Math 320 HW1 - Ammar Elsheshtawy

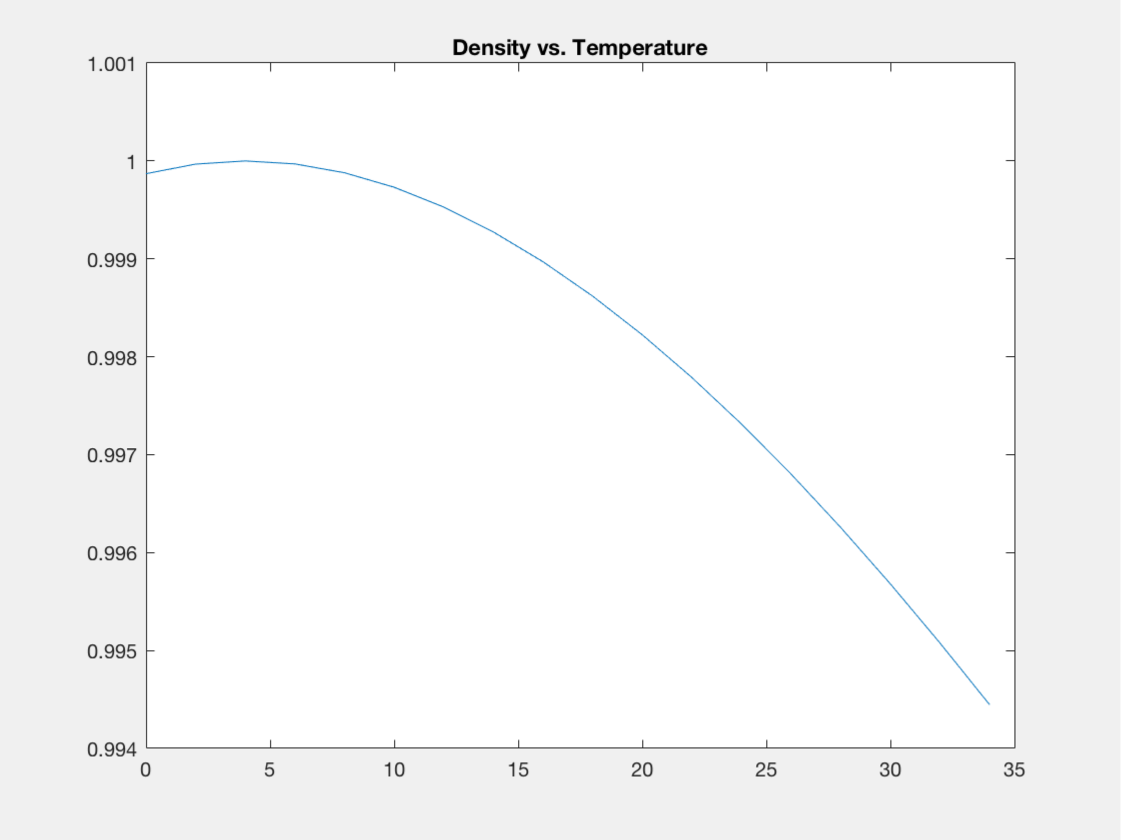
Table of Contents

Question 1

This code creates a vector of temperatures in Fahrenheit, converts to Celsius, uses the Celsius temperatures as inputs to the cubic equation to compute density, and then plots density against the Celsius temperatures.

```
temp_f = 32:3.6:93.2
temp_c = (5/9)*(temp_f-32)

tc = temp_c; tc2 = temp_c.^2; tc3 = temp_c.^3

% Cubic equation
dens = 5.5289*(10^(-8))*tc3 - 8.5016*(10^(-6))*tc2 +
 6.5622*(10^(-5))*tc + 0.99987

plot(tc, dens)
title('Density vs. Temperature')
```

