

Capacitors & the Heat Equation

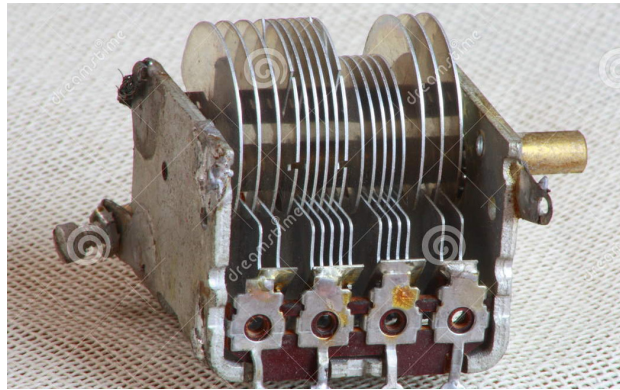


Figure 1: Parallel Plate variable capacitor (thumbs.dreamstime.com)

Topics

- Capacitors
- Heat Equation
- Eigenvalues
- Eigenvectors
- CircuitLab

Capacitors

A capacitor is a set of parallel plates¹ with the capacitance equal to

$$C = \epsilon \frac{A}{d} \text{ (Farads)}$$

where

- ϵ is the dielectric constant of the material between plates (air = $8.84 \cdot 10^{-12}$)
- A is the area of the capacitor, and
- d is the distance between plates.

The area you need for 1 Farad with plates 1mm apart is

$$1 = (8.84 \cdot 10^{-12}) \frac{A}{0.001m}$$

$$A = 113,122,171 m^2$$

The capacitor would need to have dimensions of 10.6km x 10.6km for a capacitance of 1 Farad. Typically, capacitors are in the order of micro-farads.

The charge stored in a capacitor is proportional to the voltage as

$$Q = C V$$

¹ <http://www.electronics-tutorials.ws/>

where Q is the charge in Coulombs (one Coulomb is equal to $6.242 \cdot 10^{18}$ electrons). When the voltage across a capacitor drops, the charge stored drops proportionally. This gives the fundamental equation for a capacitor:

$$I = \frac{dQ}{dt} = C \frac{dV}{dt} + V \frac{dC}{dt}$$

Assuming the capacitance is constant

$$I = C \frac{dV}{dt}$$

This means that capacitors are integrators:

$$V = \frac{1}{C} \int I \cdot dt$$

In Calculus, you will be covering integration and differentiation and how to come up with a closed-form solution to various problems. With MATLAB (i.e. in this class) you can solve using numerical methods.

Time Response of an RC filter: (Heat Equation)

Suppose you took the resistor network from before and added a capacitor at each node:

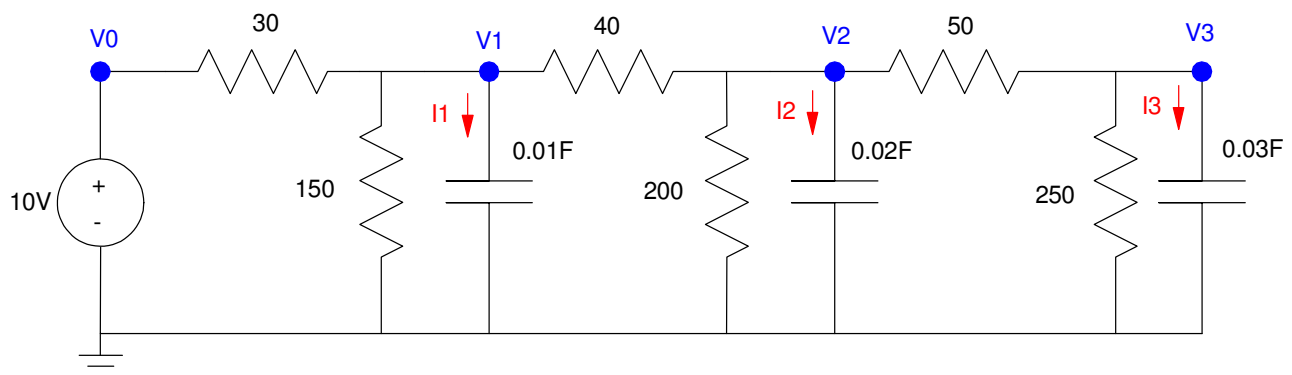


Figure 2: 3-Stage RC Filter. Write the differential equations which describe this circuit

At steady-state, the voltages are all constants. In that case, $I_1 = I_2 = I_3 = 0$

$$I_1 = 0.01F \cdot \frac{dV_1}{dt} = 0$$

The transient (time) response, however, tells you how you get to steady-state. If the capacitors start out discharged, meaning

- $V(0) = 0$
- The initial temperature = 0C

Current (heat) will start flowing in from the 100V source. This current raises the voltages (temperature) until you reach steady-state.

