Elin_Ahlstrand_Exercises_9

Problem 1: N-dimensional linear dynamics
Part 1: Diagonal multi-dimensional dynamics
Suppose you have a set of independent, separate, one-dimensional differential equations $\dot{x}_i = \lambda_i x$. Write the set of λ_i into a diagonal matrix,

$$D = \begin{bmatrix} \lambda_1 & 0 & 0 \\ 0 & \lambda_2 & 0 \\ 0 & 0 & \dots \end{bmatrix}$$

You can then think of your set of independent differential equations as a single vector differential equation, $\dot{x} = Dx$
Thinking in multidimensional terms, what are the conditions under which the multi-dimensional origin, $x = 0$, is a stable point? What happens if some λ_i are less than zero and others are greater than zero?

The conditions under which the multi-dimensional origin, $x = 0$, is a stable point are when:

- All $\mathrm{Re}(\lambda_i) < 0$

If some λ_i are less than zero and others are greater than zero we get:

- A **saddle** F.P. (mixed stability) where there is a mix of $\mathrm{Re}(\lambda_i) < 0 \;\;\&\;\; \mathrm{Re}(\lambda_i) > 0$

Part 2: Non-diagonal multi-dimensional dynamics

a) Find a matrix M that will satisfy the equation below

$$\begin{bmatrix} \dot{x} \\ \dot{y} \end{bmatrix} = M \begin{bmatrix} x \\ y \end{bmatrix}$$ (for the first network)

$$\begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{z} \end{bmatrix} = M \begin{bmatrix} x \\ y \\ z \end{bmatrix}$$ (for the second network)

```
M1
2×2 Array{Float64,2}:
 0.3  -1.2
 1.0  -0.8
```

```
M2
3×3 Array{Float64,2}:
 -0.5  -0.5   0.0
 -0.5   0.1  -1.0
 -1.0   0.0  -1.0
```

b) We can tell that [x, y] = [0, 0] and [x, y, z] = [0, 0, 0] are fixed points in each of the neural network dynamics. Predict if this fixed point is stable or not based on the eigenvalues of M.

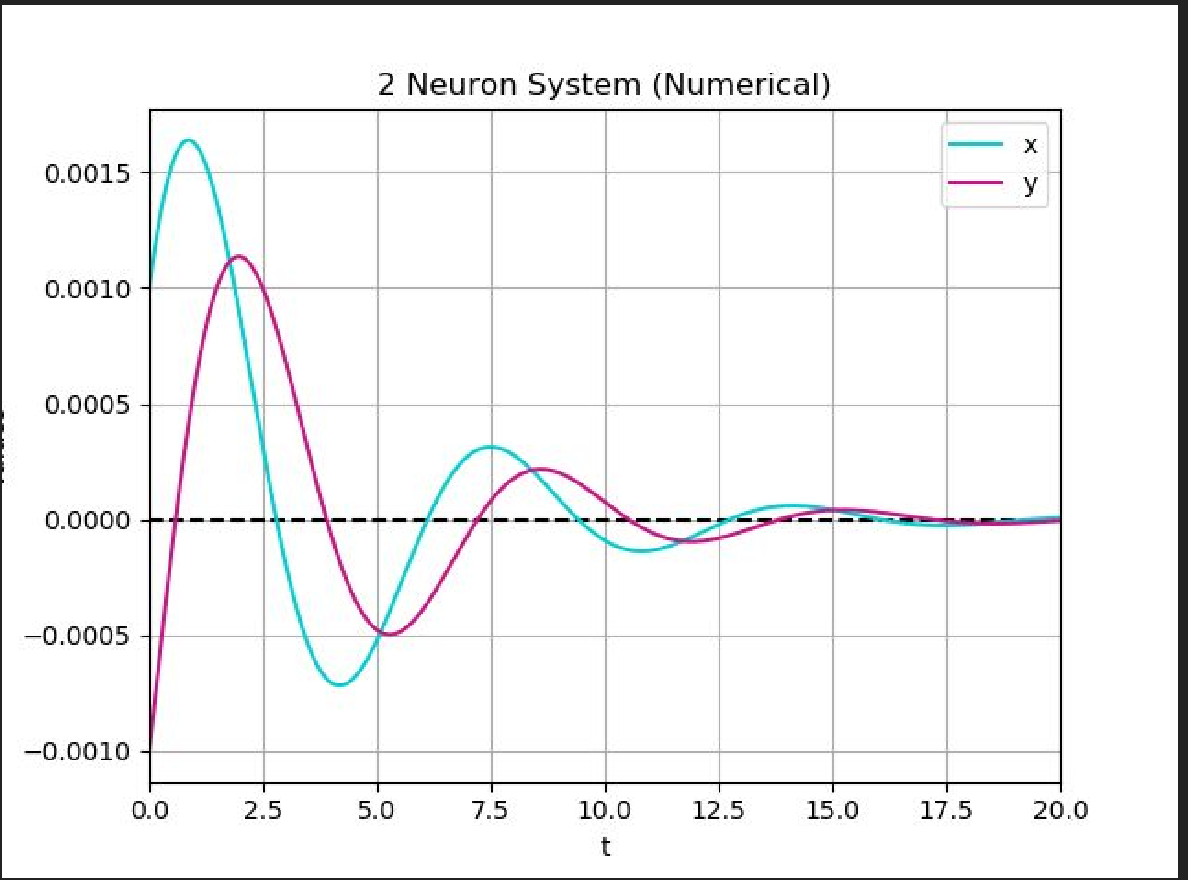
```
eigvals(M1)
2-element Array{Complex{Float64},1}:
 -0.25 + 0.945im
 -0.25 - 0.945im

Since Re(eig) < 0 we predict that the F.P. [x,y] = [0,0] is stable

eigvals(M2)
3-element Array{Complex{Float64},1}:
 -1.428 + 0.0im
 0.014 + 0.374im
 0.014 - 0.374im

Since Re(eig) < & >  0 we predict that the F.P. [x,y, z] = [0,0,0] is
unstable (saddle point)
```