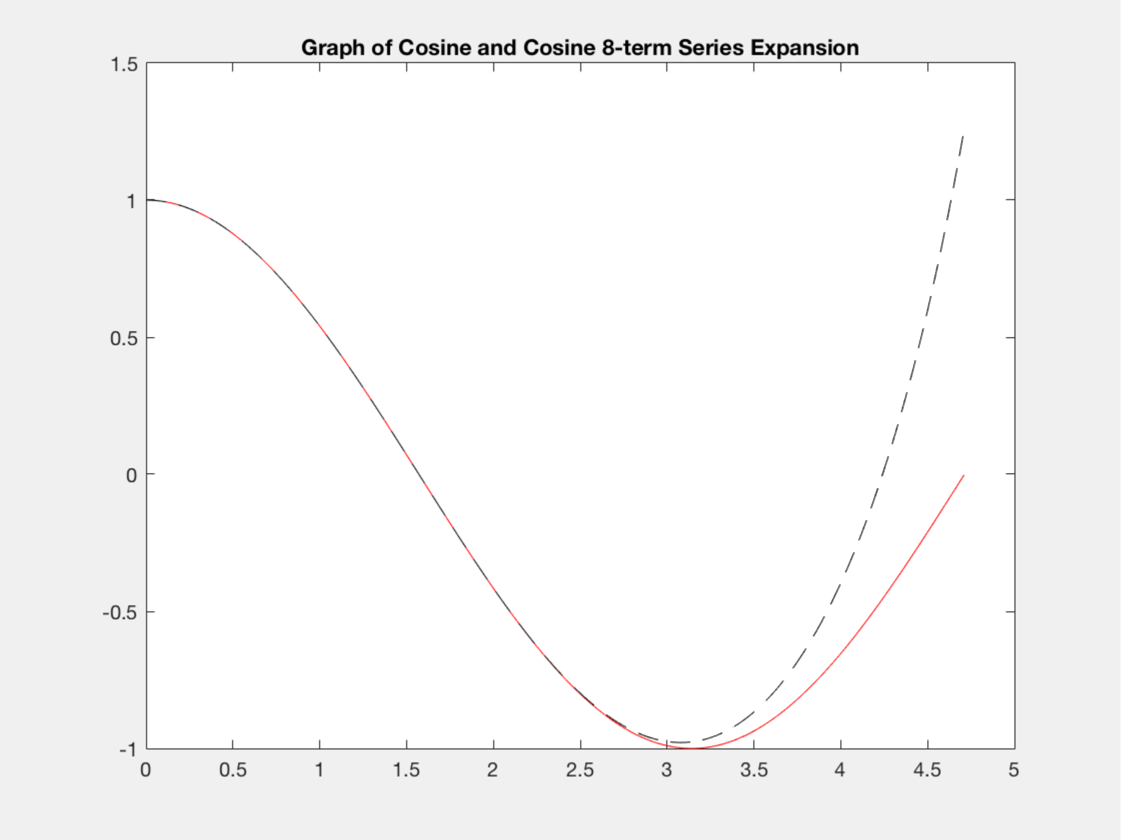
```
temp_f =

  Columns 1 through 7

   32.0000   35.6000   39.2000   42.8000   46.4000   50.0000   53.6000

  Columns 8 through 14

   57.2000   60.8000   64.4000   68.0000   71.6000   75.2000   78.8000

  Columns 15 through 18

   82.4000   86.0000   89.6000   93.2000


temp_c =

  Columns 1 through 7

         0    2.0000    4.0000    6.0000    8.0000   10.0000   12.0000

  Columns 8 through 14

   14.0000   16.0000   18.0000   20.0000   22.0000   24.0000   26.0000
  Columns 15 through 18

   28.0000   30.0000   32.0000   34.0000


tc3 =

   1.0e+04 *

  Columns 1 through 7

         0    0.0008    0.0064    0.0216    0.0512    0.1000    0.1728

  Columns 8 through 14

    0.2744    0.4096    0.5832    0.8000    1.0648    1.3824    1.7576

  Columns 15 through 18

    2.1952    2.7000    3.2768    3.9304


dens =

  Columns 1 through 7

    0.9999    1.0000    1.0000    1.0000    0.9999    0.9997    0.9995

  Columns 8 through 14

    0.9993    0.9990    0.9986    0.9982    0.9978    0.9973    0.9968

  Columns 15 through 18

    0.9963    0.9957    0.9951    0.9944
```