At any time, current must balance: the current flowing into the capacitor must equal the excess current flowing into the node. At node V1, for example:

$$I_1 = \left(\frac{V_0 - V_1}{100} \right) + \left(\frac{0 - V_1}{150} \right) + \left(\frac{V_2 - V_1}{200} \right)$$

This current changes V1 as

$$V_1 = \frac{1}{C} \int I_1 \cdot dt$$

To solve this circuit, you can use CircuitLab or Matlab. Using CircuitLab, input the circuit:

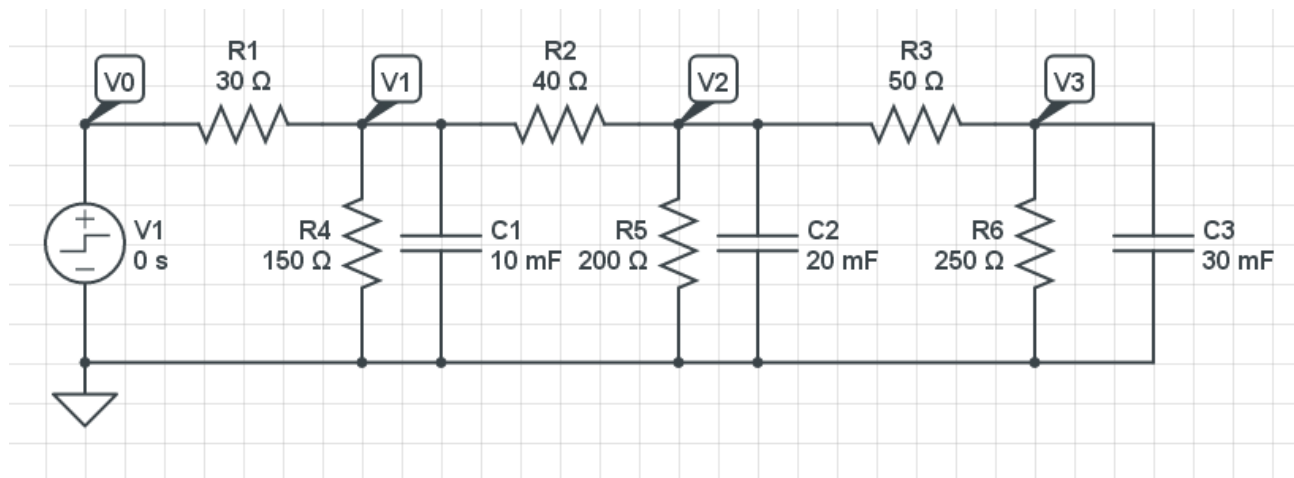


Figure 3: Input the circuit into CircuitLab

Click on Run Simulation and select Transient Response

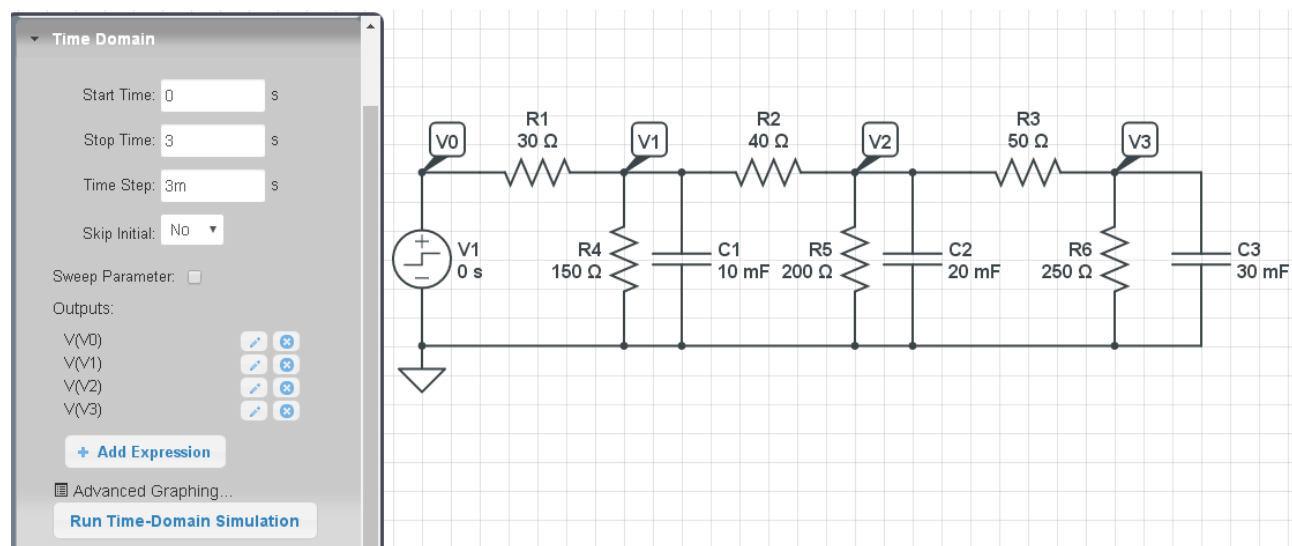


Figure 4: Run a Time-Domain simulation to find the voltages vs. time. Make the step size 1000x smaller than the stop time.

