Diego Emilio Dominguez Tableros

Plot the edges of a regular pentagon inscribed in the unit circle. Then modify the plot to make it a five-pointed star, with the inner points lying on the circle of radius 0.4.

Para graficar la circunferencia unitaria en el plano xy se utilizó la parametrización de la circunferencia dada por $x = \cos(t)\,, y = \sin(t)$ con $0 \leq t \leq 2\pi$.

Para generar el pentágono inscrito se programó una función auxiliar `dibuja_pentagono()` que recibe como parámetro un vértice inicial del pentágono y devuelve un vector con los cinco vértices generados. Su funcionamiento se basa en que cada vértice del pentágono está rotado un ángulo de $2\pi/5$ con respecto al punto anterior, para lo cual se utilizó la matriz de rotación de forma iterada:

$$\begin{pmatrix}\cos(2\pi/5) & -\sin(2\pi/5)\\ \sin(2\pi/5) & \cos(2\pi/5)\end{pmatrix}\begin{pmatrix}x\\ y\end{pmatrix} = \begin{pmatrix}x\cos(2\pi/5) - y\sin(2\pi/5)\\ x\sin(2\pi/5) + y\cos(2\pi/5)\end{pmatrix} = \begin{pmatrix}x_{rotado}\\ y_{rotado}\end{pmatrix}$$

```
%Función dibuja_pentagono
%Genera los vértices (x,y) de un pentágono regular
%Esta función genera un vector de puntos en el plano
%que son los vértices de un pentágono regular
%Se pide un vértice inicial (x_i,y_i)
function [x_pent,y_pent] = dibuja_pentagono(x_ini,y_ini)
    x_pent=zeros(6,1); x_pent(1)=x_ini; x_pent(6)=x_ini;
    y_pent=zeros(6,1); y_pent(1)=y_ini; y_pent(6)=y_ini;
    ang=2*pi/5;
    for i=[1:4]
        x_pent(i+1)=x_pent(i)*cos(ang)-y_pent(i)*sin(ang);
        y_pent(i+1)=x_pent(i)*sin(ang)+y_pent(i)*cos(ang);
    end
end
```

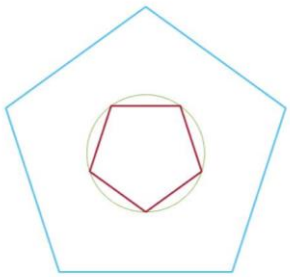


Fig. 2. Dos pentágonos inscritos en las circunferencias de radios 1.0 y 0.4.

De manera que, para graficar el pentágono inscrito, basta con graficar con `plot()` los dos vectores que regresa esta función. Se tomó el vértice generador en el punto (0,1) y se obtuvo el resultado mostrado en la Fig. 1.

Para generar la estrella se tuvo en cuenta un segundo pentágono inscrito en la circunferencia de radio 0.4 con vértice inicial en el punto (0,-0.4), como se muestra en la Fig. 2. La idea fue tomar los puntos del pentágono grande en combinación con los puntos del pentágono pequeño en un orden tal que dibujen la estrella. Recordemos que la función `dibuja_pentagono()` devuelve vectores con los vértices, así que utilizando un ciclo `for` se combinaron los puntos de cada pentágono en dos vectores nuevos que contienen ahora los vértices de la estrella.

Finalmente se graficaron los vectores con los puntos de la estrella, obteniendo el resultado pedido, mostrado en la Fig. 3.

```
%%%%%Opcional: Graficar círculos de radio 1 y 0.4
%t=[0:2*pi/100:2*pi];
%circ_1x=cos(t); circ_1y=sin(t);
%plot(circ_1x,circ_1y);
%circ_4x=0.4*cos(t); circ_4y=0.4*sin(t);
% plot(circ_4x,circ_4y);

%Propiedades y limites de los ejes
axis square;
axis([-1.2,1.2,-1.2,1.2]);
xlabel('Eje X'); ylabel('Eje Y');
hold on;

%Generamos los puntos de los dos pentagonos
[x_g,y_g]=dibuja_pentagono(0,1);
[x_p,y_p]=dibuja_pentagono(0,-0.4);
plot(x_g,y_g,'LineWidth',3);
%plot(x_p,y_p,'LineWidth',2);

%Generamos los puntos de la estrella
x_estrella=zeros(11,1);
x_estrella(11)=x_g(1);
y_estrella=zeros(11,1);
y_estrella(11)=y_g(1);
pos_g=1; pos_p=4;
for i=[1:10]
    if(mod(i,2)==1)
        x_estrella(i)=x_g(pos_g);
        y_estrella(i)=y_g(pos_g);
        pos_g=pos_g+1;
    else
        x_estrella(i)=x_p(pos_p);
        y_estrella(i)=y_p(pos_p);
        pos_p=pos_p+1;
        if(pos_p==6)
            pos_p=1;
        end
    end
end
%Graficamos los puntos de la estrella
plot(x_estrella,y_estrella,'Linewidth',3);
```

