# Capacitors & The Heat Equation

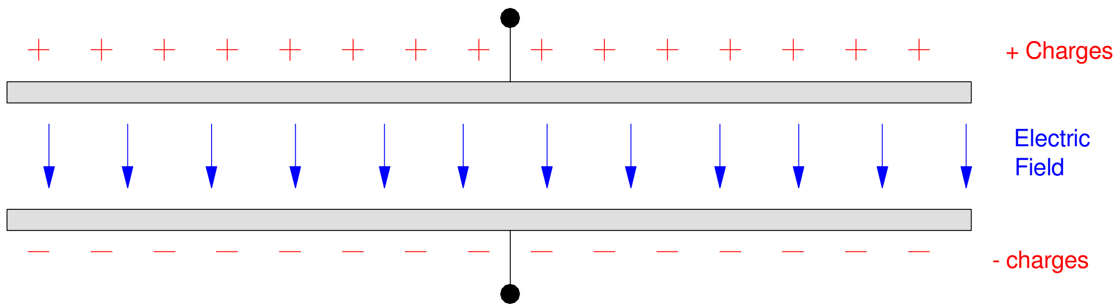
## Capacitors

A capacitor is a set of parallel plates[1] with the capacitance equal to

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

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.




A capacitor is two parallel plates. They store energy in the electric field between the plates

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

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

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

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 if micro-farads.

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

$$Q = C \cdot V$$

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

[1] http://www.electronics-tutorials.ws/

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

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.

### Capacitors and Energy Storage

Capacitors store energy when charged. The energy stored is

$$P = VI = V \cdot C \frac{dV}{dt}$$

$$E = \int P\, dt = \int \left( VC \frac{dV}{dt} \right) dt = \int (VC) dV$$

$$E = \frac{1}{2} C V^2$$

The energy stored in a capacitor isn't large - but it is there. To put this in perspective, the energy stored in a 1F capacitor compared to other common items are as follows:

| Item | Energy | Cost | $ / MJ |
|---|---|---|---|
| 1 pound Wyoming Coal | 3,600,000 Joules | $0.028 | $0.0078 |
| 1 pound ND Lignite | 1,565,217 Joules | $0.017 | $0.0108 |
| 1 pint of gasoline | 15,000,000 Joules | $0.37 | $0.0247 |
| Lithium battery (D cell) | 246,240 Joules | $22 | $89.43 |
| 1F Capacitor (5V) | 12.5 Joules | $2.87 | $229,600 |

In theory, you *could* build an electric car using capacitors to store the energy. This would have advantages:

- It would take seconds to charge rather than hours,
- The efficiency of capacitors is near 100% - much better than chemical batteries.
- The number of charge / recharge cycles is large (almost unlimited),

But,

- Electric cars would cost 2,500 times more for the same energy storage.

Instead, capacitors are used in electronic circuits to provide energy for a short period of time - often on the order of milliseconds or microseconds. When dealing with such short time frames, even 1mJ can be significant. For example, 1mJ may not seem like a lot, but 1mJ dissipated offer 1us is 1000W

$$P = \frac{\text{Joules}}{\text{second}} = \frac{1mJ}{1\mu s} = 1000W$$

## Numerical Integration and Capacitors

Capacitors are inherently integrators:

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

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

Likewise, any circuit which has capacitors implements a differential equation. To solve for the voltages, just integrate the current to each capacitor.

In Calculus, you studies methods to solve differential equations. In Circuits II and Signals and Systems, you will study LaPlace transforms - an easier way to solve differential equations. Here, in Circuits I, we will focus on

- Obtaining the differential equations that describe RC circuits, and
- Solving for the voltages using numerical methods and Matlab.

The latter is also how programs like CircuitLab solve these problems.

Numerical integration covers how to find the integral of a function using a computer program. There are many different forms of numerical integration, and all are off slightly. Three common forms are

- Euler Integration
- Trapezoid Rule
- Runge Kutta Integration

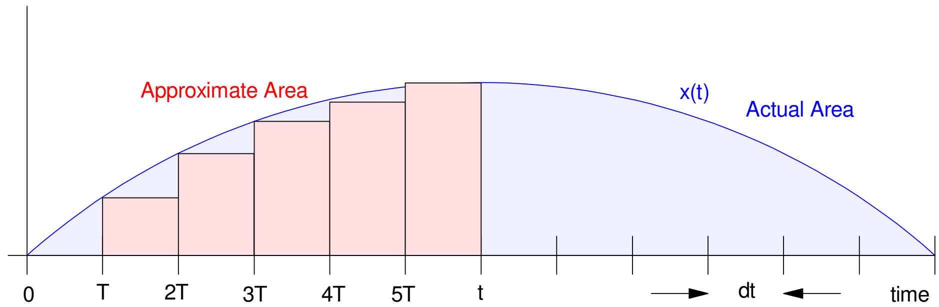
**Euler Integration:** Suppose you want to find the integral of x(t)

$$y(t) = \int x(t)\,dt$$

The integral is the area under the curve. One way to approximate this area is to

- Sample x(t) every T seconds,
- Use rectangles to approximate the area each T seconds, and
- Sum up the area of each rectangle.

