```
                                                 10000000000
----------------------------------------------------------------------------------------------------------
(s+39.31) (s+36.72) (s+32.67) (s+27.51) (s+21.69) (s+15.75) (s+10.2) (s+5.539) (s+2.181) (s+0.4234)
```

```
eig(A10)

   -39.3115       something decays as exp(-39.3115t)
   -36.7248
   -32.6698
   -27.5068
   -21.6946
   -15.7496
   -10.2000
    -5.5390
    -2.1806
    -0.4234       something decays as exp(-0.4234t)   dominant pole
```

# Eigenvectors

- Eigenvectors tell you *what* behaves that way

A is a 10x10 matrix

- It has 10 eigenvalues
- It has 10 eigenvectors

```
>> [a,b] = eig(A10)

a =   eigenvector
```

| fast | | | | | | | | | slow |
|---|---|---|---|---|---|---|---|---|---|
| **-0.1286** | -0.2459 | 0.3412 | 0.4063 | 0.4352 | 0.4255 | 0.3780 | 0.2969 | -0.1894 | **0.0650** |
| **0.2459** | 0.4063 | -0.4255 | -0.2969 | -0.0650 | 0.1894 | 0.3780 | 0.4352 | -0.3412 | **0.1286** |
| **-0.3412** | -0.4255 | 0.1894 | -0.1894 | -0.4255 | -0.3412 | -0.0000 | 0.3412 | -0.4255 | **0.1894** |
| **0.4063** | 0.2969 | 0.1894 | 0.4352 | 0.1286 | -0.3412 | -0.3780 | 0.0650 | -0.4255 | **0.2459** |
| **-0.4352** | -0.0650 | -0.4255 | -0.1286 | 0.4063 | 0.1894 | -0.3780 | -0.2459 | -0.3412 | **0.2969** |
| **0.4255** | -0.1894 | 0.3412 | -0.3412 | -0.1894 | 0.4255 | 0.0000 | -0.4255 | -0.1894 | **0.3412** |
| **-0.3780** | 0.3780 | -0.0000 | 0.3780 | -0.3780 | 0.0000 | 0.3780 | -0.3780 | -0.0000 | **0.3780** |
| **0.2969** | -0.4352 | -0.3412 | 0.0650 | 0.2459 | -0.4255 | 0.3780 | -0.1286 | 0.1894 | **0.4063** |
| **-0.1894** | 0.3412 | 0.4255 | -0.4255 | 0.3412 | -0.1894 | 0.0000 | 0.1894 | 0.3412 | **0.4255** |
| **0.0650** | -0.1286 | -0.1894 | 0.2459 | -0.2969 | 0.3412 | -0.3780 | 0.4063 | 0.4255 | **0.4352** |

diag(b) - eigenvalues

| | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| **-39.3115** | -36.7248 | -32.6698 | -27.5068 | -21.6946 | -15.7496 | -10.2000 | -5.5390 | -2.1806 | **-0.4234** |