c) Use Euler integration to numerically solve for x and y (and z in the second neural network) for t running from 0 to 20 with dt = 0.001. Set the initial values as $x(t=0)=0.001, y(t=0)=-0.001$ (, and $z(t=0)=0.002$ in the second neural network). Plot the time courses of x and y (and z in the second neural network) against t resulting from the Euler integration.

d) Get the analytical solutions for x and y (and z in the second neural network) for t running from 0 to 20 with dt = 0.001. Use the same initial values as c). Plot the time courses of x and y (and z in the second neural network) against t resulting from the analytical solution. Compare this plot with the plot you made in c). They should look similar if everything is correct.

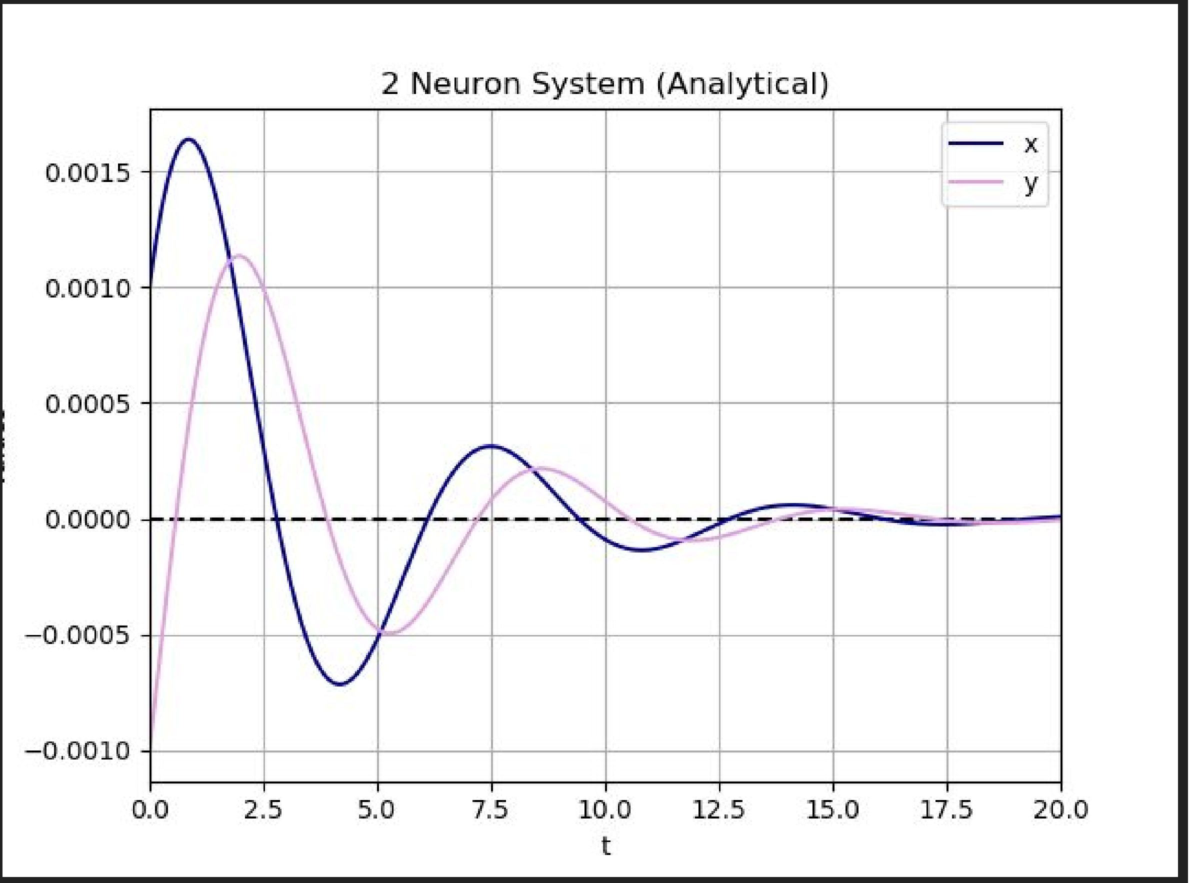
```
##===== 2-Neuron System
# Numerical
M = [0.3 -1.2; 1 -0.8];
      dt = 0.001; t = 0:dt:20;
      x = zeros(2,length(t))
      x[:,1] = [0.001,-0.001]
```

```
for i=1:length(t)-1
	x[:,i+1] = x[:,i] + dt*M*x[:,i]  # Euler recipe
end

figure("EX9_1.2d1"); clf();
	plot(t, x[1,:], color="darkturquoise", label="x")
	plot(t, x[2,:], color="mediumvioletred", label="y")

	hlines(0, 0, 20, linestyle = "--", color="black")
	xlabel("t"); ylabel("values");grid("on");xlim(0,20)
	title("2 Neuron System (Numerical)");legend()
```