Euler integration is the simplest and least accurate of these three forms. It's simple though and not too bad if you keep the sampling time (dt) small.



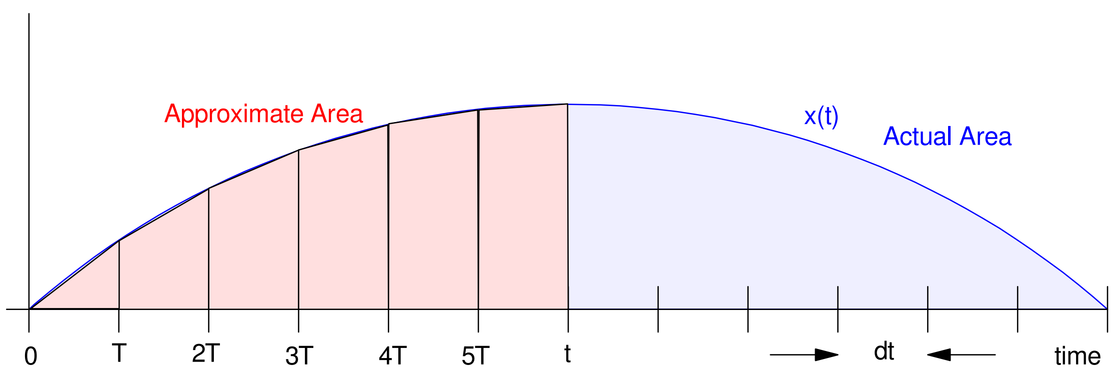

Euler Integration: Approximate the area under the curve with rectangles

The nice thing about Euler Integration is it's simple and requires no memory. The integral at time t is

$$y(t) = y(t-T) + x(t) \cdot dt$$

**Trapezoid (Bilinear) Integration:** A significantly improved version of integration approximates the area under the curve with trapezoids. This is slightly more complicated in that you need to remember the previous value of x(t):

$$y(t) = y(t-T) + \left(\frac{x(t)+x(t-T)}{2}\right) \cdot dt$$




Trapezoid Integration: Approximate the area under the curve with trapezoids

**Runge Kutta Integration:** Even better results are obtained if you also include points within each sample and curve fit a parabola to the curve, and compute the area under the parabola (3rd order Runge Kutta) or a cubic (4th order Runge Kutta), or 4th-order polynomial (5th order Runge Kutta).

Here, we will stick with Euler Integration, meaning

To find the voltage across a capacitor

- Compute the current to the capacitor, and
- Integrate using Euler integration:

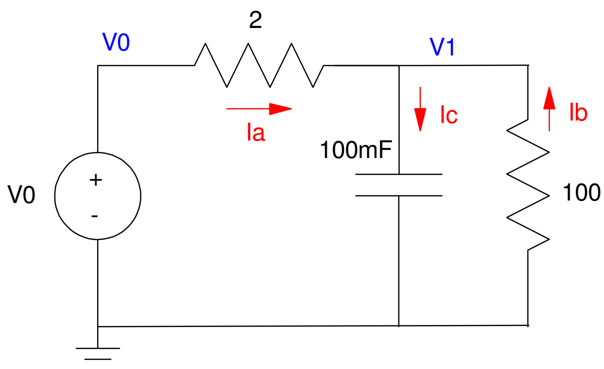
```
dV = I / C
V = V + dV * dt
```

**Example 1: 1-Stage RC Circuit**

Assume

$$V_0(t) = 10u(t) = \begin{cases} 0V & t < 0 \\ 10V & t > 0 \end{cases}$$

Find V1(t)