This will show you how the voltages change over time:

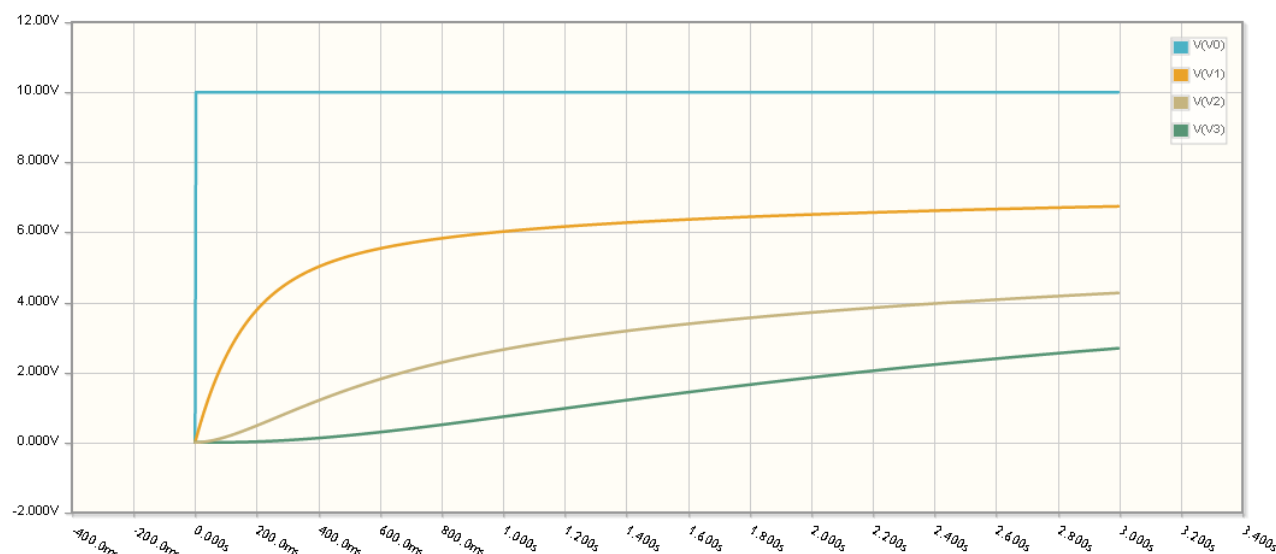


Figure 5: Transient voltages on V0, V1, V2, and V3: The capacitors are charging up to their steady-state value

What's happening is this:

- Initially, the capacitors are discharged ($V = 0$ at $t = 0$)
- When the input turns on to 10V, a current imbalance results in current flowing into the capacitors, charging them up.
- Eventually, you reach equilibrium. At this point, the current in equals the current out and no excess current remains to charge up the capacitors. At this point, you're at the steady-state solution we found last week.

In Matlab, you can repeat this simulation. First compute the currents I_1 , I_2 , and I_3 (current out = current in)

$$I_1 = \left(\frac{V_0 - V_1}{30} \right) + \left(\frac{0 - V_1}{150} \right) + \left(\frac{V_2 - V_1}{40} \right)$$

$$I_2 = \left(\frac{V_1 - V_2}{40} \right) + \left(\frac{0 - V_2}{200} \right) + \left(\frac{V_3 - V_2}{50} \right)$$

$$I_3 = \left(\frac{V_2 - V_3}{50} \right) + \left(\frac{0 - V_3}{250} \right)$$

Note that the current is equal to $C \frac{dV}{dt}$

$$0.01 \frac{dV_1}{dt} = I_1 = \left(\frac{V_0 - V_1}{30} \right) + \left(\frac{0 - V_1}{150} \right) + \left(\frac{V_2 - V_1}{40} \right)$$

$$0.02 \frac{dV_2}{dt} = I_2 = \left(\frac{V_1 - V_2}{40} \right) + \left(\frac{0 - V_2}{200} \right) + \left(\frac{V_3 - V_2}{50} \right)$$

$$0.03 \frac{dV_3}{dt} = I_3 = \left(\frac{V_2 - V_3}{50} \right) + \left(\frac{0 - V_3}{250} \right)$$

Solve for $\frac{dV_i}{dt}$

$$\frac{dV_1}{dt} = 3.333V_0 - 6.500V_1 + 2.500V_2$$

$$\frac{dV_2}{dt} = 1.250V_1 - 2.500V_2 + 1.000V_3$$

$$\frac{dV_3}{dt} = 0.667V_2 - 0.800V_3$$

Integrate to find $V_1..V_3$

$$V_1(t) = \int_0^t \frac{dV_1}{dt} \cdot d\tau$$

$$V_2(t) = \int_0^t \frac{dV_2}{dt} \cdot d\tau$$

$$V_3(t) = \int_0^t \frac{dV_3}{dt} \cdot d\tau$$

In MATLAB, start at $t = 0$ with all voltages equal to zero

```
t = 0;
dt = 0.01;
V0 = 100;
V1 = 0;
V2 = 0;
V3 = 0;

% Compute dV/dt
dV1 = 3.333*V0 - 6.500*V1 + 2.500*V2;
dV2 = 1.250*V1 - 2.500*V2 + 1.000*V3;
dV3 = 0.667*V2 - 0.800*V3;

% Integrate
V1 = V1 + dV1*dt;
V2 = V2 + dV2*dt;
V3 = V3 + dV3*dt;
```

Repeat 1000 times and you have computed the voltages for 10 seconds (1000 * dt = 10 seconds)

