**Source Code:**

function [t,y] = CubicSplineVector(t0,y0)

%this function will call the private function below in order to obtain the

%coefficients of the cubic spline, then generates a vector of points to

%represent the spline itself. t0 (t-naught) and y0 (y-naught) are the knots

%defining the spline

[z,h]=CubicSplineCoefficients(t0,y0);

t=[];

y=[];

for i=1:length(t0)-1

x=linspace(t0(i),t0(i+1));%tmesh on interval between knots

S=(z(i)/(6\*h(i)))\*(t0(i+1)-x).^3+(z(i+1)/(6\*h(i)))\*(x-t0(i)).^3;

S=S+(y0(i+1)/h(i)-z(i+1)\*h(i)/6)\*(x-t0(i))+(y0(i)/h(i)-z(i)\*h(i)/6)\*(t0(i+1)-x);

%spline values on tmesh

t=[t,x];

y=[y,S];

end

end

function [z,h]=CubicSplineCoefficients(t,y)

%This function simply computes and outputs the coefficients z and h for

%the cubic spline, using the pseudocode found in the book.

h=t(2:end)-t(1:end-1);

%computes the h's using a vectorized version of the for loop found in the book's pseudocode

b=6\*(y(2:end)-y(1:end-1));

b=b./h;

u(1)=0;

v(1)=0;%these two assignments simply make certain that the indices on the u and v vectors line up with the h and b vector, as there is no u\_0 or v\_0 in the book's notation;

u(2)=2\*(h(1)+h(2));

v(2)=b(2)-b(1);

for i=3:length(t)-1

u(i)=2\*(h(i)+h(i-1))-(h(i-1))^2/u(i-1);

v(i)=b(i)-b(i-1)-h(i-1)\*v(i-1)/u(i-1);

end

z=zeros(size(t));

for i=length(t)-1:-1:2

z(i)=v(i)-h(i)\*z(i+1)/u(i);

end

end

**Function Test:**

t0=linspace(-1,1,12);

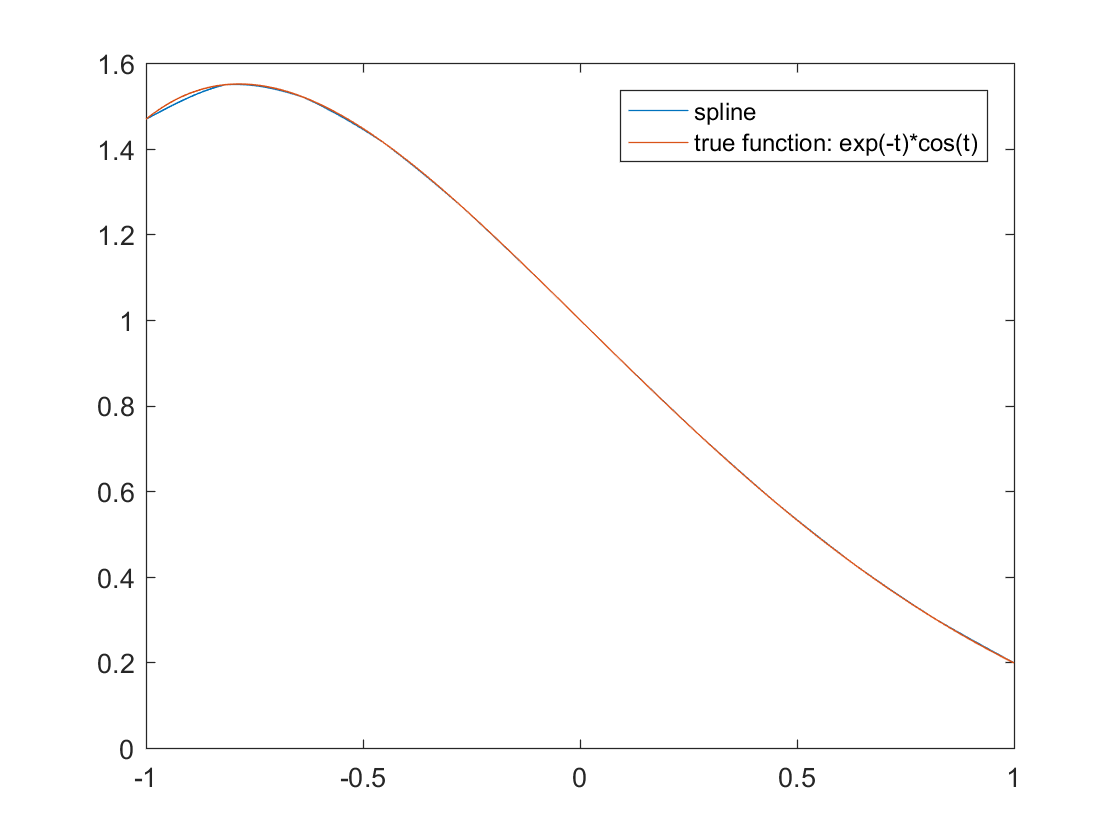
y0=exp(-t0).\*cos(t0);

>> [t,y]=CubicSplineVector(t0,y0);

>> plot(t,y,t,exp(-t).\*cos(t))

>> legend('spline','true function: exp(-t)\*cos(t)')

Output:



**Addressing the problem:**

t=[0:24];

>> x=[0,0,2,2,1,0,-1,-2,-2,-1,0,1,1,-1,-2,-3,-4,-4,-2,0,2,4,4,1,0];

>> y=[0,-2,-1,1,2,2,1,0,-2,-1,1,2.5,4,5,4,3,1,-1,-3,-4,-2,0,1,1,0];

>> [tx,sx]=CubicSplineVector(t,x);

>> [ty,sy]=CubicSplineVector(t,y);

plot(sx,sy)

(Plot on its own page)

**Conclusion:**

The plot reveals that there is some overfitting, but it is a surprisingly good recreation of what was a terrible doodle. Cubic splines do a decent job