Solution: V1 is

$$V_1(t) = \frac{1}{C} \int I_c \, dt$$

$$I_c = C \frac{dV_1}{dt}$$

To find V1, you first have to find Ic

$$I_c = I_a + I_b$$

$$I_c = \left(\frac{V_0 - V_1}{2}\right) + \left(\frac{0 - V_1}{100}\right)$$

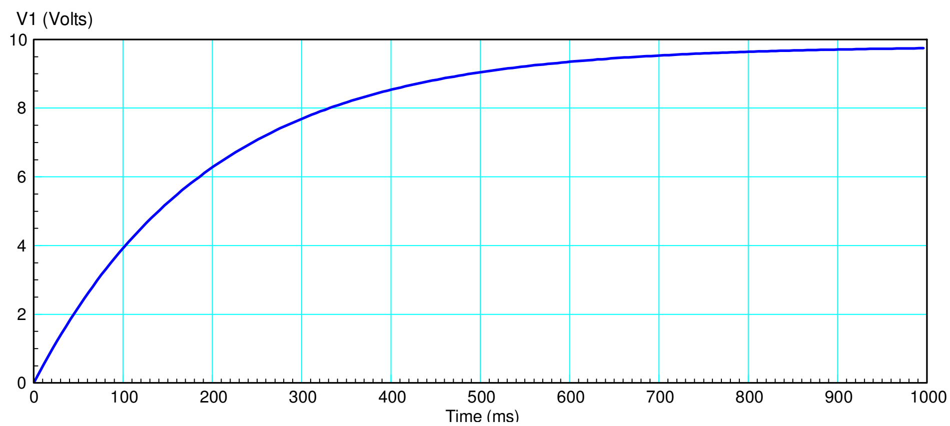
$$\frac{dV}{dt} = \frac{1}{C} I_c$$

meaning

$$\frac{dV_1}{dt} = -5.1\, V_1 + 5\, V_0$$

In Matlab, you can integrate and plot V1 vs time:

```
% 1-stage RC Filter

V = 0;
V0 = 10;
dt = 0.01;
t = 0;
Y = [];

while(t < 1)
   dV  = -5.1*V + 5*V0;
   V = V +  dV*dt;
   t = t + dt;
   Y = [Y ; V];
end

t = [1:length(Y)]' * dt;

plot(t, Y);
```

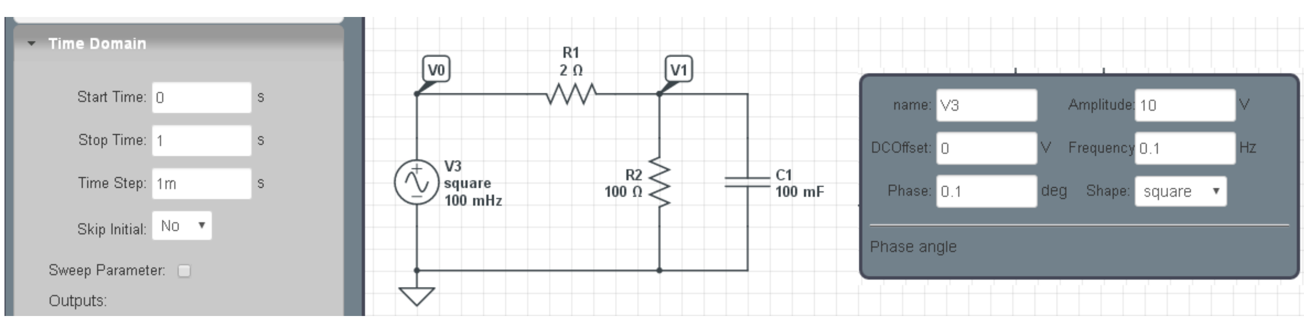



Voltage V1 vs. Time
Note that this shape is very typical of RC filters:  you are charging up the capacitor

In CircuitLab, you can generate the same plot with CircutLab

- Change the input to a square wave, 0.1Hz, 0.1 degree phase shift
- Run a time-domain simulation for 1 second

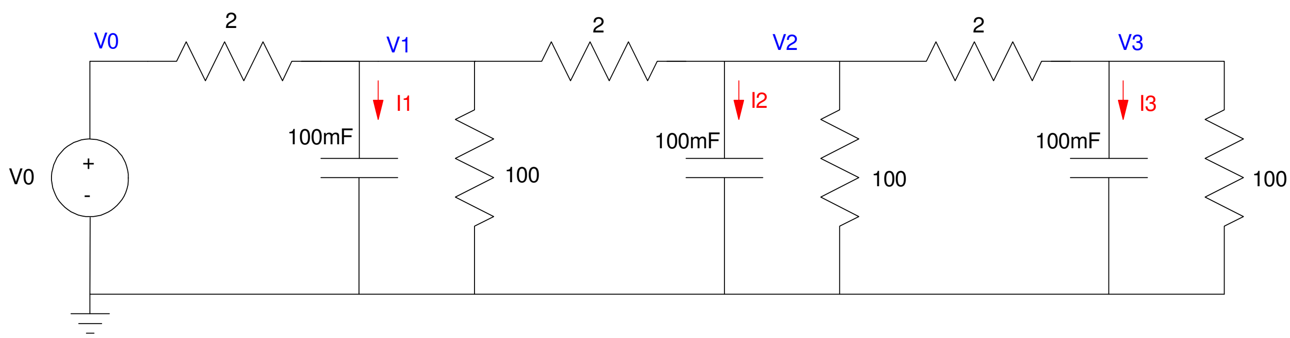
CircuitLab will run the same numerical integration scheme we did in Matlab (except probably with trapezoid rule or Runge Kutta integration).



