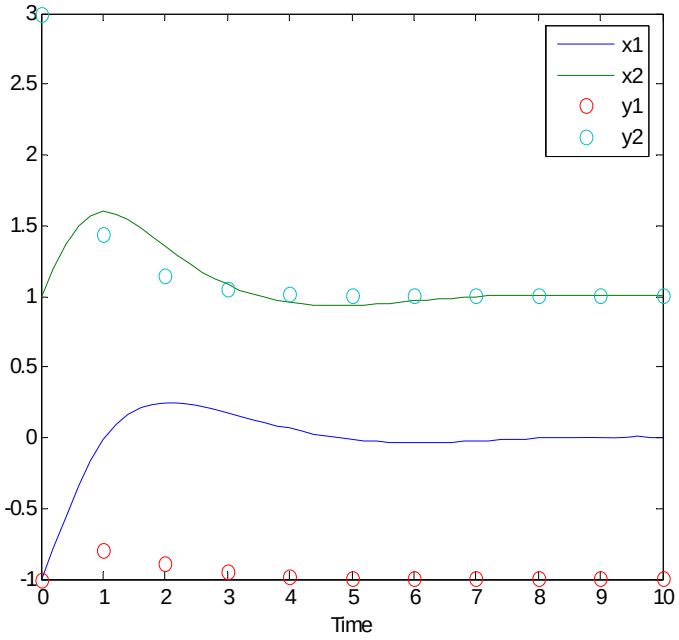
1b2.

```
>> %Get the time from Part a
>> time = [tout(1)];
>> for i = 1:10
time = [time; tout(i*5 + 1)];
end
>> time = time'
time =
0  1.0000  2.0000  3.0000  4.0000  5.0000  6.0000  7.0000  8.0000  9.0000
10.0000

>> %Get the output from Part a
>> x = [xout(1,:)];
>> for i = 1:10
x = [x; xout(i*5 + 1,1) xout(i*5 + 1,2)];
end
>> x = x'
x =
-1.0000  -0.0144  0.2457  0.1804  0.0652  -0.0133  -0.0361  -0.0236  -0.0050  0.0047
0.0055
1.0000   1.5969   1.3539  1.0824  0.9581  0.9379   0.9653   0.9961   1.0100   1.0093
1.0036

>> I = eye(2);
>> E = [0 1; -1 0];
>> x0 = [-1;1];
>> %Assumptions: One layer, transfer function = purelin (derivate = 1),
&
>> %learning rate = 0.1

>> %Backpropagation algorithm to reduce mean square error from textbook
>> %Let w1[0] = 10 and w2[0] = 10
>> w = [10;10];
>> weights = [];
>> out = [];
>> for t = 0:1:10
y = (2*I - ( inv( I + (exp(w(1)*t)*E) ) * inv( I + (exp(w(2)*t)*E) ) ))
* x0;
out = [out y];
error = x(:,1) - y;
s = -2 * [1 0; 0 1] * error;
w = w - .1*s*(t)';
weights = [weights w];
end

>> w
w =
   21.0000
   -1.0000
```

1b3.

```
>> out = [];
>> for t = 0:1:10
y = (2*I - ( ( I + inv(exp(w(1)*t)*E) ) * inv( I + (exp(w(2)*t)*E) ) ))
* x0;
out = [out y];
end
```

```
>> plot(tout,xout,time,out,'o')
>> xlabel('Time')
```




```
>> sum = 0;
>> for t = 1:11
sum = sum + (x(t)-out(t))^2;
end
>> mean_square_error = sum/22
mean_square_error =
    0.4227

>> %y(t) is represented by 'o' in the diagram. They do not match up
with x(t) but demonstrate a similar curvature. As t->∝, y(t) goes
towards the initial conditions over time. With the weight of y1 = 21
and the weight of y2 = –1, the mean square error is 0.4227.
```