\[
\dot{x} = f(x), \quad x(0) = x_0
\]

- \(x\) may be vector of states
- \(f\) may be nonlinear function

\[
\dot{x} = Ax, \quad x(0) = x_0 \quad \text{is much simpler for matrix } A.
\]

System of first-order linear differential Eqs.

\[
x(t) = e^{At} x_0 \quad \text{... different class.}
\]

We are interested in numerically solving this, by starting with \(x_0\) and iterating to get \(x_0 \rightarrow x_1 \rightarrow x_2 \rightarrow \ldots \rightarrow x_n\). (trajectory)

**Forward Euler:**

\[
\frac{x_{k+1} - x_k}{\Delta t} \approx \dot{x} = f(x_k) \implies x_{k+1} = x_k + \Delta t f(x_k)
\]

If \(\dot{x} = Ax\) \implies

\[
x_{k+1} = (I + \Delta t A) x_k
\]

(Not very stable)

**Backward (Implicit) Euler:**

\[
\frac{x_{k+1} - x_k}{\Delta t} \approx f(x_{k+1}) \implies x_{k+1} = x_k + \Delta t f(x_{k+1})
\]

If \(\dot{x} = Ax\) \implies

\[
x_{k+1} = x_k + A \Delta t x_{k+1}
\]

\implies

\[
x_{k+1} = (I - A \Delta t)^{-1} x_k \quad \text{better stability.}
\]
Spring-Mass-Damper

\[ m \dddot{x} = -Kx - c \dot{x} \]

\[ m \dddot{x} + kx + c \dot{x} = 0 \]

\[ \dddot{x} + \frac{k}{m} x + \frac{c}{m} \dot{x} = 0 \]

If \( \omega_0 = \sqrt{\frac{k}{m}} \) natural frequency

\[ \zeta = \frac{c}{2\zeta km} \] damping ratio.

\[ \dddot{x} + 2\zeta \omega_0 \dot{x} + \omega_0^2 x = 0 \]

Second order linear differential equation.

\[
\begin{align*}
\dot{x} &= v \\
\dot{v} &= -2\zeta \omega_0 v - \omega_0^2 x
\end{align*}
\]

\[
\begin{bmatrix} 0 & 1 \\
\omega_0^2 & -2\zeta \omega_0
\end{bmatrix}
\]

\( \omega_0 \) and \( \zeta \) determine eigenvalues of \( A \), hence, the behavior of the system.
Cases:

1) Under-damped $\zeta < 1$
   
   System oscillates w/ freq $\omega_d = \omega_0 \sqrt{1 - \zeta^2}$

2) Over-damped $\zeta > 1$

3) Critically Damped $\zeta = 1$

Let's code up Forward Euler

$$x_{k+1} = (I + A \Delta t)x_k$$

... try $\Delta t = .01$, $T = 10$

... compare w/ RK4

... try $\Delta t = 0.1$, $d = 0.5$, $d = 1$, $d = 2$.

What went wrong?..

Look at $\text{eig}(I + A \Delta t)$.

Next time:

- Error analysis
- Global stability properties
- Why is ODE45 so good?
clear all

w = 2*pi;
d = .5; % will break for d=20

A = [0 1; -w^2 -2*d*w];

dt = .01; % time step
T = 10; % amount of time to integrate

x0 = [2; 0]; % initial condition

% iterate forward euler
x(:,1) = x0;
t1(1) = 0;
for i=1:T/dt
    t1(i+1) = i*dt;
    x(:,i+1) = (eye(2) + A*dt)*x(:,i);
end

plot(t1,x(1,:))

% compute better integral using build-in Matlab code
[t,y] = ode45(@(t,y) A*y, 0:dt:T,x0);
hold on
plot(t,y(:,1), 'r')
clear all

w = 2*pi;
d = 1.75; % will break for d=20

A = [0 1; -w^2 -2*d*w];

dt = .1; % time step
T = 10; % amount of time to integrate

x0 = [2; 0]; % initial condition

% iterate forward euler
xF(:,1) = x0;
tF(1) = 0;
for i=1:T/dt
    tF(i+1) = i*dt;
    xF(:,i+1) = (eye(2) + A*dt)*xF(:,i);
end
plot(tF,xF(1,:),'k')
hold on

% iterate backward euler
xB(:,1) = x0;
tB(1) = 0;
for i=1:T/dt
    tB(i+1) = i*dt;
    xB(:,i+1) = inv(eye(2)-A*dt)*xB(:,i);
end
plot(tB,xB(1,:),'b')
hold on

% compute better integral using build-in Matlab code
[t,y] = ode45(@(t,y) A*y, 0:dt:T,x0);
hold on
plot(t,y(:,1),'r')