MATH 320 Homework 1

Jack Norleans

9/8/2016

1. Freshwater Density Problem

```
function freshwater_density = freshdens()
    %X produces a vector of temperatures from 32 to 93.2 degrees with an
    %interval of 3.4 degrees between points.
    %XC converts termperatures in X to celsius via the formula Celsius =
    %(5/9)*(Farenheit - 32).
    %P generates freshwater densities at each temperature in Celsius, and
    %plot generates a trace of density as function of temperature.
    X = linspace(32, 93.2, 18);
    XC = (5/9)*(X-32);
    P = 5.5289*10^-8*(XC.^3)-8.5016*10^-6*(XC.^2)+6.5622*10^-5*(XC)+0.99987
    plot(XC,P)
    xlabel('Temperature (degrees Celsius)')
    ylabel('Freshwater Density (kg/L)')
end
```

Output:

P =

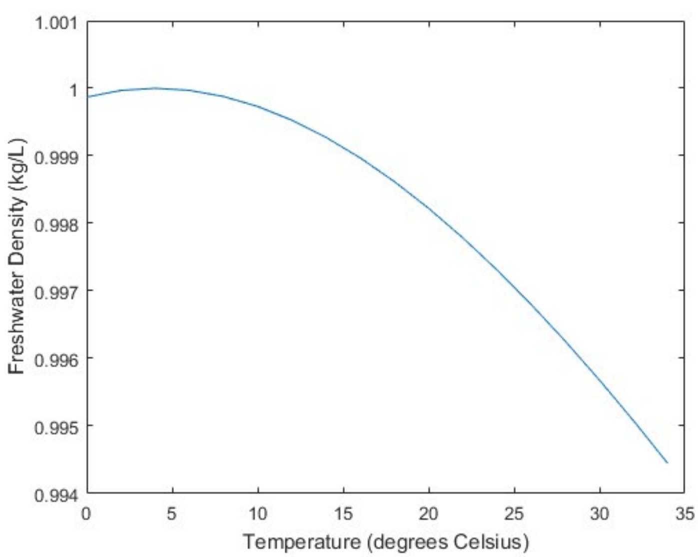
Columns 1 through 9

0.9999	1.0000	1.0000	1.0000	0.9999	0.9997	0.9995	0.9993	0.9990

Columns 10 through 18

0.9986	0.9982	0.9978	0.9973	0.9968	0.9963	0.9957	0.9951	0.9944

Figures:

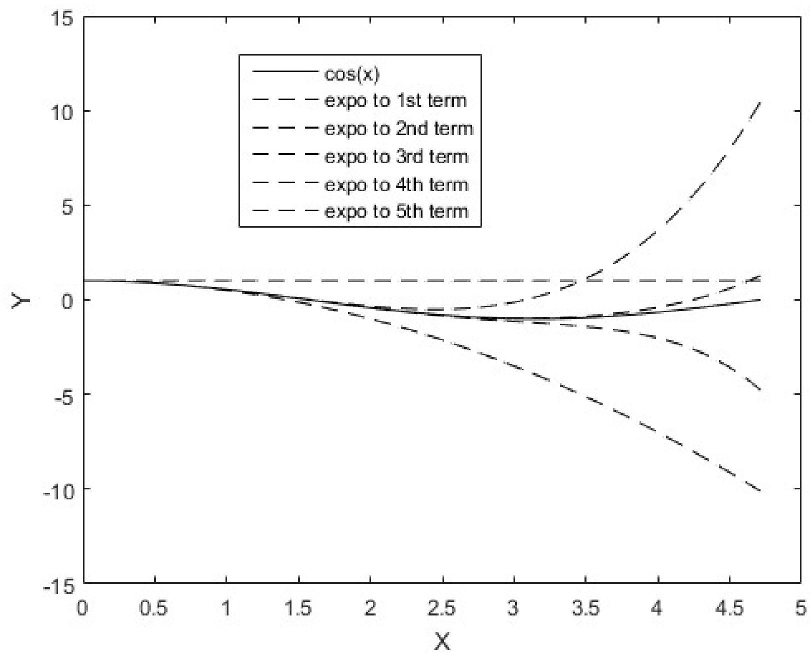


2. MacLaurin Series of Cosine

```
function maclaurin_series = macser()
    %X generates a regularly spaced series of points between 0 and 3*pi/2
    %for evaluating Maclaurin series
    %Y provides a reference trace of what cos(X) should look like
    X = linspace(0,1.5*pi,361);
    Y = cos(X);
    plot(X,Y,'k')
    hold on
    %A is a cell of arrays, with each array containing a term of the
    %Maclaurin series for cos(x) applied to each position in array X.
    A = cell(1,5);
    for k = 0:4
        A{k+1} = zeros(1,length(X));
        A{k+1} = (-1)^-k*X.^(k*2)/factorial(k*2);
    end

    %Y1 is an array that displays the results of summing varying number of
    %terms from the Maclaurin series for cos(x). In each iteration of the
    %for loop, Y1 is calculated, added to the plot, and reset.  Each repeat
    %of the loop sees an additional series term added to the sum and
    %plotted.
    for k = 1:5
        Y1 = zeros (1,length(X));
        j = 1;
        while j <= k
            Y1 = Y1 + A{j};
            j = j+1;
        end
        plot(X,Y1,'--k')
    end
    legend('cos(x)','expo to 1st term','expo to 2nd term','expo to 3rd
term','expo to 4th term','expo to 5th term')
    hold off
    xlabel('X')
    ylabel('Y')
end
```

Figures:



3. Cartesian to Polar Problem

```
function polar_cartesian = polcar()
    X = [2,2,0,-3,-2,-1,0,0,2];
    Y = [0,1,3,1,0,-2,0,-2,2];
    %R produces an array of radii for each 2-D cartesian coordinate.
    %Theta yields the angle corresponding to radius for each 2-D cartesian
    %coordinate.
    R = (X.^2 + Y.^2).^0.5
    Theta = zeros(1,9);
    for i = 1:9
       if X(1,i) == 0 && Y(1,i) == 0
           Theta(1,i) = 0;
       else if X(1,i) <= 0 && Y(1,i) == 0
           Theta(1,i) = 180;
       else if X(1,i) == 0 && Y(1,i) >= 0
           Theta(1,i) = 90;
       else if X(1,i) == 0 && Y(1,i) <= 0
           Theta(1,i) = 270;
       else if X(1,i) <= 0 && Y(1,i) >= 0
           Theta(1,i) = atand(Y(1,i)/X(1,i)) + 180;
       else if X(1,i) <= 0 && Y(1,i) <= 0
           Theta(1,i) = atand(Y(1,i)/X(1,i)) - 180;
           else
               Theta(1,i) = atand(Y(1,i)/X(1,i));
           end
           end
           end
           end
           end
       end
```

```
    end
    Theta
end
```

Output:

R =

2.0000 2.2361 3.0000 3.1623 2.0000 2.2361 0 2.0000 2.8284

Theta =

0 26.5651 90.0000 161.5651 180.0000 -116.5651 0 270.0000 45.0000

4. Cartesian Vectors

```
function crossproduct = crospro(x1,y1,z1,x2,y2,z2)
    %inputs x1, y1, z1 correspond to xyz coordinates of the first vector.
    %x2, y2, z2 represent the coords of the second vector.
    a = [x1,y1,z1];
    b = [x2,y2,z2];
%     a = [6,4,2;3,2,-6;2,-2,1;-1,0,0];
%     b = [2,6,4;4,-3,1;4,2,-4;0,-1,0];
    z = zeros(1,3);
    %z provides a reference point at the origin from which vectors will be
    %plotted.
    %dotpro calculates the dot product for 3D vectors a and b.
    %magpro yields the products of magnitudes of a and b.
    %theta is an array of angles between vectors in a and b.
    %in above three lines, vectors in a and b are matched by the order they
    %appear in their respective arrays.
    %c is an array of 3D vectors that are the cross products of a and b.
    %magc yields the magnitude of the cross product vector as determined by
    %product of magnitudes of a and b and the sine of their subtended
    %angle.
    dotpro = sum(a.*b,2);
    magpro = (sum(a.^2,2).*sum(b.^2,2)).^0.5;
    theta = acosd(dotpro./magpro)
    c = cross(a,b,2)
    magc = magpro.*sind(theta)
        quiver3(z(1,1),z(1,2),z(1,3),a(1,1),a(1,2),a(1,3));
        hold on
        quiver3(z(1,1),z(1,2),z(1,3),b(1,1),b(1,2),b(1,3));
        quiver3(z(1,1),z(1,2),z(1,3),c(1,1),c(1,2),c(1,3));
        hold off
        xlabel('X axis')
        ylabel('Y axis')
        zlabel('Z axis')
end
```