A for-loop works well here

```
t = [];  
y = [];  
dt = 0.01;  
  
V0 = 10;  
V1 = 0;  
V2 = 0;  
V3 = 0;  
  
for i=1:1000  
  
    dV1 = 3.333*V0 - 6.500*V1 + 2.500*V2;  
    dV2 = 1.250*V1 - 2.500*V2 + 1.000*V3;  
    dV3 = 0.667*V2 - 0.800*V3;  
  
    V1 = V1 + dV1*dt;  
    V2 = V2 + dV2*dt;  
    V3 = V3 + dV3*dt;  
  
    y = [y; V1, V2, V3];  
  
end  
  
t = [1:1000]' * dt;  
plot(t,y);  
xlabel('Time (seconds)');  
ylabel('V(t)');
```

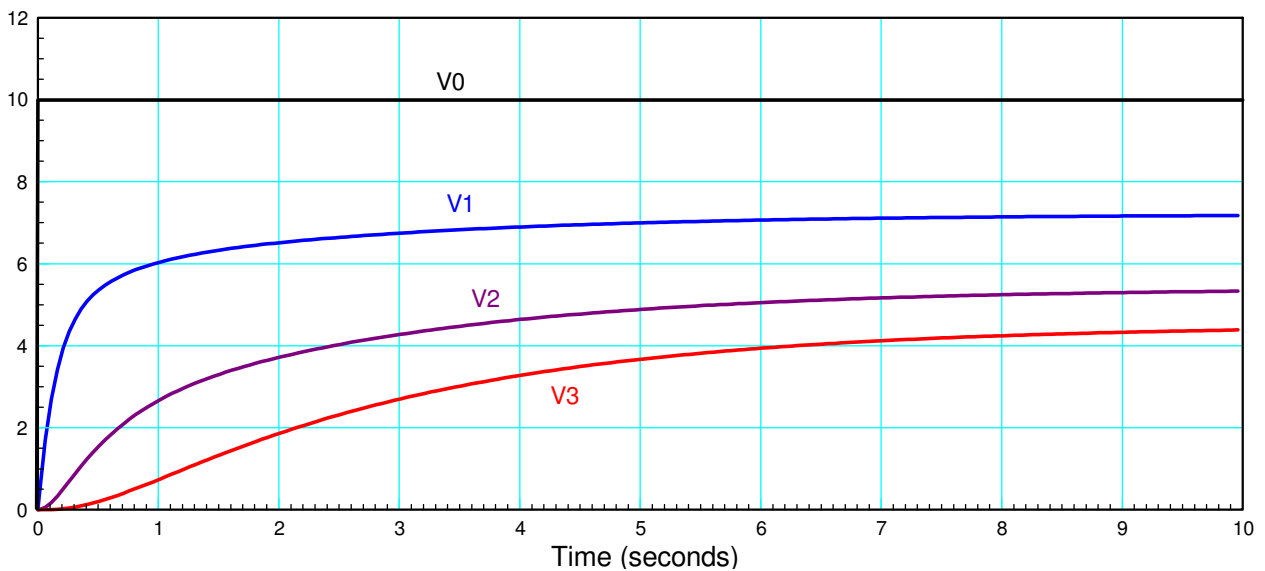


Figure 6: Computed voltages vs. time. Note that these match the results from CircuitLab

Animation in MATLAB

You can also watch the voltages change vs. time. The trick is

- Plot your function (the node voltages in this case), and
- Insert a `pause(0.01)` command to pause the MATLAB program and display the current temperature

```
t = [];
y = [];
dt = 0.01;

V0 = 10;
V1 = 0;
V2 = 0;
V3 = 0;

for i=1:1000

    dV1 = 3.333*V0 - 6.500*V1 + 2.500*V2;
    dV2 = 1.250*V1 - 2.500*V2 + 1.000*V3;
    dV3 = 0.667*V2 - 0.800*V3;

    V1 = V1 + dV1*dt;
    V2 = V2 + dV2*dt;
    V3 = V3 + dV3*dt;

    plot([0:3], [V0; V1; V2; V3], '.-');
    ylim([0,10]);
    pause(0.01);

end
```

Animation of a 10-Stage RC Filter

Take the 10-stage RC filter from last week and add a 0.1F capacitor to each node:

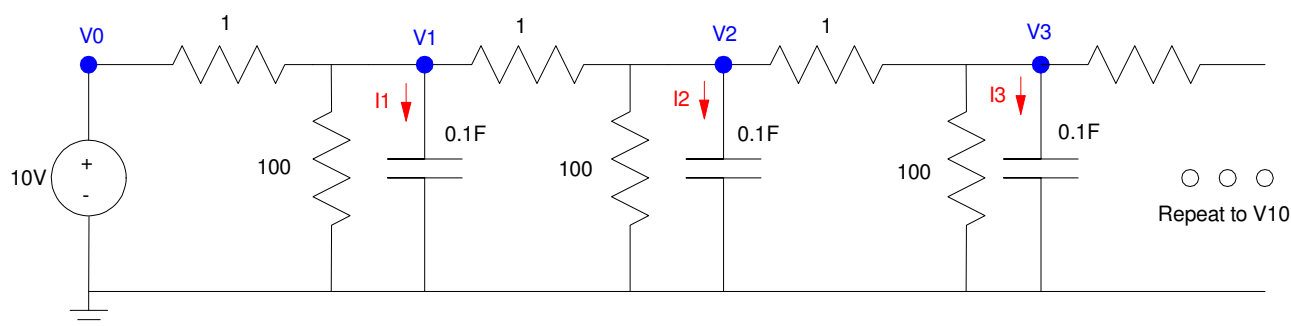


Figure 7: 10-Stage RC Filter. R and C for each stage are the same.