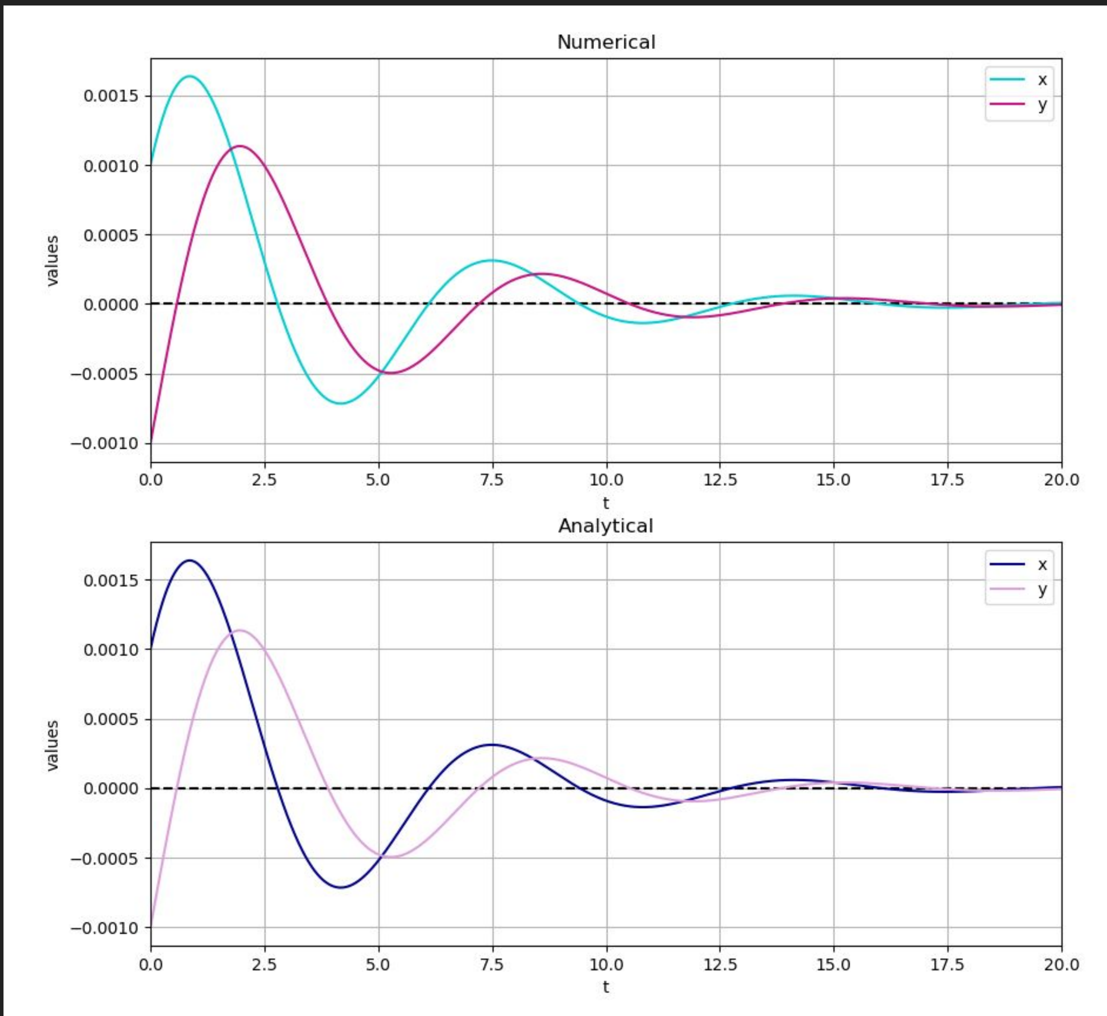

```
# Analytical
eigenspace = eigen(M).vectors
	eigenvalues = eigen(M).values
	initial_xy = [0.001,-0.001];
	uv_o = inv(eigenspace) * initial_xy

	uv_t = zeros(length(initial_xy), length(t))
```

```
    uv_t = complex(uv_t)

    for i = 1:length(initial_xy)
        uv_t[i, :] = uv_o[i] * exp.(t * eigenvalues[i])
    end
    xy_t_analytical =  eigenspace * uv_t;

    figure("EX9_1.2d2"); clf();
        plot(t, xy_t_analytical[1,:], color="darkblue", label = "x")
        plot(t, xy_t_analytical[2,:], color="plum", label = "y")
            hlines(0, 0, 20, linestyle = "--", color="black")
            xlabel("t"); ylabel("values");grid("on");xlim(0,20)
            title("2 Neuron System (Analytical)");legend()
    #
```




```
fig = figure("EX9_1.2d3",figsize=(10,10));
    subplot(211)
    plot(t, x[1,:], color="darkturquoise", label="x")
    plot(t, x[2,:], color="mediumvioletred", label="y")

    hlines(0, 0, 20, linestyle = "--", color="black")
    xlabel("t"); ylabel("values");grid("on");xlim(0,20)
    title("Numerical");legend()
```

