Tornillo Arquímedes lodos mineros:

[x,y]=meshgrid(0:0.1:4.2)

z=5.7.\*atan(y./x) ; mesh(x,y,z) %Helicoide recto

hold on

% cilindro

t=linspace(0,2\*pi,20);

radio=2.7

zv=[0,4.2];

[U V]=meshgrid(t,zv);

X1=radio\*cos(U);Y1=radio\*sin(U);Z1=V;

surf(X1,Y1,Z1);

axis equal

hold on

% a. Halle las ecuaciones paramétricas de la intersección de las dos

%superficies

% b. Grafique la curva.

%(Hecho a mano el hallar la intersección de las dos curvas)

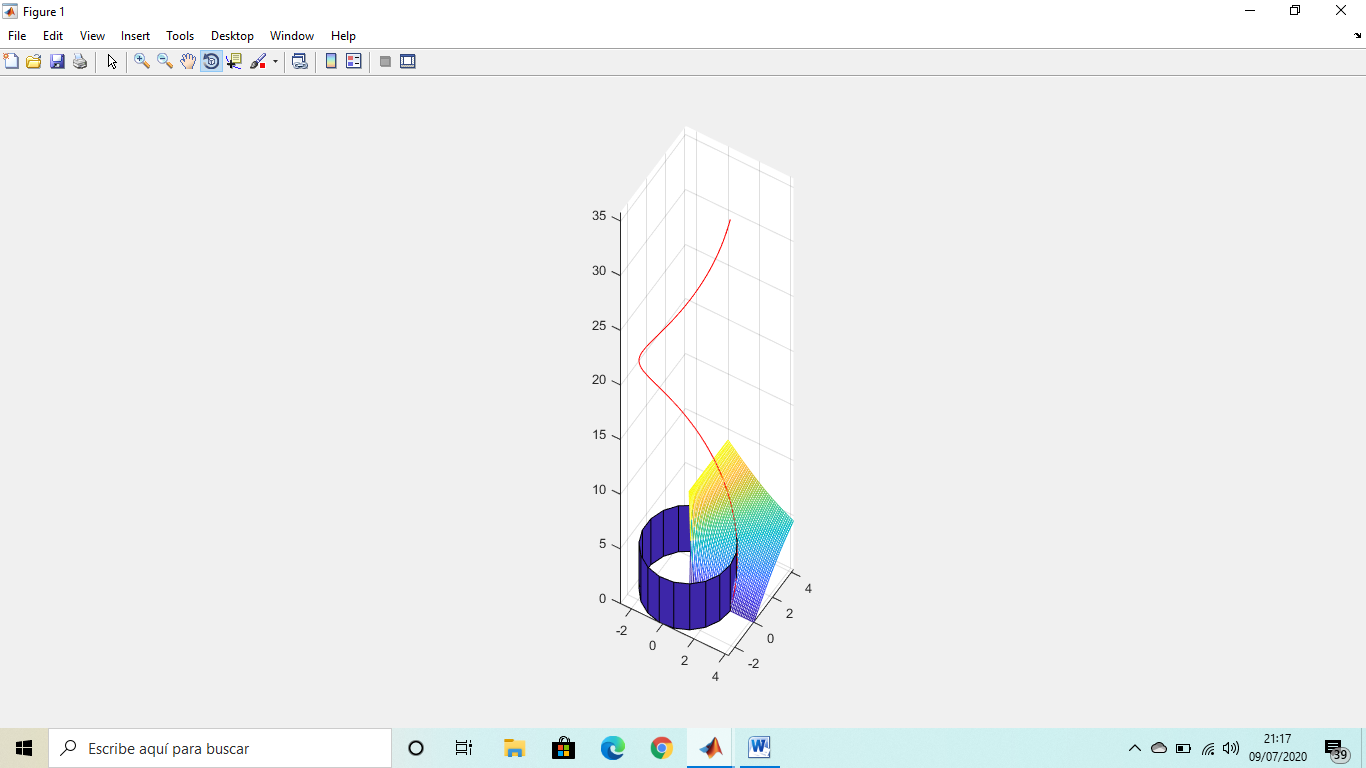
t=0:pi/32:2\*pi;

u=2.7\*cos(t);

v=2.7\*sin(t);

w=5.7\*t

plot3(u,v,w,'r')



%calculando la torsion en el problema pedido:

%Vectores que debemos definir:

%tp es el valor de t en el punto que nos indique el ejercicio donde

%queremos calcular el triedro de Frenet, En este ejercicio decia en el

%punto de la coordenada z=4.2/2 lo que me permitio sacar tp=4.2/(2\*5.7)

syms t

modulo = @(v) simplify ( sqrt ( v \* transpose(v) ) )

unitario = @(v) v / modulo (v)

dibuja\_vector = @(v,r,color) plot3 ( [r(1), r(1)+v(1)], [r(2), r(2)+v(2)], [r(3), r(3)+v(3)], color)

xt = 2.7\*cos(t), yt = 2.7\*sin(t), zt = 5.7\*t, tmin = 0, tmax = 2\*pi, tp = 4.2/(2\*5.7)

r = [ xt, yt, zt]

pretty(r)

rp = subs(r , {t}, {tp} ), xp = rp(1), yp = rp(2), zp =rp(3)

%Vectores velocidad y aceleración

ezplot3( xt, yt, zt, [tmin, tmax] ), hold on, scatter3( xp, yp, zp, 'filled' ), hold off

axis equal, title( 'CURVA TRABAJO')

v = simplify( diff (r) ), pretty(v), modv = simplify( modulo (v) ), pretty(modv) %velocidad

a = simplify( diff (v) ), pretty(a), moda = simplify( modulo (a) ), pretty(moda) %aceleración

vp = subs(v , {t}, {tp} ), ap = subs(a , {t}, {tp} ) %velocidad y aceleración en el punto

ezplot3( xt, yt, zt, [tmin, tmax]), hold on,

scatter3( xp, yp, zp, 'filled' ),

dibuja\_vector( vp, rp, 'red')

dibuja\_vector( ap, rp, 'black')

hold off

axis equal, title( 'VELOCIDAD Y ACELERACIÓN')

subplot(2,1,1), ezplot( modv, [tmin, tmax] ), xlabel ( ' Parámetro t '), ylabel( 'Velocidad' ), title('Módulo de la Velocidad')

subplot(2,1,2), ezplot( moda, [tmin, tmax] ), xlabel( ' Parámetro t '), ylabel( 'Aceleración' ), title('Módulo de la Aceleración')

%Longitud del arco de curva.

L\_t = simplify( int ( modv, t ) )

L = simplify( int (modv, t, tmin, tmax) )

double(L)

%Triedro de Frénet.

T = simplify( unitario(v) ) %Vector Tangente.

pretty (T)

N = simplify( unitario( diff (T) ) ) %Vector Normal Principal.

pretty(N)

B = simplify( cross(T,N) ) %Vector Binormal.

pretty(B)

%Triedro de Frénet en el punto.

Tp = subs(T , {t}, {tp} ), Np = subs(N , {t}, {tp} ), Bp = subs(B , {t}, {tp} )

%Dibujo del Triedro de Frénet en el punto.

ezplot3( xt, yt, zt, [tmin, tmax]), hold on,

scatter3( xp, yp, zp, 'filled' ),

dibuja\_vector( Tp, rp, 'red')

dibuja\_vector( Np, rp, 'black')

dibuja\_vector( Bp, rp, 'magenta')

hold off

axis equal, title( 'Triedro de Frénet')

%Planos del Triedro de Frénet.

syms x y z

X = [ x, y, z]

%Plano Normal.

Plano\_N = simplify( (X - r) \* transpose(T) )

pretty(Plano\_N)

%Plano Normal en el punto.

Plano\_N\_p = vpa(subs(Plano\_N , {t}, {tp} ), 3 )

%Dibujo del Plano Normal.

ezplot3( xt, yt, zt, [tmin, tmax]), hold on

scatter3( xp, yp, zp, 'filled' )

dibuja\_vector( Tp, rp, 'red')

hold off

implicitplot3d( Plano\_N\_p, 0, xp-1, xp+1, yp-1, yp+1, zp-1, zp+1, 40,'blue')

axis equal, title( 'Plano Normal y Tangente')

%Plano Rectificante.

Plano\_R = simplify( (X - r) \* transpose(N) )

pretty(Plano\_R)

%Plano Rectificante en el punto.

Plano\_R\_p = vpa(subs(Plano\_R , {t}, {tp} ), 3 )

%Dibujo del Plano Rectificante.

ezplot3( xt, yt, zt, [tmin, tmax]), hold on

scatter3( xp, yp, zp, 'filled' )

dibuja\_vector( Np, rp, 'red')

hold off

implicitplot3d( Plano\_R\_p, 0, xp-1, xp+1, yp-1, yp+1, zp-1, zp+1, 40,'blue')

axis equal, title( 'Plano Rectificante y Normal Principal')

%Plano Osculador.