Simulate the response for a 10V input.

Solution: This looks like a nasty problem. It's actually not that bad. Each node V1 .. V9 has the same form. Take node V2 for example. Without the capacitor, the currents to node V2 must add to zero:

$$0 = \left(\frac{V_1 - V_2}{1} \right) + \left(\frac{V_3 - V_2}{1} \right) + \left(\frac{0 - V_2}{100} \right)$$

With the capacitor, the current sums to the current to the capacitor

$$I_{C_2} = C_2 \frac{dV_2}{dt} = \left(\frac{V_1 - V_2}{1} \right) + \left(\frac{V_3 - V_2}{1} \right) + \left(\frac{0 - V_2}{100} \right)$$

Simplifying and substituting $C_2 = 0.1F$

$$\frac{dV_2}{dt} = 10V_1 - 20.1V_2 + 10V_3$$

This same pattern holds for $V_1 \dots V_9$ (with V_1 using the input V_{in} instead of V_0)

The exception is node V_{10} where there is only one resistor attached:

$$I_{C_{10}} = C_{10} \frac{dV_{10}}{dt} = \left(\frac{V_9 - V_{10}}{1} \right) + \left(\frac{0 - V_{10}}{100} \right)$$

$$\frac{dV_{10}}{dt} = 10V_9 - 10.1V_{10}$$

This results in the matrix form of the dynamics being

In Matlab, you can animate this system as:

```
% 10-stage RC Filter

V = zeros(10,1);
dV = zeros(10,1);
V0 = 10;
dt = 0.01;
t = 0;

while(t < 100)

    dV(1) = 10*V0 - 20.1*V(1) + 10*V(2);
    dV(2) = 10*V(1) - 20.1*V(2) + 10*V(3);
    dV(3) = 10*V(2) - 20.1*V(3) + 10*V(4);
    dV(4) = 10*V(3) - 20.1*V(4) + 10*V(5);
    dV(5) = 10*V(4) - 20.1*V(5) + 10*V(6);
    dV(6) = 10*V(5) - 20.1*V(6) + 10*V(7);
    dV(7) = 10*V(6) - 20.1*V(7) + 10*V(8);
    dV(8) = 10*V(7) - 20.1*V(8) + 10*V(9);
    dV(9) = 10*V(8) - 20.1*V(9) + 10*V(10);
    dV(10) = 10*V(9) - 10.1*V(10);

    V = V + dV*dt;
    t = t + dt;

    plot([0:10], [V0;V], '.-');
    ylim([0,10]);
    pause(0.01);
end
```