Question 2

This code creates a vector of x values, computes the values of the 4 term Cosine series expansion at these values, and then plots these expansion values on the same plot as values from the built-in Matlab Cosine function.

```
x = 0:0.01:(3*pi)*(0.5);

expans = 1 - (x.^2)/(factorial(2)) + (x.^4)/(factorial(4)) - (x.^6)/
(factorial(6)) + (x.^8)/(factorial(8));

plot(x, cos(x), 'r', x, expans, '--k')
title('Graph of Cosine and Cosine 8-term Series Expansion')
```



Question 3

This code first creates the test array and an output array to store outputs. Then, a for loop iterates through each row of the test array - each r is computed right away using the cartesian values in the relevant row, while the manner in which theta is assigned depends on the values of the inputs. This varied theta assignment is accomplished through a series of if and else statements. Once the loop is complete, theta is converted to degrees and the array of polar coordinates is displayed.

```
test = [2, 2, 0, -3, -2, -1, 0, 0, 2; 0, 1, 3, 1, 0, -2, 0, -2, 2]'
out  = zeros(size(test,1),size(test,2))


for i=1:size(test,1)
    x = test(i,1)
    y = test(i,2)

    out(i,1) = sqrt(x^2 + y^2)

if (x < 0)
    if (y > 0) out(i,2) = atan(y/x) + pi
    elseif (y < 0) out(i,2) = atan(y/x) - pi
    else out(i,2) = pi
    end
elseif (x > 0) out(i,2) = atan(y/x)
elseif (y > 0) out(i,2) = pi/2
elseif (y < 0) out(i,2) = -pi/2
```

```
else out(i,2) = 0
end
end

out(:,2) = rad2deg(out(:,2))

display(out)
```

```
test =

     2     0
     2     1
     0     3
    -3     1
    -2     0
    -1    -2
     0     0
     0    -2
     2     2


out =

     0     0
     0     0
     0     0
     0     0
     0     0
     0     0
     0     0
     0     0
     0     0


x =

     2


y =

     0


out =

     2     0
     0     0
     0     0
     0     0
     0     0
     0     0
     0     0
     0     0
     0     0


out =

     2     0
     0     0
     0     0
     0     0
     0     0
     0     0
     0     0
     0     0
     0     0


x =

     2


y =

     1


out =

    2.0000         0
    2.2361         0
         0         0
         0         0
         0         0
         0         0
         0         0
         0         0
         0         0


out =

    2.0000         0
    2.2361    0.4636
         0         0
         0         0
         0         0
         0         0
         0         0
         0         0
         0         0


x =
     0


y =

     3


out =

    2.0000         0
    2.2361    0.4636
    3.0000         0
         0         0
         0         0
         0         0
         0         0
         0         0
         0         0


out =

    2.0000         0
    2.2361    0.4636
    3.0000    1.5708
         0         0
         0         0
         0         0
         0         0
         0         0
         0         0


x =

    -3


y =

     1


out =

    2.0000         0
    2.2361    0.4636
    3.0000    1.5708
    3.1623         0
         0         0
         0         0
         0         0
         0         0
         0         0


out =

    2.0000         0
    2.2361    0.4636
    3.0000    1.5708
    3.1623    2.8198
         0         0
         0         0
         0         0
         0         0
         0         0


x =

    -2


y =

     0


out =

    2.0000         0
    2.2361    0.4636
    3.0000    1.5708
    3.1623    2.8198
    2.0000         0
         0         0
         0         0
         0         0
         0         0


out =

    2.0000         0
    2.2361    0.4636
    3.0000    1.5708
    3.1623    2.8198
    2.0000    3.1416
         0         0
         0         0
         0         0
         0         0


x =
    -1


y =

    -2


out =

    2.0000         0
    2.2361    0.4636
    3.0000    1.5708
    3.1623    2.8198
    2.0000    3.1416
    2.2361         0
         0         0
         0         0
         0         0


out =

    2.0000         0
    2.2361    0.4636
    3.0000    1.5708
    3.1623    2.8198
    2.0000    3.1416
    2.2361   -2.0344
         0         0
         0         0
         0         0


x =

     0


y =

     0


out =

    2.0000         0
    2.2361    0.4636
    3.0000    1.5708
    3.1623    2.8198
    2.0000    3.1416
    2.2361   -2.0344
         0         0
         0         0
         0         0


out =

    2.0000         0
    2.2361    0.4636
    3.0000    1.5708
    3.1623    2.8198
    2.0000    3.1416
    2.2361   -2.0344
         0         0
         0         0
         0         0


x =

     0


y =

    -2


out =

    2.0000         0
    2.2361    0.4636
    3.0000    1.5708
    3.1623    2.8198
    2.0000    3.1416
    2.2361   -2.0344
         0         0
    2.0000         0
         0         0


out =

    2.0000         0
    2.2361    0.4636
    3.0000    1.5708
    3.1623    2.8198
    2.0000    3.1416
    2.2361   -2.0344
         0         0
    2.0000   -1.5708
         0         0


x =
     2


y =

     2


out =

    2.0000         0
    2.2361    0.4636
    3.0000    1.5708
    3.1623    2.8198
    2.0000    3.1416
    2.2361   -2.0344
         0         0
    2.0000   -1.5708
    2.8284         0


out =

    2.0000         0
    2.2361    0.4636
    3.0000    1.5708
    3.1623    2.8198
    2.0000    3.1416
    2.2361   -2.0344
         0         0
    2.0000   -1.5708
    2.8284    0.7854


out =

    2.0000         0
    2.2361   26.5651
    3.0000   90.0000
    3.1623  161.5651
    2.0000  180.0000
    2.2361 -116.5651
         0         0
    2.0000  -90.0000
    2.8284   45.0000


out =

    2.0000         0
    2.2361   26.5651
    3.0000   90.0000
    3.1623  161.5651
    2.0000  180.0000
    2.2361 -116.5651
         0         0
    2.0000  -90.0000
    2.8284   45.0000
```