# Matlab Simulation (Heat.m)

- Dynamic simulation of a 10-stage RC filter

Set the forcing function to zero

- Line #4

Set the initial conidtion

- Line #3

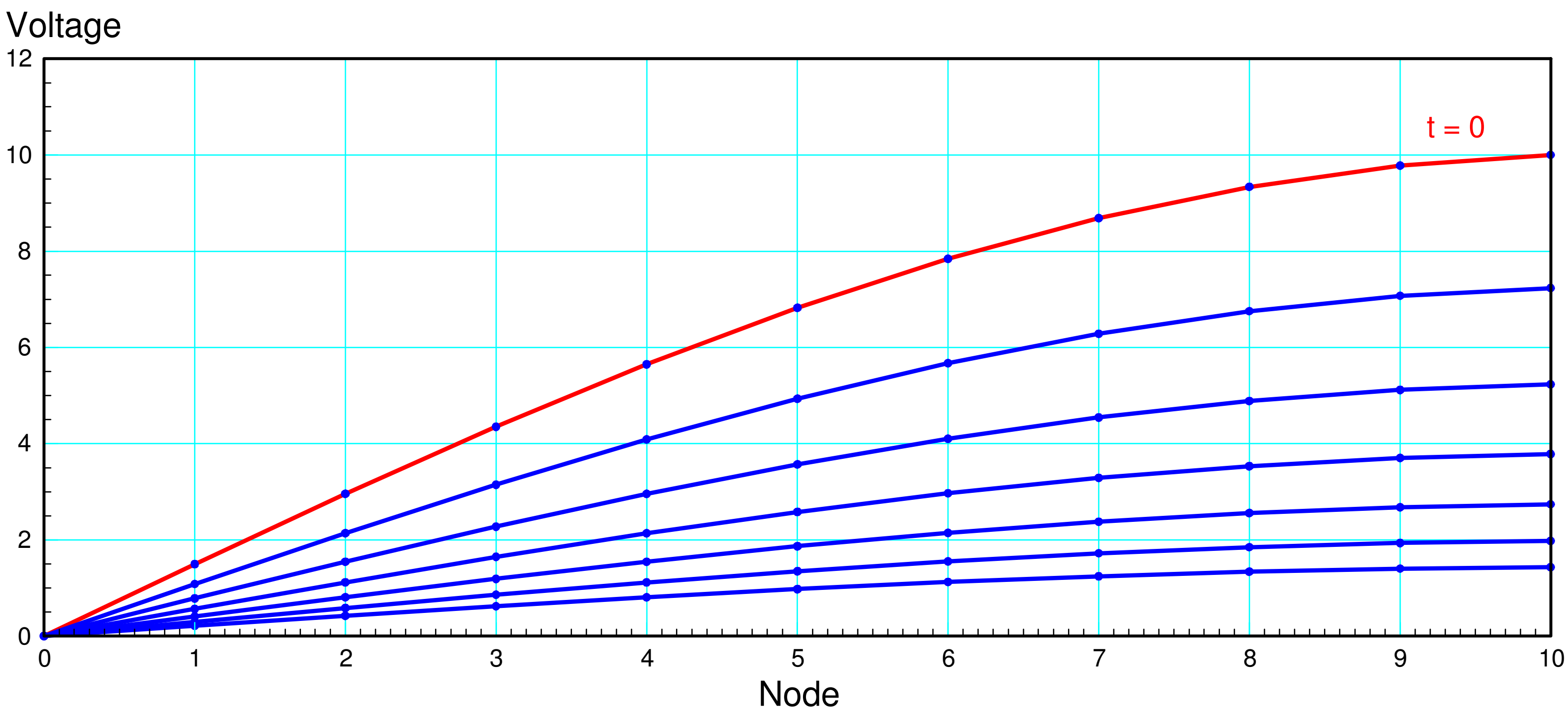
you can see the natural resposne

Editor - C:\Documents and Settings\Administrator\My Documents\MATLAB\Heat.m*

```
1   % 10-stage RC Filter
2
3   V = 10 * rand(10,1);
4   V0 = 0;
5   DATA = [V0;V];
6   dV = zeros(10,1);
7
8   dt = 0.001;
9   t = 0;
10
11  while(t < 10)
12
13     dV(1)  = 10*V0    - 20.2*V(1) + 10*V(2);
14     dV(2)  = 10*V(1)  - 20.2*V(2) + 10*V(3);
15     dV(3)  = 10*V(2)  - 20.2*V(3) + 10*V(4);
16     dV(4)  = 10*V(3)  - 20.2*V(4) + 10*V(5);
17     dV(5)  = 10*V(4)  - 20.2*V(5) + 10*V(6);
18     dV(6)  = 10*V(5)  - 20.2*V(6) + 10*V(7);
19     dV(7)  = 10*V(6)  - 20.2*V(7) + 10*V(8);
20     dV(8)  = 10*V(7)  - 20.2*V(8) + 10*V(9);
21     dV(9)  = 10*V(8)  - 20.2*V(9) + 10*V(10);
22     dV(10) = 10*V(9)  - 10.2*V(10);
23
24     V = V +  dV*dt;
25     t = t + dt;
26
```