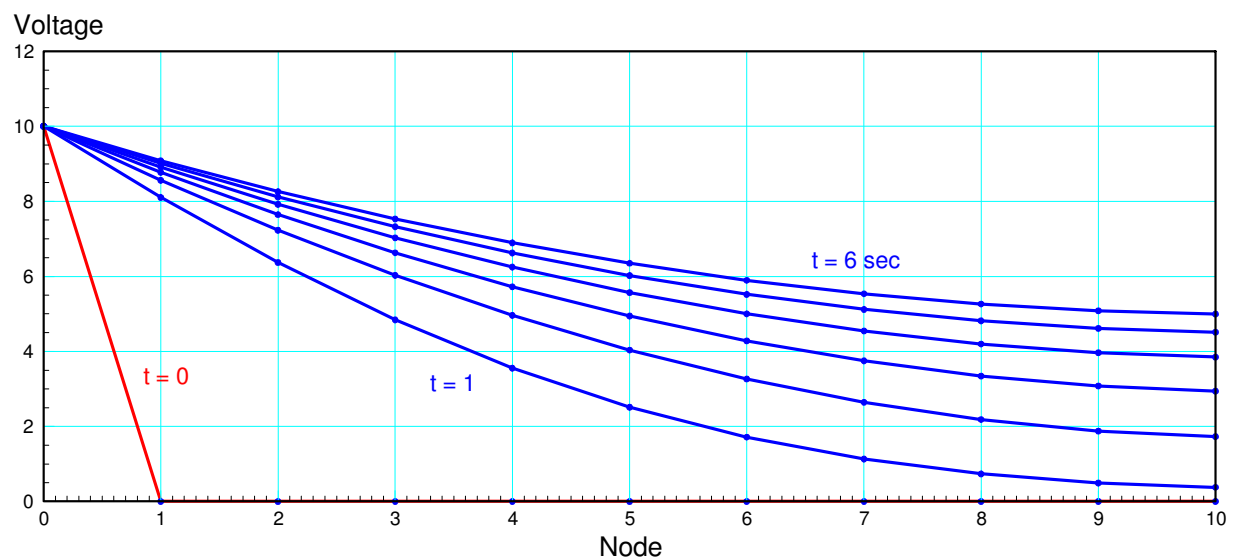


Figure 8: Temperature Along a Bar plotted every 1.00 second

Eigenvalues and Eigenvectors

Suppose you want to solve the differential equation

$$\frac{dx}{dt} = -3x$$

with

$$x(0) = x_0.$$

In Math 166, you assume $x(t)$ is in the form of

$$x(t) = e^{st}.$$

Then

$$\frac{dx}{dt} = s \cdot e^{st} = sx$$

Substituting into the above differential equation results in

$$sx = -3x$$

$$(s + 3)x = 0$$

Either

- $x(t) = 0$ (the trivial solution), or
- $s = -3$

This means $x(t)$ is in the form of

$$x(t) = a \cdot e^{-3t}$$

Plug in the initial conditions and you get

$$x(t) = x_0 \cdot e^{-3t}$$

This also works for matrices. If

$$\dot{X} = AX$$

then

$$X(t) = e^{At}X_0$$

or in terms of eigenvalues and eigenvectors

$$X(t) = a_1\Lambda_1e^{\lambda_1t} + a_2\Lambda_2e^{\lambda_2t} + \dots a_{10}\Lambda_{10}e^{\lambda_{10}t}$$

where

- Λ_i is the i th eigenvector,
- λ_i is the i th eigenvalue, and
- a_i are constants determined by the initial condition.

Eigenvalues tell you how the system behaves

Eigenvectors tell you what behaves that way.

If $X(0)$ is equal to an eigenvector, then only that one mode is excited.

- The shape of $x(t)$ remains the same (only one eigenvector is excited)
- $x(t)$ then goes to zero according to its eigenvalue.

If $X(0)$ excites multiple eigenvectors, then $X(t)$ will be the combination of all its eigenmodes.

Example: Take for example, the 10-stage RC filter. In matrix form

$$\begin{bmatrix} \dot{V}_1 \\ \dot{V}_2 \\ \dot{V}_3 \\ \dot{V}_4 \\ \dot{V}_5 \\ \dot{V}_6 \\ \dot{V}_7 \\ \dot{V}_8 \\ \dot{V}_9 \\ \dot{V}_{10} \end{bmatrix} = \begin{bmatrix} -20.1 & 10 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 10 & -20.1 & 10 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 10 & -20.1 & 10 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 10 & -20.1 & 10 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 10 & -20.1 & 10 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 10 & -20.1 & 10 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 10 & -20.1 & 10 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 10 & -20.1 & 10 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 10 & -20.1 & 10 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 10 & -10.1 \end{bmatrix} \begin{bmatrix} V_1 \\ V_2 \\ V_3 \\ V_4 \\ V_5 \\ V_6 \\ V_7 \\ V_8 \\ V_9 \\ V_{10} \end{bmatrix} + \begin{bmatrix} 10 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix} V_0$$

In Matlab, you can input this 10x10 system as

```
>> A = zeros(10,10);
>> for i=1:9
A(i,i) = -20.1;
A(i,i+1) = 10;
A(i+1,i) = 10;
end
>> A(10,10) = -10.1;
>> A
```

-20.1000	10.0000	0	0	0	0	0	0	0	0
10.0000	-20.1000	10.0000	0	0	0	0	0	0	0
0	10.0000	-20.1000	10.0000	0	0	0	0	0	0
0	0	10.0000	-20.1000	10.0000	0	0	0	0	0
0	0	0	10.0000	-20.1000	10.0000	0	0	0	0
0	0	0	0	10.0000	-20.1000	10.0000	0	0	0
0	0	0	0	0	10.0000	-20.1000	10.0000	0	0
0	0	0	0	0	0	10.0000	-20.1000	10.0000	0
0	0	0	0	0	0	0	10.0000	-20.1000	10.0000
0	0	0	0	0	0	0	0	10.0000	-10.1000

The eigenvalues of the 10x10 matrix are:

```
>> eig(A)
```

-39.2115
-36.6248
-32.5698
-27.4068
-21.5946
-15.6496
-10.1000
-5.4390
-2.0806
-0.3234

Eigenvalues tell you how the system behaves.

- There is a fast mode which decays as

$$x(t) = e^{-39.21t}$$

- There is a slow mode which decays as

$$x(t) = e^{-0.3234t}$$

- There are eight other modes as well

Eigenvectors tell you what behaves that way:

```
>> [a,b] = eig(A);
>> a

a =

-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

>> eig(A) '

-39.2115   -36.6248   -32.5698   -27.4068   -21.5946   -15.6496   -10.1000    -5.4390    -2.0806    -0.3234
```

The first eigenvector (column of a) is the initial condition which decays as

$$V(t) = e^{-39.2115t}$$

If you set the input to zero and the initial conditions to this (or a multiple of this) eigenvector,

- The shape stays the same and
- The amplitude decays as $e^{-39.21t}$

```
>> V0 = a(:,1) * 200

V0 =

-25.7283
 49.1706
-68.2438
 81.2533
-87.0431
 85.0987
-75.5929
 59.3703
-37.8725
 13.0095
```