**Example 2: 3-Stage RC Filter**

Next, find the voltages for the following 3-stage RC filter when

$$V_0(t) = 10u(t)$$




3-Stage RC Filter

To solve for V1, V2, and V3

- Determine the currents to each capacitor: I1, I2, and I3
- From this find dV/dt,
- From this, integrate to find V(t)

Step 1: Determine I1, I2, and I3. From *current in = current out*

$$C_1 \frac{dV_1}{dt} = I_1 = \left(\frac{V_0 - V_1}{2}\right) + \left(\frac{0 - V_1}{100}\right) + \left(\frac{V_2 - V_1}{2}\right)$$

$$C_2 \frac{dV_2}{dt} = I_2 = \left(\frac{V_1 - V_2}{2}\right) + \left(\frac{0 - V_2}{100}\right) + \left(\frac{V_3 - V_2}{2}\right)$$

$$C_3 \frac{dV_3}{dt} = I_3 = \left(\frac{V_2 - V_3}{2}\right) + \left(\frac{0 - V_3}{100}\right)$$

Next, determine dV/dt

$$\frac{dV_1}{dt} = 5V_0 - 10.1V_1 + 5V_2$$

$$\frac{dV_2}{dt} = 5V_1 - 10.1V_2 + 5V_3$$

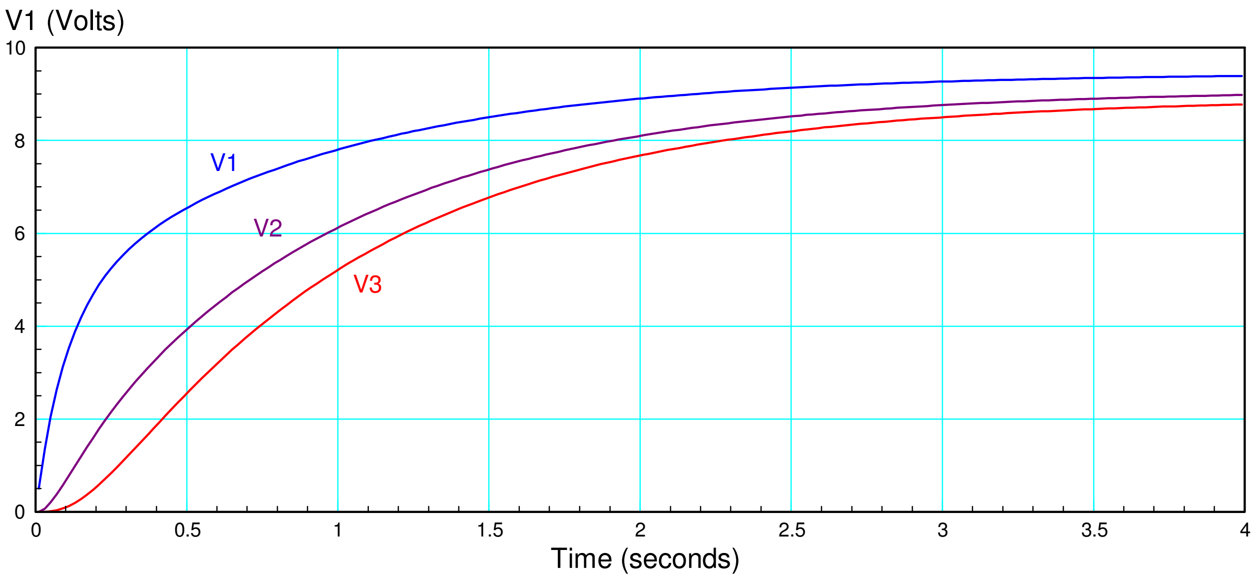
$$\frac{dV_3}{dt} = 5V_2 - 5.1V_3$$

Now integrate using Euler integration and Matlab.