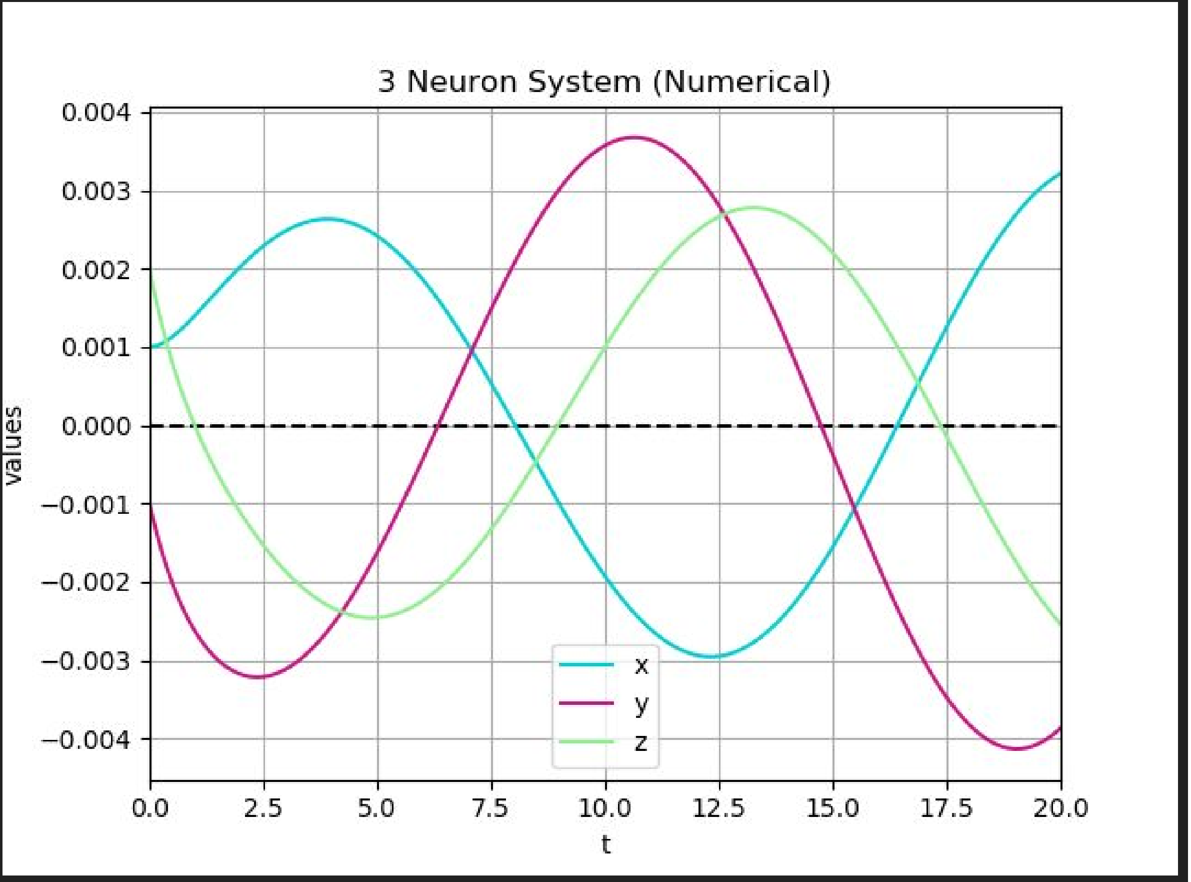
```
    subplot(212)
    plot(t, xy_t_analytical[1,:], color="darkblue", label = "x")
    plot(t, xy_t_analytical[2,:], color="plum", label = "y")
        hlines(0, 0, 20, linestyle = "--", color="black")
        xlabel("t"); ylabel("values");grid("on");xlim(0,20)
        title("Analytical");legend()
##=====
```



These two solutions look similar!