Change the previous code to

```
% 10-stage RC Filter

V = [    -2.9558
        5.6490
       -7.8402
        9.3348
      -10.0000
        9.7766
       -8.6845
        6.8208
       -4.3510
        1.4946    ];

dV = zeros(10,1);
V0 = 0;
dt = 0.001;
t = 0;

while(t < 1)

    dV(1) = 10*V0 - 20.1*V(1) + 10*V(2);
    dV(2) = 10*V(1) - 20.1*V(2) + 10*V(3);
    dV(3) = 10*V(2) - 20.1*V(3) + 10*V(4);
    dV(4) = 10*V(3) - 20.1*V(4) + 10*V(5);
    dV(5) = 10*V(4) - 20.1*V(5) + 10*V(6);
    dV(6) = 10*V(5) - 20.1*V(6) + 10*V(7);
    dV(7) = 10*V(6) - 20.1*V(7) + 10*V(8);
    dV(8) = 10*V(7) - 20.1*V(8) + 10*V(9);
    dV(9) = 10*V(8) - 20.1*V(9) + 10*V(10);
    dV(10) = 10*V(9) - 10.1*V(10);

    V = V + dV*dt;
    t = t + dt;

    plot([0:10], [V0;V], '.-');
    ylim([-90,90]);
    pause(0.01);
end

>> plot([0;V0])
```

The response at $t=0$, 1, and 2 seconds looks like:

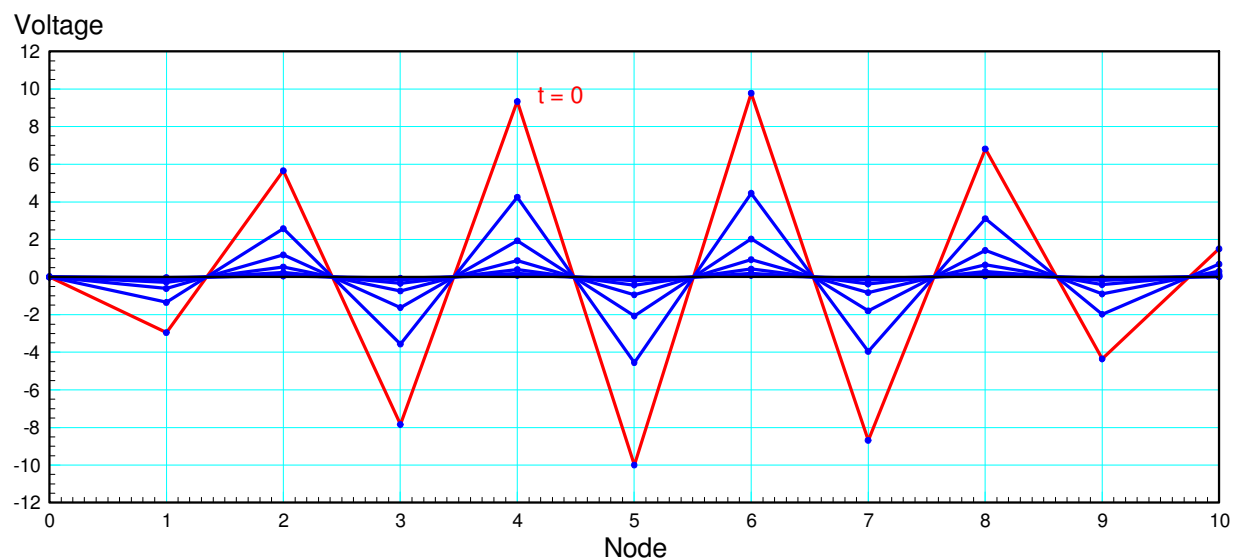


Figure 9: Node voltages plotted every 20ms for the fast eigenvector

Note that

- The shape of the curve remains the same as defined by the eigenvector that's excited
- This mode decays very quickly as

$$V(t) = V_0 e^{-39.21t}$$

If you excite the slow eigenvector instead, the system decays very slowly. The slow eigenvector is

```
>> a(:,10)/max(abs(a(:,10)))*10
```

```
1.4946
2.9558
4.3510
5.6490
6.8208
7.8402
8.6845
9.3348
9.7766
10.0000
```