Question 4

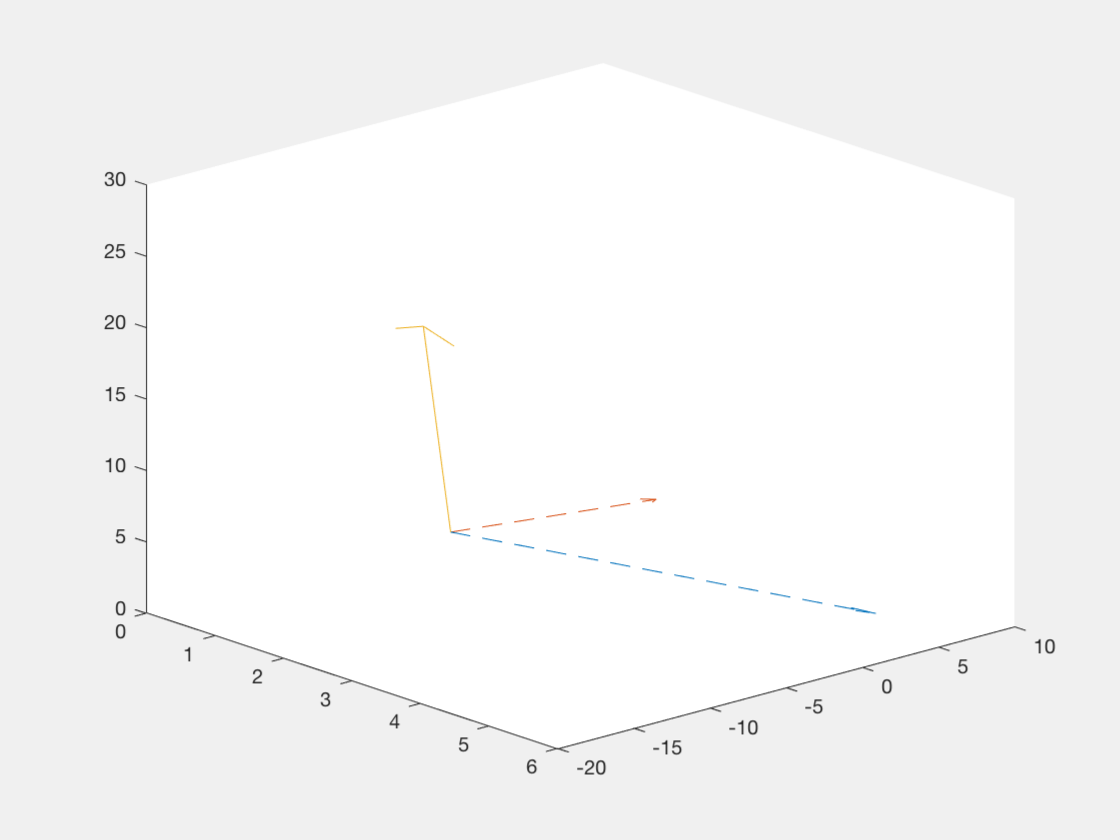
This function first calculates the angle between the tail of two input vectors with the help of some built in matlab functions for magnitude (norm), dot product (dot), and arccos (acos). The normal vector is then calculated using (cross) and its corresponding magnitude using (norm). The three vectors are then plotted in 3D, with each line being drawn on the same plot separately. The dashed lines are the input vectors, while the solid is their cross product. A viewing angle is specified so multiple outputs from the function can be compared more easily.