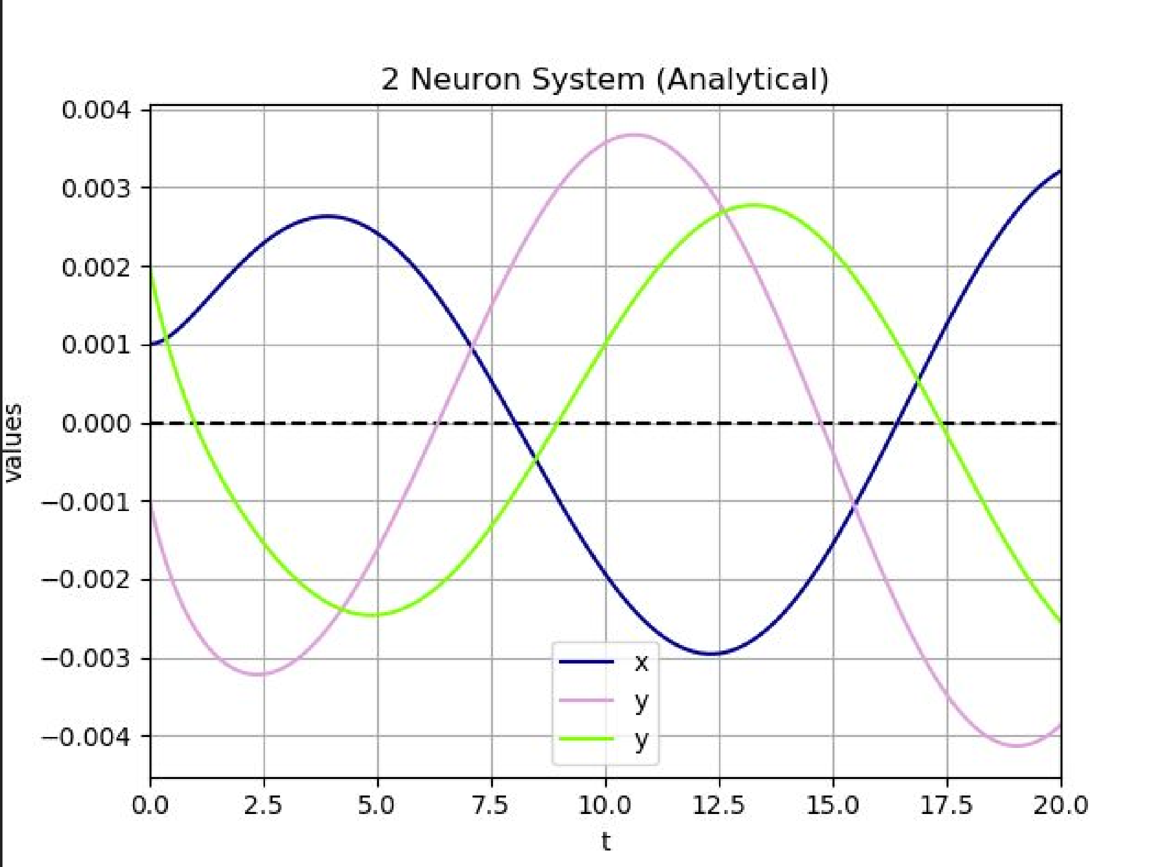
```
##===== 3-Neuron System
M = [-0.5 -0.5 0; -0.5 0.1 -1; -1 0 -1]

dt = 0.001; t = 0:dt:20
    x = zeros(3,length(t))
    x[:,1,1] = [0.001,-0.001,0.002]

    for i=1:length(t)-1
```

```
    x[:,i+1] = x[:,i] + dt*M*x[:,i]  # Euler recipe
    end

    figure("EX9_1.2d4"); clf();
    plot(t, x[1,:], color="darkturquoise", label="x")
    plot(t, x[2,:], color="mediumvioletred", label="y")
    plot(t, x[3,:], color="lightgreen", label="z")

    hlines(0, 0, 20, linestyle = "--", color="black")
    xlabel("t"); ylabel("values");grid("on");xlim(0,20)
    title("3 Neuron System (Numerical)");legend()
    #
```