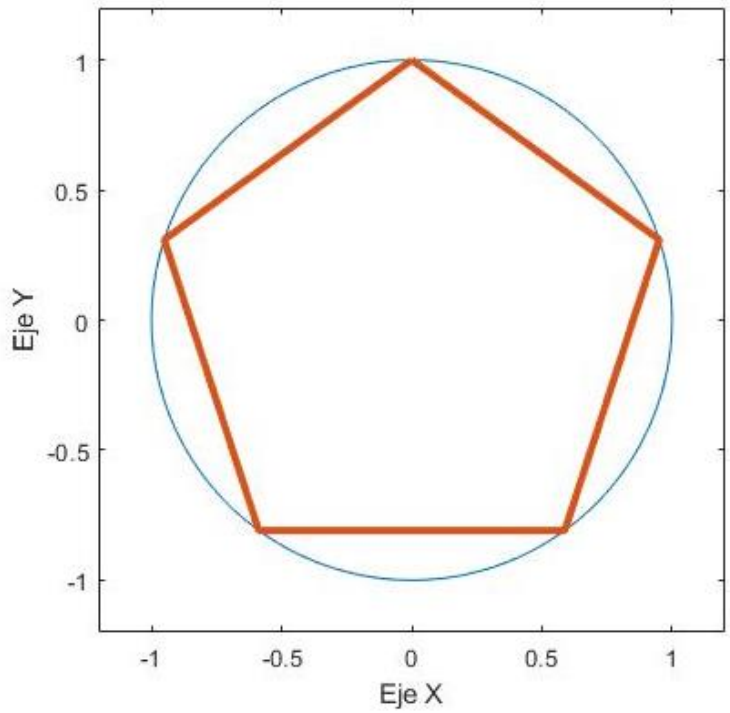


Fig. 1. Pentágono inscrito en la circunferencia unitaria.

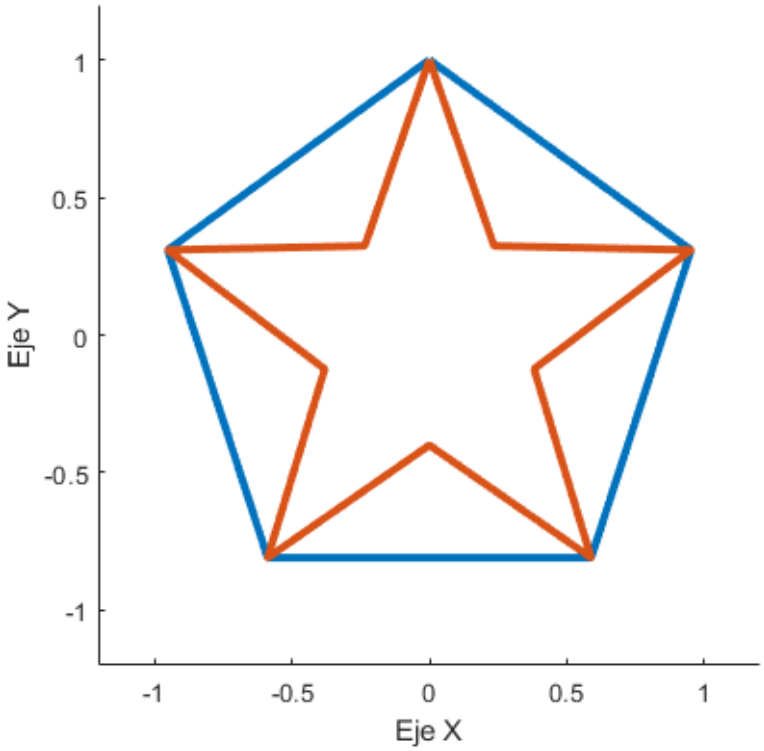


Fig. 3. Estrella de cinco puntas inscrita en el pentágono.

Make an mfile with a "for loop" to graph the family of line segments l_θ through the origin in three-dimensional space

$$l_\theta = \{(t\cos(\theta), t\sin(\theta), 0.7\,t), -1 \le t \le 1\}$$

where $\theta = 2\pi j/n$, $j = 0, \cdots, n$. Choose $n = 24$. What would be the resultant figure as $n \to \infty$?