The code below the function evaluates it for 4 test cases.

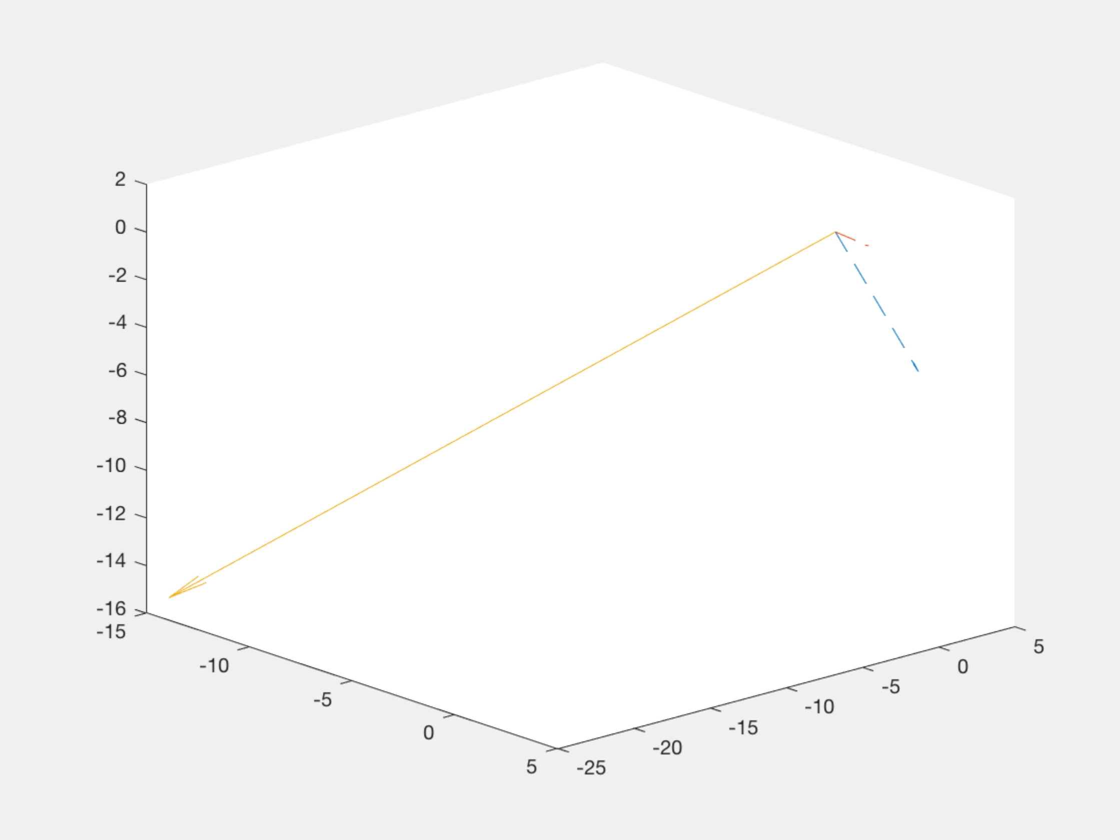
```
function [theta, c, c_mag] = vectors(a,b)
% [theta, c, c_mag] = vectors(a,b) gives the angle between two vectors
% their cross product, the magnitude of their cross product, and
 produces
% a 3D plot of the two vectors and their cross product at the origin
% input:
%   a, b = two 3 element vectors
% output:
%   theta = angle between a and b
%   c = cross product of a and b
%   c_mag = magnitude of c

theta = acos(dot(a,b) / (norm(a) * norm(b)));
c = cross(a,b);
c_mag = norm(c);

clf
quiver3(0,0,0,a(1),a(2),a(3), 'LineStyle','--');
hold on
quiver3(0,0,0,b(1),b(2),b(3), 'LineStyle','--');
hold on
quiver3(0,0,0,c(1),c(2),c(3), 'LineStyle','-');
view(48, 23)
grid
```

```
[theta, c, c_mag] = vectors([6 4 2],[2 6 4])
```