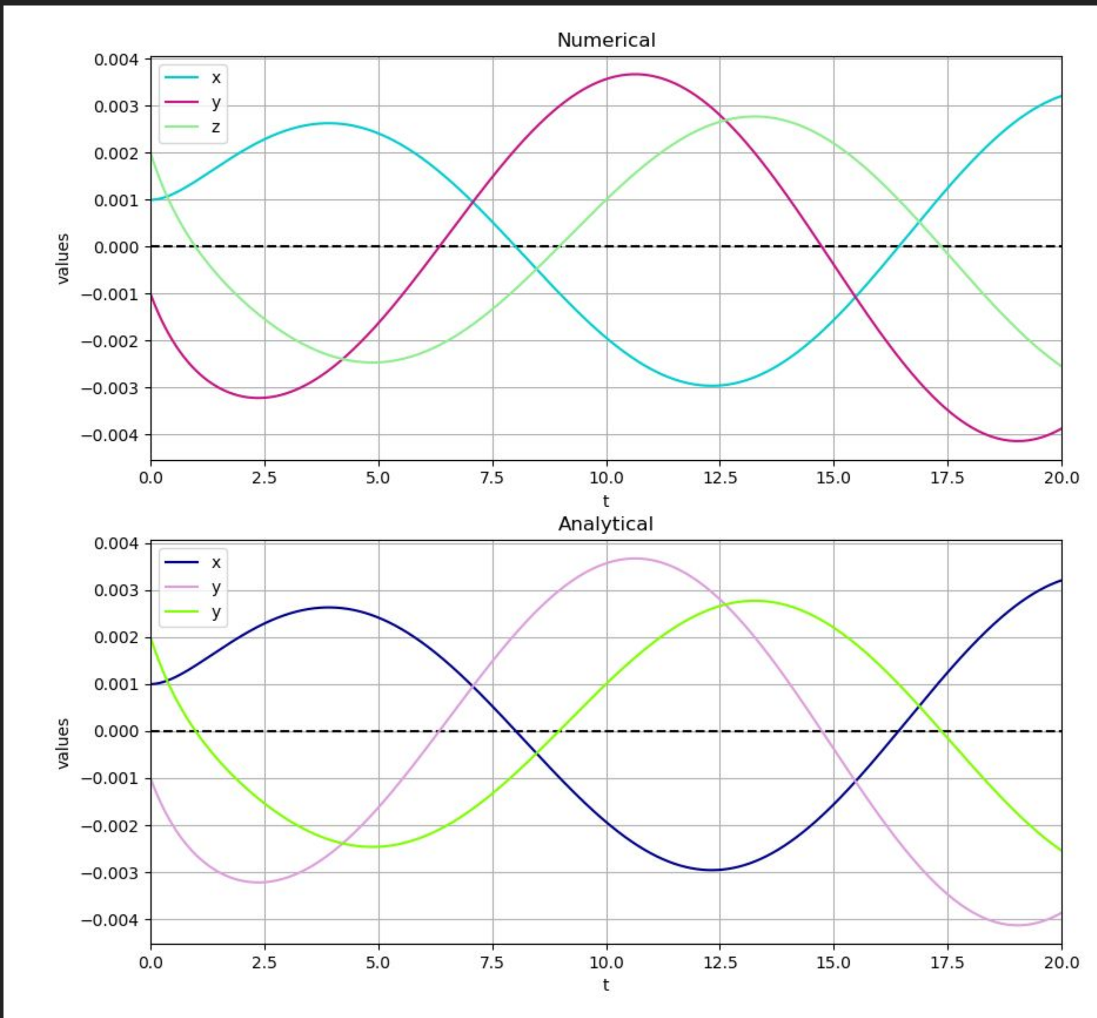

```
## Analytical
eigenspace = eigen(M).vectors
eigenvalues = eigen(M).values
initial_xy = [0.001,-0.001,0.002];
    uv_o = inv(eigenspace) * initial_xy

    uv_t = zeros(length(initial_xy), length(t))
    uv_t = complex(uv_t)

    for i = 1:length(initial_xy)
```

```
            uv_t[i, :] = uv_o[i] * exp.(t * eigenvalues[i])
      end
      xy_t_analytical =  eigenspace * uv_t;

      figure("EX9_1.2d5"); clf();
            plot(t, xy_t_analytical[1,:], color="darkblue", label = "x")
            plot(t, xy_t_analytical[2,:], color="plum", label = "y")
            plot(t, xy_t_analytical[3,:], color="chartreuse", label =
"y")
      hlines(0, 0, 20, linestyle = "--", color="black")
      xlabel("t"); ylabel("values");grid("on");xlim(0,20)
      title("2 Neuron System (Analytical)");legend()
      #
```




```
fig = figure("EX9_1.2d6",figsize=(10,10));
      subplot(211)
      plot(t, x[1,:], color="darkturquoise", label="x")
      plot(t, x[2,:], color="mediumvioletred", label="y")
      plot(t, x[3,:], color="lightgreen", label="z")