Plano\_O = simplify( (X - r) \* transpose(B) )

pretty(Plano\_O)

%Plano Osculador en el punto.

Plano\_O\_p = vpa(subs(Plano\_O , {t}, {tp} ), 3 )

%Dibujo del Plano Osculador.

ezplot3( xt, yt, zt, [tmin, tmax]), hold on

scatter3( xp, yp, zp, 'filled' )

dibuja\_vector( Bp, rp, 'red')

hold off

implicitplot3d( Plano\_O\_p, 0, xp-1, xp+1, yp-1, yp+1, zp-1, zp+1, 40,'blue')

axis equal, title( 'Plano Osculador y Binormal')

%Curvatura.

kappa = simplify( modulo( diff (T) ) / modv )

pretty(kappa)

%Torsión.

tau = simplify( - diff (B) \* transpose(N) / modv )

pretty(tau)

%Dibujo de ambas.

subplot(2,1,1), ezplot( kappa, [tmin, tmax] ), xlabel( ' Parámetro t '), ylabel( 'Curvatura' ), title('CURVATURA')

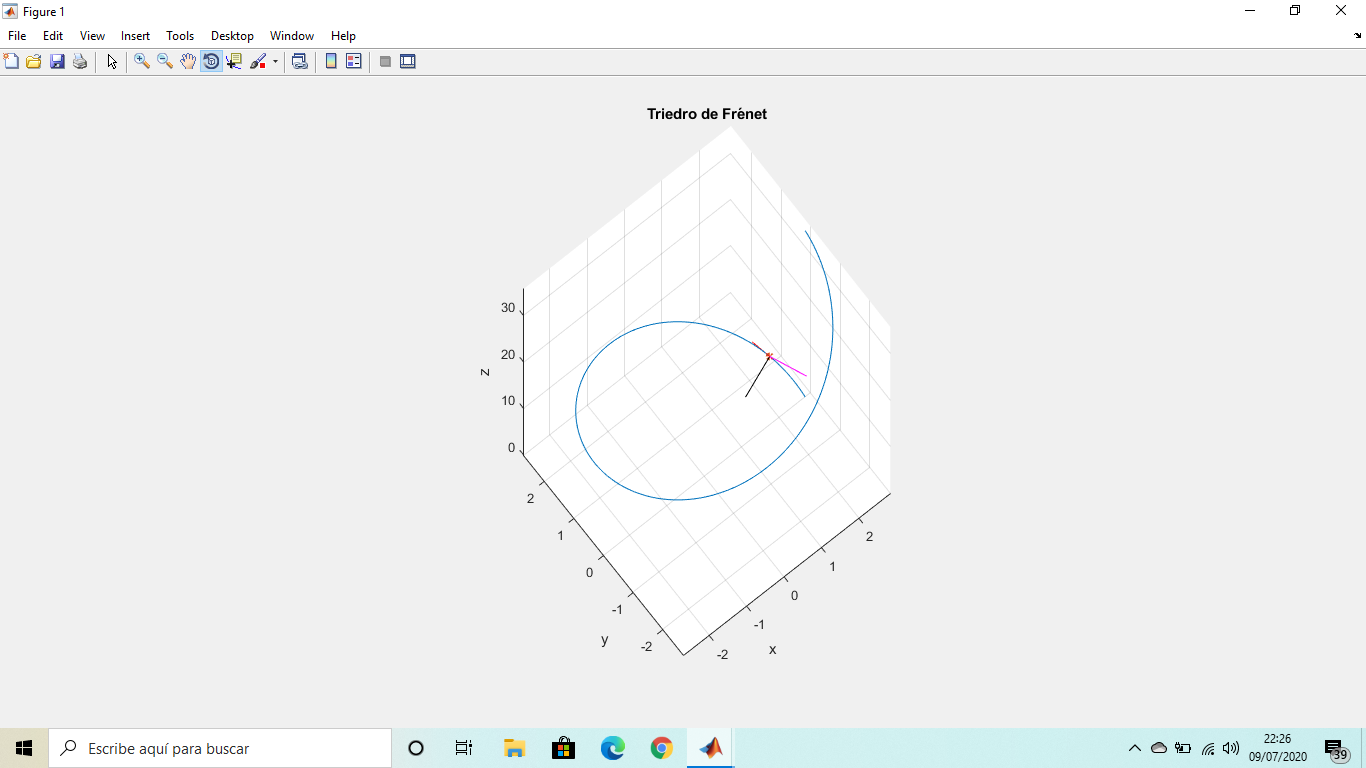
subplot(2,1,2), ezplot( tau, [tmin, tmax] ), xlabel( ' Parámetro t '), ylabel( 'Torsión' ), title('TORSIÓN')