- Note: You will get the same results if you run a Time Domain simulation in CircuitLab

In Matlab:

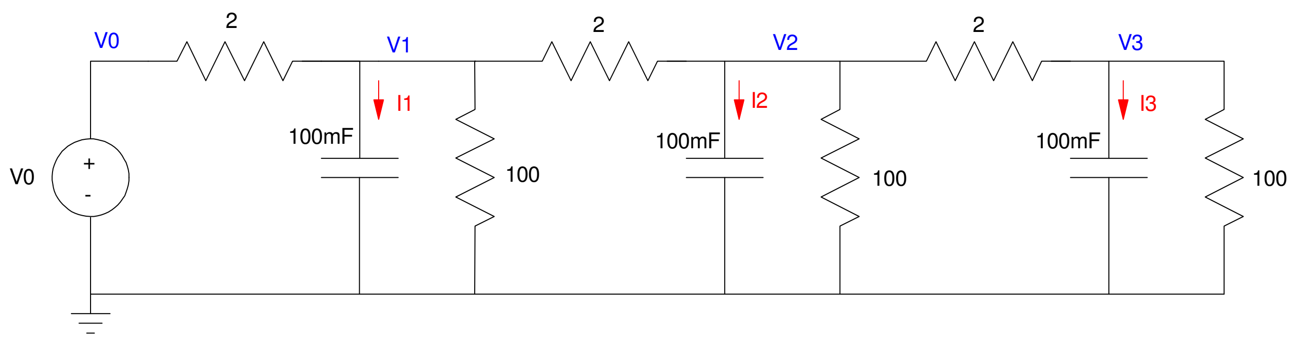
```
% 3-stage RC Filter

V0 = 10;
V1 = 0;
V2 = 0;
V3 = 0;
dt = 0.01;
t = 0;
Y = [];

while(t < 4)
   dV1  = 5*V0 -10.1*V1 + 5*V2;
   dV2  = 5*V1 -10.1*V2 + 5*V3;
   dV3  = 5*V2 -5.1*V3;

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

   t = t + dt;

   Y = [Y ; [V1, V2, V3] ];
end

t = [1:length(Y)]' * dt;

plot(t, Y);
```




Votlages for a 3-stage RC filter

## Case 3: 10-Stage RC Filter: Heat Equation

Now repeat for 10 stages




Note that stage 1 .. 9 all have the same differential equation except for the last stage

$$\frac{dV_2}{dt} = 5V_1 - 10.1V_2 + 5V_3$$

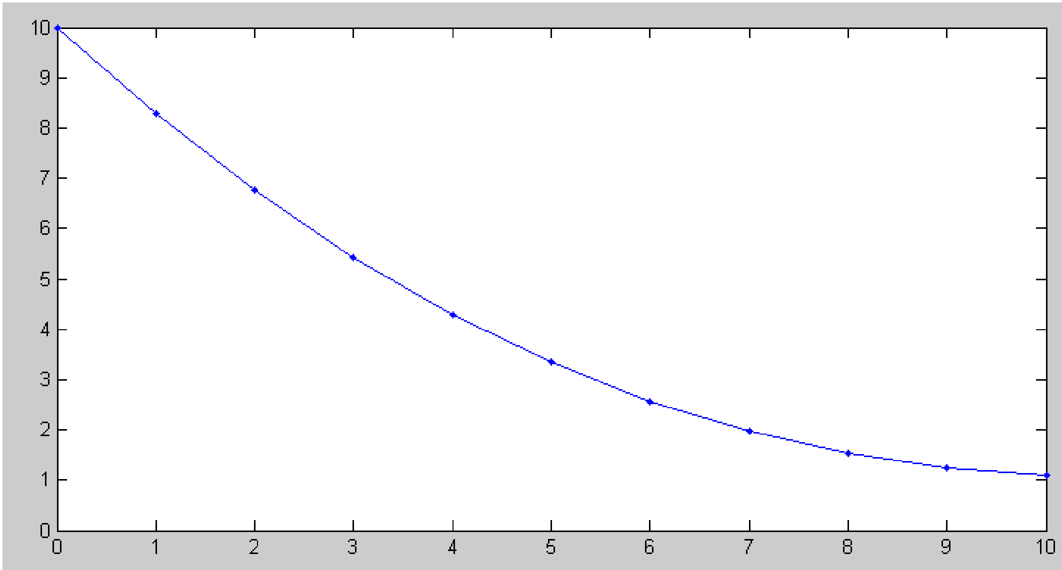
The only different one will be the last stage (which only has a single 2-Ohm resistor attached to it)

$$\frac{dV_{10}}{dt} = 5V_9 - 5.1V_{10}$$

In Matlab:

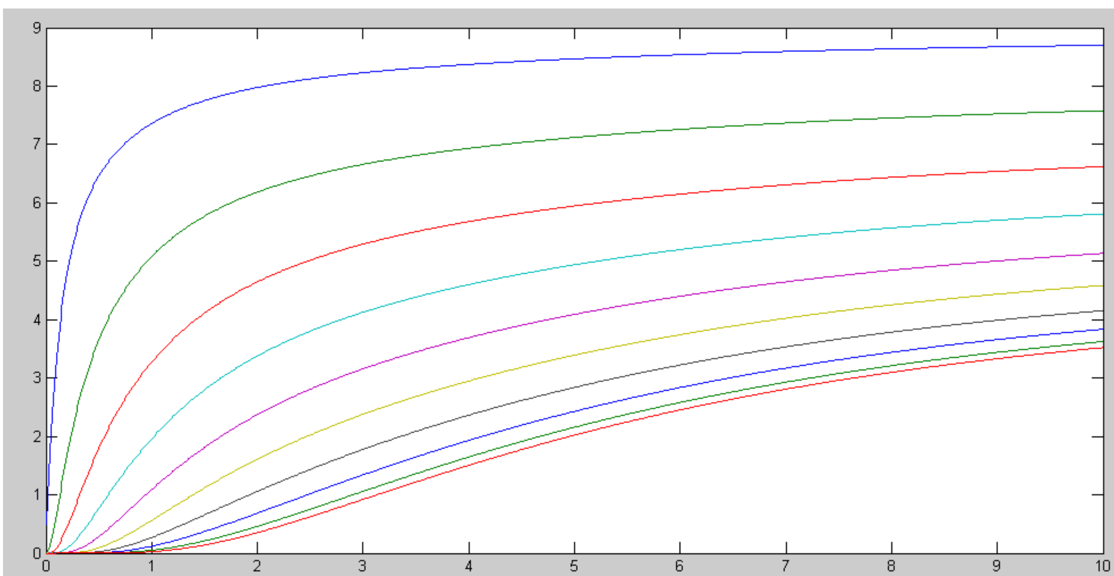
```
% 10-stage RC Filter

V0 = 10;
V = zeros(10,1);
dV = 0*V;
dt = 0.01;
t = 0;
Y = [];

while(t < 10)
   dV(1)  = 5*V0 -10.1*V(1) + 5*V(2);
   for i=2:9
       dV(i) = 5*V(i-1) - 10.1*V(i) + 5*V(i+1);
   end
   dV(10) = 5*V(9) - 5.1*V(10);

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

   N = [0:10];
   plot(N, [V0; V], 'b.-');
   ylim([0,10]);
   pause(0.01);

   Y = [Y ; [V'] ];
end

pause(5);
t = [1:length(Y)]' * dt;
plot(t, Y);
```

For the first 10 seconds, this program shows the voltages along the circuit




Voltages V1, V2, ... V10 at each time point

Then, the final voltages vs. time are displayed




Votlages V1.. V10 vs. Time

Note that this program simulates

- The charging of 10 capacitors in an RC circuit,
- The temperature along a metal rod as it heats up when the base is connected to 10 degrees

Coupled fist-order differential equations like this also describe heat flow - hence differential equations of this form are caller *the heat equation*

### Eigenvalues and Eigenvectors

The dynamics for the 10-stage RC filter are:

$$\frac{dV_1}{dt} = \dot{V}_1 = 5V_0 - 10.1V_1 + 5V_2$$

$$\frac{dV_2}{dt} = \dot{V}_2 = 5V_1 - 10.1V_2 + 5V_3$$

$$\vdots$$

$$\frac{dV_9}{dt} = \dot{V}_9 = 5V_8 - 10.1V_9 + 5V_{10}$$

$$\frac{dV_{10}}{dt} = \dot{V}_{10} = 5V_9 - 5.1V_{10}$$

In matrix form, this can be written as

$$\dot{V} = AV + BV_0$$

or

$$\begin{bmatrix} \dot{V}_1 \\ \dot{V}_2 \\ \dot{V}_3 \\ \dot{V}_4 \\ \dot{V}_5 \\ \dot{V}_6 \\ \dot{V}_7 \\ \dot{V}_8 \\ \dot{V}_9 \\ \dot{V}_{10} \end{bmatrix} = \begin{bmatrix} -10.1 & 5 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 5 & -10.1 & 5 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 5 & -10.1 & 5 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 5 & -10.1 & 5 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 5 & -10.1 & 5 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 5 & -10.1 & 5 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 5 & -10.1 & 5 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 5 & -10.1 & 5 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 5 & -10.1 & 5 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 5 & -5.1 \end{bmatrix} \begin{bmatrix} V_1 \\ V_2 \\ V_3 \\ V_4 \\ V_5 \\ V_6 \\ V_7 \\ V_8 \\ V_9 \\ V_{10} \end{bmatrix} + \begin{bmatrix} 5 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix} V_0$$