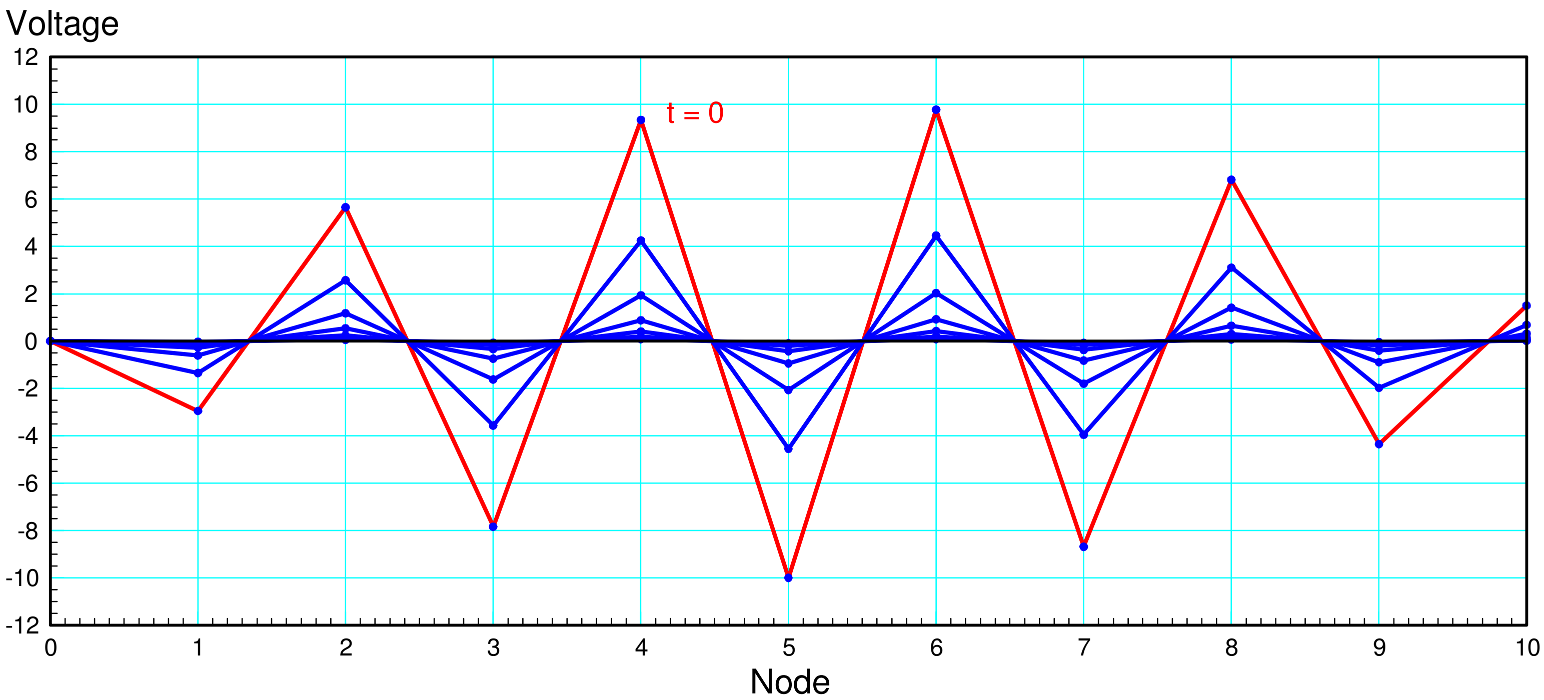
# Eigenvalue = -0.5025

- Set the forcing function to zero (V0 = 0)
- Make the initial condition proportional to the slow eigenvector
- The shape stays the same (the eigenvector)
- The amplitude decays as exp(-0.5025t)