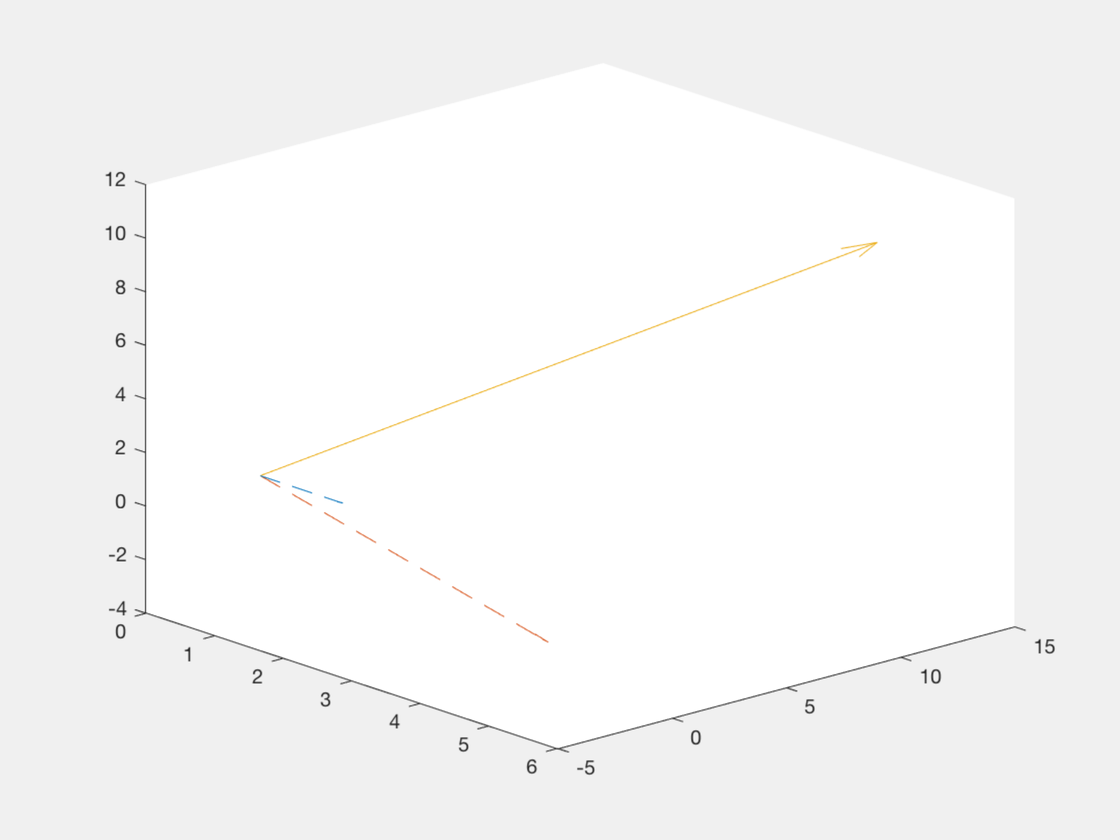
```
theta =
    0.6669


c =

     4   -20    28


c_mag =

   34.6410
```



```
[theta, c, c_mag] = vectors([3 2 -6],[4 -3 1])

theta =

    1.5708


c =

   -16   -27   -17


c_mag =
    35.6931
```