Output

a.

theta =

38.2132

c =

4 -20 28

magc =

34.6410

b.

theta =

90

c =

-16 -27 -17

magc =

35.6931

c.

theta =

90

c =

6 12 12

magc =

18

d.

theta =

90

c =

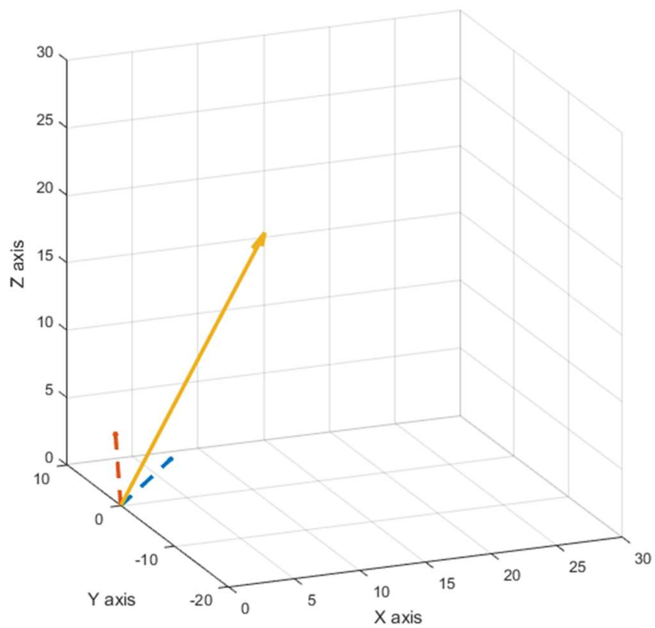
0 0 -1

magc =

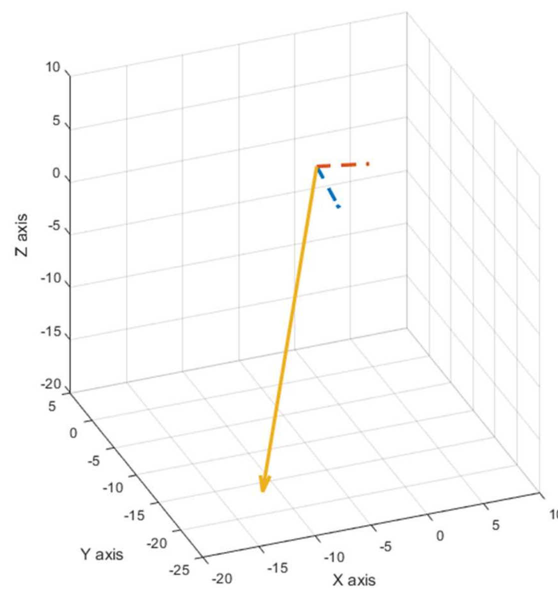
1

Figures:

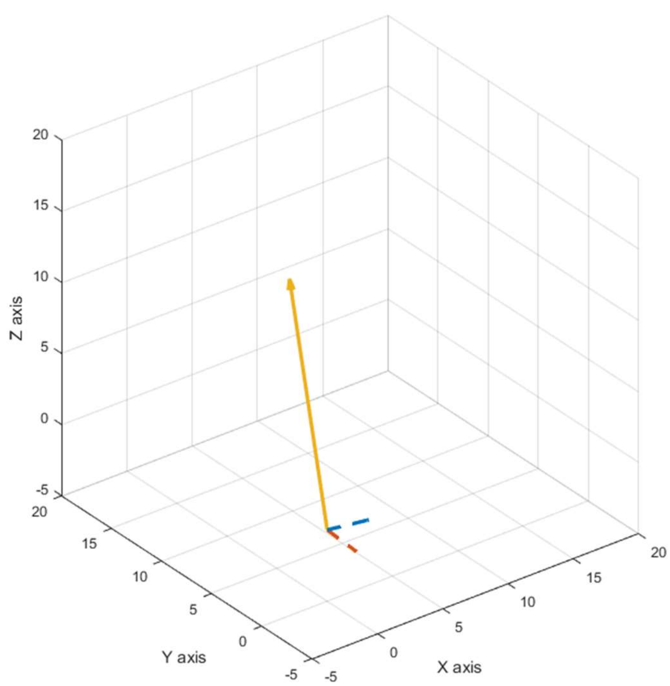
a.



b.



c.



d.