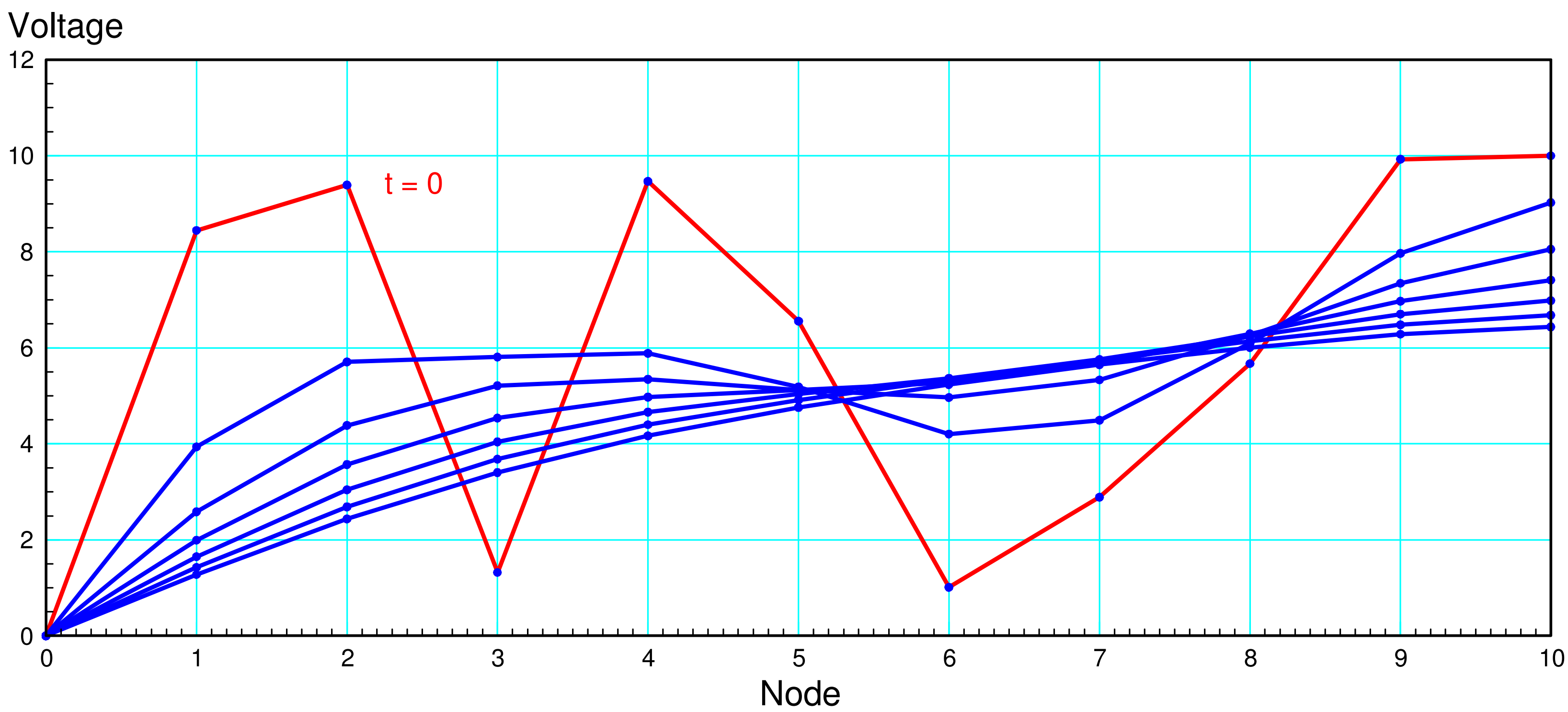
# Eigenvalue = -39.31

- Set the forcing funciton to zero (V0 = 0)
- Make the initial condition proportional to the fast eigenvector
- The shape stays the same (the eigenvector)
- The amplitude decays as exp(-39.31t)



# Random Initial Condition

- All 10 modes are excited
- The fast modes decay quickly
- Leaving the slow (dominant) mode



## Summary:

- Very few people understand what eigenvalues are
- Even fewer understand eigenvectors.

They are useful though

- Eigenvalues are poles. They tell you *how* the system behaves
- Eigenvectors tell you *what* behaves that way

For the rest of the course, we'll be only looking at eigenvalues

- Specifically, we'll look at the dominant pole
- If the dominant pole is stable, the system is stable
- If the dominant pole has a 2% settling time of 4 seconds, the system has a 2% settling time of 4 seconds (worst case)