If this is the initial condition then the response is

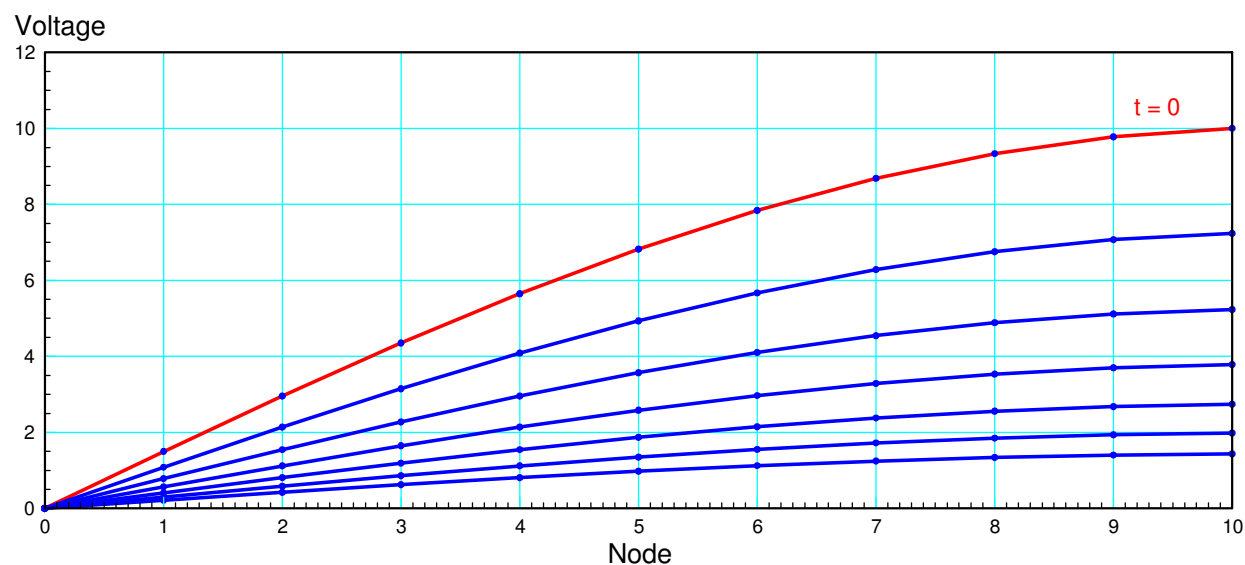


Figure 10: Voltages plotted every 1.00 second for the slow eigenvector

Note that

- The shape of the curve remains the same as defined by the eigenvector that's excited
- This mode decays very slowly as

$$V(t) = V_0 e^{-0.33234t}$$

If you have a random initial condition, then all ten eigenmodes will be excited. Quickly, the fast modes die out. All you're left with then is the slow mode.

For example, let the initial condition be:

```
>> V0 = 10*rand(10,1)
```

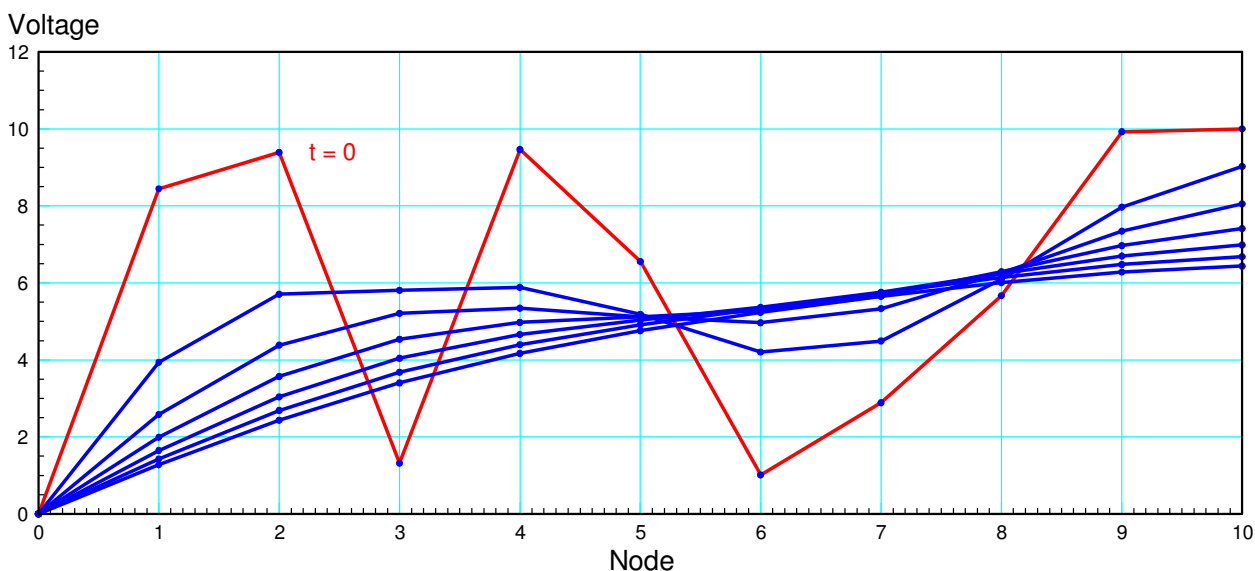


Figure 11: Node voltages plotted every 100ms with a random initial condition (exciting all eigenvectors)

In this case

- The initial condition (shown in red) excites multiple eigenvectors
- After about 0.4 seconds, the fast modes die out and you're left with the slow mode.

Summary

When you have a circuit with resistors and capacitors,

- It takes time for the capacitors to charge and discharge,
- You need to use differential equations to describe the behaviour of this circuit, and
- Each capacitor adds a 1st-order coupled differential equation.

Furthermore, eigenvalues and eigenvectors are actually useful:

- Eigenvalues tell you how the circuit will behave.
- Eigenvectors tell you what behaves that way

If your initial conditions are proportional to an eigenvector, then only one eigenmode will be excited.

If your initial condition excites multiple eigenvectors (the usual case),

- The fast modes decay quickly,
- Leaving the slow (dominant) mode.

This slow mode is the eigenvector associated with the slow eigenvalue.