The *eigenvalues* of matrix A tell you *how* the system behaves. Matrix A is a 10x10 matrix:

```
A = zeros(10,10);
for i=1:9
   A(i,i) = -10.1;
   A(i,i+1) = 5;
   A(i+1,i) = 5;
   end

A(10,10) = -5.1;

A
```

```
A =

  -10.1000    5.0000         0         0         0         0         0         0         0         0
    5.0000  -10.1000    5.0000         0         0         0         0         0         0         0
         0    5.0000  -10.1000    5.0000         0         0         0         0         0         0
         0         0    5.0000  -10.1000    5.0000         0         0         0         0         0
         0         0         0    5.0000  -10.1000    5.0000         0         0         0         0
         0         0         0         0    5.0000  -10.1000    5.0000         0         0         0
         0         0         0         0         0    5.0000  -10.1000    5.0000         0         0
         0         0         0         0         0         0    5.0000  -10.1000    5.0000         0
         0         0         0         0         0         0         0    5.0000  -10.1000    5.0000
         0         0         0         0         0         0         0         0    5.0000   -5.1000
```

A has 10 eigenvalues and 10 eigenvectors:

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

M = Eigenvectors:

   -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

V = Eigenvalues:

  -19.6557  -18.3624  -16.3349  -13.7534  -10.8473   -7.8748   -5.1000   -2.7695   -1.0903   -0.2117
```

The eigenvalues tell you *how* the mode behaves

The eigenvector tells you *what* behaves that way.

For example, assume

- V0 = 0, and
- V(0) = the last column (show in red: the eigenvector associated with the slow eigenvalue)

Then, V(t) will be

$$V(t) = V_0\, e^{-0.2117t}$$

In Matlab, you can see this by

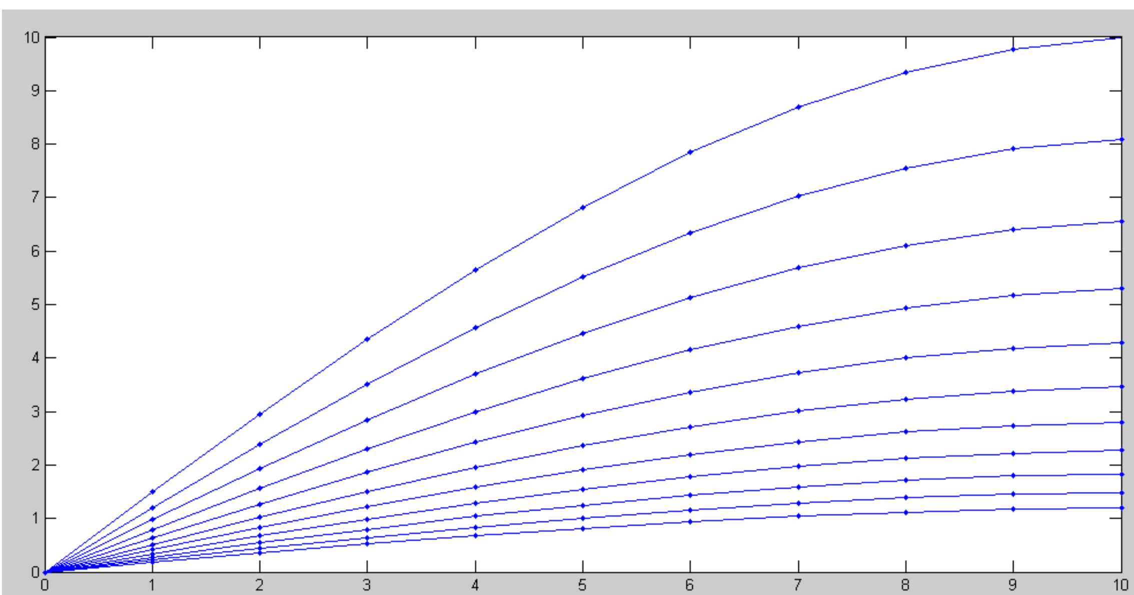
- Setting V0 = 0, and
- Changing the initial conditions