            hlines(0, 0, 20, linestyle = "--", color="black")
            xlabel("t"); ylabel("values");grid("on");xlim(0,20)
            title("Numerical");legend()
```

```
	subplot(212)
	plot(t, xy_t_analytical[1,:], color="darkblue", label = "x")
	plot(t, xy_t_analytical[2,:], color="plum", label = "y")
	plot(t, xy_t_analytical[3,:], color="chartreuse", label = "y")
		hlines(0, 0, 20, linestyle = "--", color="black")
		xlabel("t"); ylabel("values");grid("on");xlim(0,20)
		title("Analytical");legend()
#
```




```
These solutions look similar!
```

e) For the first neural network, plot the trajectory on a x-y plane for t = 0 to 20.

```
figure("EX9_1.2e",figsize=(8.5,7)); clf();
	xlim([-0.001, 0.002]); ylim([-0.0011, 0.0012]);
	hlines(0, -0.001, 0.002, color="black")
	vlines(0, -0.0011, 0.0012, color="black")
	grid("on")
```

```
plot(x[1,:], x[2,:], color="darkturquoise", label= "[x(t),y(t)]")
# plot(x[1,:], x[2,:], "d")
plot(x[1,1], x[2,1], "x", color="slategrey", label="start")
plot(x[1,end], x[2,end], "o", color="red", label="stop")
xlabel("x"); ylabel("y"); title("[x,y] time course")
legend()
```