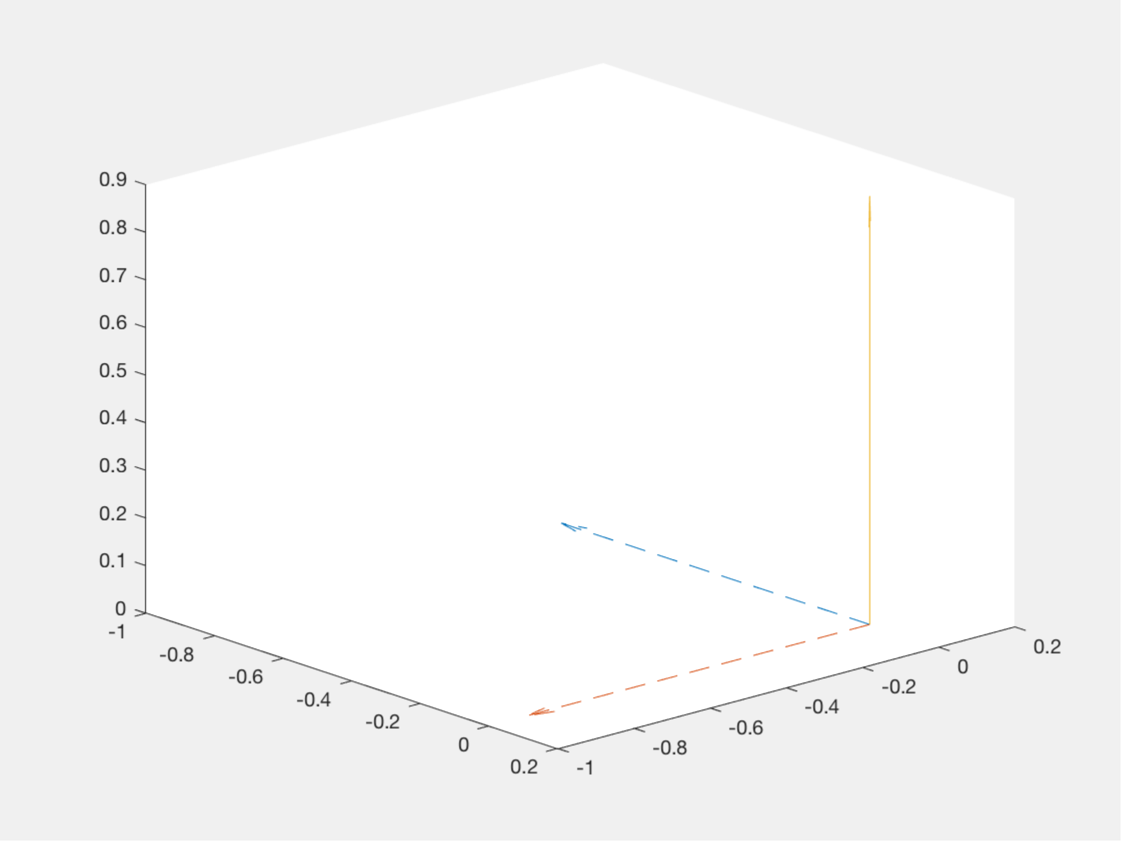


```
[theta, c, c_mag] = vectors([2 -2 1],[4 2 -4])


theta =

    1.5708


c =

     6    12    12


c_mag =

    18
```



```
[theta, c, c_mag] = vectors([-1 0 0],[0 -1 0])
```

```
theta =

    1.5708


c =

     0     0     1


c_mag =

     1
```



Published with MATLAB® R2016a