```
V0 = M(:,10) * 10 / max(M(:,10))
```

```
% 10-stage RC Filter

V0 = 0;
V = [    1.4946
    2.9558
    4.3510
    5.6490
    6.8208
    7.8402
    8.6845
    9.3348
    9.7766
   10.0000];
dV = 0*V;
dt = 0.01;
t = 0;
Y = [];

while(t < 10)
   % rest of code stays the same
```

The result is as follows:

- The shape of the curve stays the same:  the shape is the slow eigenvector (what behaves this way)
- The amplitude decays slowly, as exp(-0.2117t)  ( the slow eigenvalue, how it behaves )




Volages plotted every 1.00 second when the initial condition is the slow eigenvector

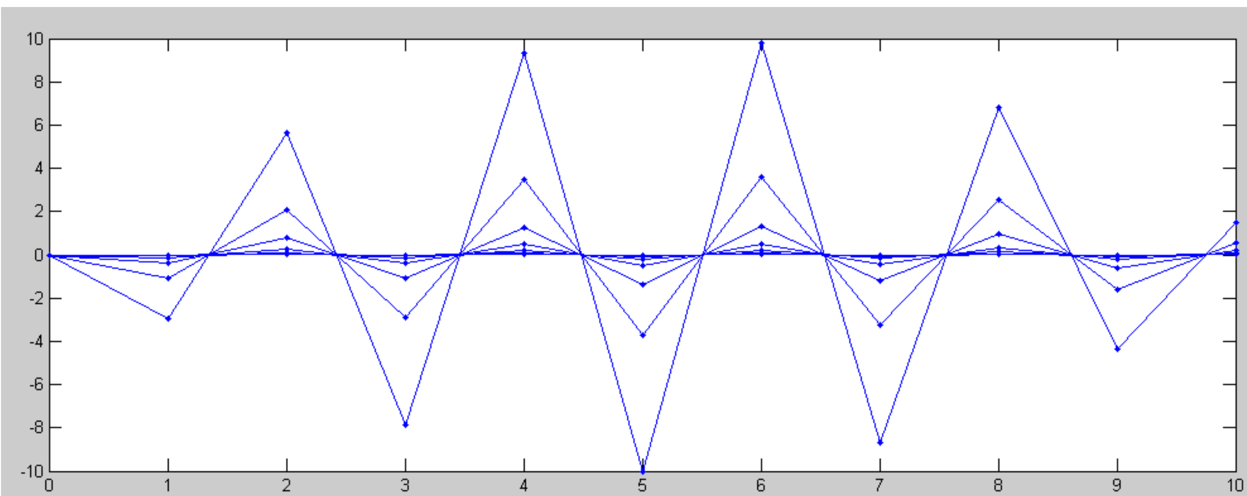
If you make the initial condition the fast eigenvector (shown in blue)

```
V0 = M(:,1) * 10 / max(abs(M(:,1)))
```

```
% 10-stage RC Filter

V0 = 0;
V = [ -2.9558
    5.6490
   -7.8402
    9.3348
  -10.0000
    9.7766
   -8.6845
    6.8208
   -4.3510
    1.4946];
dV = 0*V;
dt = 0.01;

etc...
```

the response is as shown below. Note

- The shape of the cuve is the same for all time: only one eigenvector is present
- The amplitude drops quickly: as exp(-19.65t) (the fast eigenvalue)




Voltages plotted every 0.05 seconds when the initial condition is the fast eigenvector

If you make the initial condition a set of random voltages,

- All eigenvectors will be excited
- The fast ones quickly decay,
- Leaving the slow eigenvector

```
% 10-stage RC Filter

V0 = 0;
V = 10*rand(10,1);
dV = 0*V;
dt = 0.001;
t = 0;
Y = [];

N = [0:10];
plot(N, [V0; V], 'r.-');
hold on
t1 = 0;
while(t < 2)
   dV(1)  = 5*V0 -10.1*V(1) + 5*V(2);
   for i=2:9
       dV(i) = 5*V(i-1) - 10.1*V(i) + 5*V(i+1);
   end
   dV(10) = 5*V(9) - 5.1*V(10);

   V = V + dV * dt;
   t = t + dt;
   t1 = t1 + dt;
   N = [0:10];

   if(t1 >= 0.15)
      plot(N, [V0; V], 'b.-');
      t1 = 0;
   end
   pause(0.01);

   Y = [Y ; [V'] ];
end
```