%Aceleración Tangencial.

aT = simplify( diff (modv) ), pretty(aT)

%Aceleración Normal.

aN = simplify( kappa \* modv^2), pretty(aN)



function out=implicitplot3d(varargin)

%IMPLICITPLOT3D 3-D implicit plot

% IMPLICITPLOT3D(eq, val, xvar, yvar, zvar, xmin, xmax,

% ymin, ymax, zmin, zmax) plots an implicit equation

% eq=val, where eq is either symbolic expression of (symbolic)

% variables xvar, yvar, and zvar in the indicated ranges, or

% a string representing such an expression, and val is a number.

% If xvar, yvar, and zvar are not specified, it is assumed they are

% x, y, z in the symbolic case, or 'x', 'y',and 'z' in the

% string form of the command, respectively.

% The optional parameter plotpoints (added at the end)

% gives the number of steps in each direction between plotting points.

%

% Example: implicitplot3d('x^2+y^2+z^2', 5, -3, 3, -3, 3, -3, 3)

% plots the sphere 'x^2+y^2+z^2=5' with 'x', 'y', and 'z'

% going from -3 to 3.

% implicitplot3d('x^2+y^2+z^2', 5, -3, 3, -3, 3, -3, 3, 30)

% does the same with higher accuracy.

% implicitplot3d('x^2+y^2+z^2', 5, -3, 3, -3, 3, -3, 3, 30, 'color')

% does the same with a given color.

% written by Jonathan Rosenberg, 7/30/99

% rewritten for MATLAB 7, 8/22/05

% modified by Santiago de Vicente, 3/20/12

if nargin<8

error('not enough input arguments — need at least expression string, value, xmin, xmax, ymin, ymax, zmin, zmax');

end

if nargin==11, error('impossible number of input arguments'); end

if nargin>13, error('too many input arguments'); end

% Default value of plotpoints is 10.

plotpoints=10; color='black';

eq=varargin{1}; val=varargin{2};

stringflag=ischar(eq); % This is 'true' in the string case,

% 'false' in the symbolic case.

% Next, handle subcase where variable names are missing.

if nargin<11

if stringflag % First we deal with the string case.

xvar='x'; yvar='y'; zvar='z';

else % Now deal with the case where eq is symbolic.

syms x y z; xvar=x; yvar=y; zvar=z;

end

xmin=varargin{3}; xmax=varargin{4};

ymin=varargin{5}; ymax=varargin{6};

zmin=varargin{7}; zmax=varargin{8};

if nargin==9, plotpoints=varargin{9}; end

if nargin==10, plotpoints=varargin{9}; color=char(varargin(10)); end

end

% Next, handle subcase where variable names are included.

if nargin>11

xvar=varargin{3}; yvar=varargin{4}; zvar=varargin{5};

xmin=varargin{6}; xmax=varargin{7};

ymin=varargin{8}; ymax=varargin{9};

zmin=varargin{10}; zmax=varargin{11};

if nargin==12, plotpoints=varargin{12}; end

if nargin==13, plotpoints=varargin{12}; color=char(varargin(10)); end

end

if stringflag

F = vectorize(inline(eq,xvar,yvar,zvar));

else

F = inline(vectorize(eq),char(xvar),char(yvar),char(zvar));

end

[X Y]= meshgrid(xmin:(xmax-xmin)/plotpoints:xmax, ymin:(ymax-ymin)/plotpoints:ymax);

% Go through zvalues one at a time. For each one, plot corresponding

% contourplot in x and y, with that z-value. We could use "contour"

% except that it makes a "shadow", so we copy some of

% the code of "contour".

for z=zmin:(zmax-zmin)/plotpoints:zmax

lims = [min(X(:)),max(X(:)), min(Y(:)),max(Y(:))];

c = contours(X,Y,F(X,Y,z), [val val]);

limit = size(c,2);

i = 1;

h = [];

while(i < limit)

npoints = c(2,i);

nexti = i+npoints+1;

xdata = c(1,i+1:i+npoints);

ydata = c(2,i+1:i+npoints);

zdata = z + 0\*xdata; % Make zdata the same size as xdata

line('XData',xdata,'YData',ydata,'ZData',zdata,'Color',color); hold on;

i = nexti;

end

end

view(3)

xlabel(char(xvar))

ylabel(char(yvar))

zlabel(char(zvar))

title([char(eq),' = ',num2str(val)], 'Interpreter','none')

hold off

end