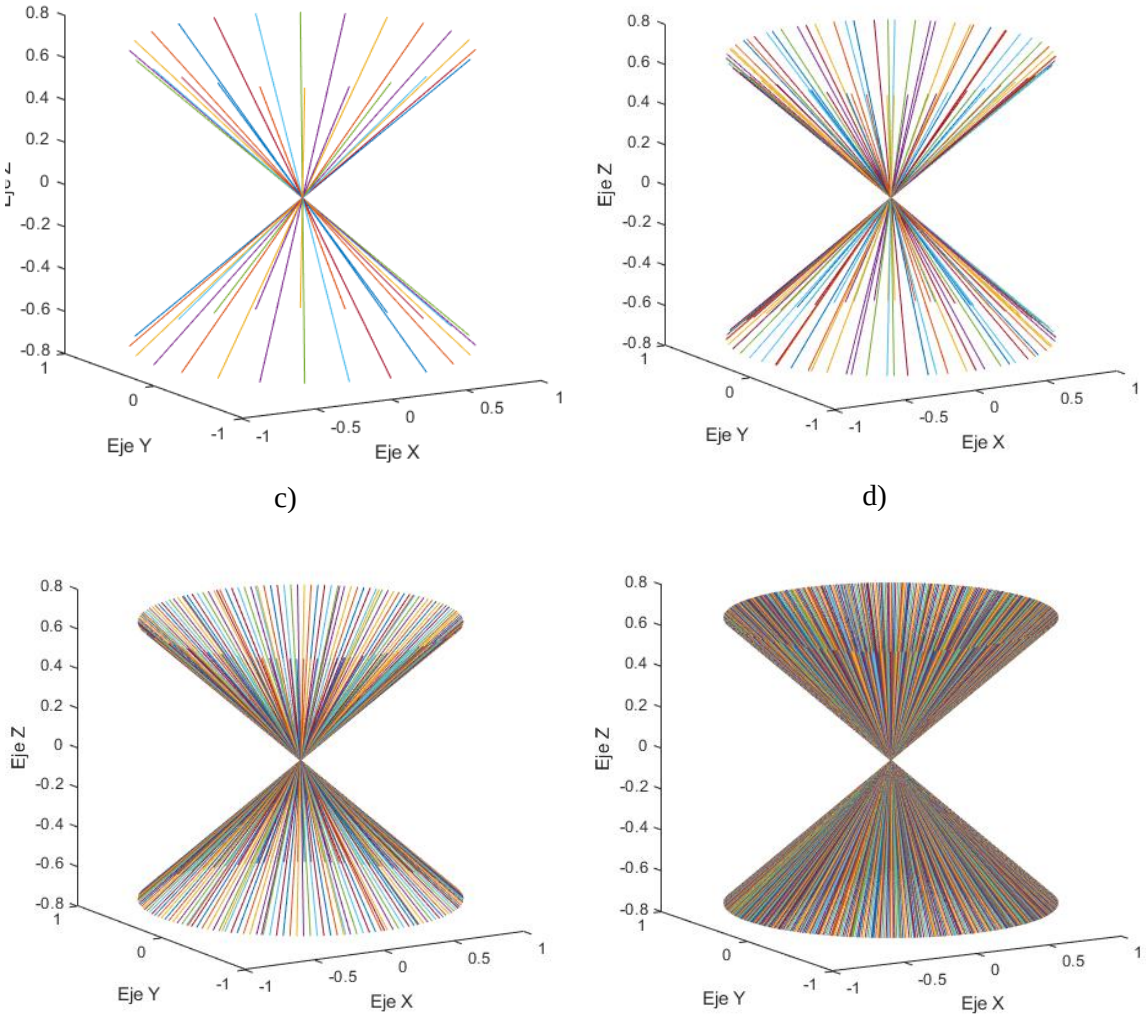
Este ejercicio dice explícitamente que debemos graficar la familia de segmentos de línea indicados. Lo único que hay que hacer es un ciclo `for` que itere sobre j todos los valores desde 0 a 2π con incrementos de $2\pi/n$. Se utilizó la función `plot3` con sus respectivos vectores para cada segmento.

```
%Creamos el vector t
t=[-1:1];
%Pedimos n, el numero de segmentos a graficar
n=input('n: ');
%Hacemos variar j desde 0 hasta n
for j=[0:n]
    %Calculamos el theta para este j
    theta=2*j*pi/n;
    %Generamos los componentes del segmento
    lx=t*cos(theta);
    ly=t*sin(theta);
    lz=t*0.7;
    %Graficamos el segmento
    plot3(lx,ly,lz);
    %Propiedades de ejes
    hold on;
    axis square;
end
```

Los resultados obtenidos para `n=24`, `50`, `150` y `500` se muestran en la Fig. 4. Se puede notar que, conforme damos valores cada vez más grandes para `n`, se divide en más partes el intervalo $[0,2\pi]$, obteniendo un mayor número de segmentos más próximos uno del otro.

Este conjunto de rectas forma una superficie cónica en el espacio sobre y debajo del plano xy, asemejándose mucho a la gráfica de la superficie dada por $z = \pm\sqrt{x^2 + y^2}$ ya que conforme se toman valores más grandes de z, se forman circunferencias concéntricas en este eje y con radios cada vez más grandes.

a) b)



c) d)

Fig. 4. Conjunto de rectas l_θ para distintos valores de n. a) n=24. b) n=50. c) n=150. d) n=500.

Use the command `plane` and a `for` loop to construct an array of 8 parallel cooling fins. The fins should be square, with side of length 2. They should be centered on the x axis, making an angle of $\pi/6$ with the yz plane. The perpendicular distance between the fins should be one unit. What is the spacing of the fins in a direction parallel to the x axis? Make the fin at one end of the array contain the origin.

El comando `plane` escrito por el autor se encuentra en la página web dada al inicio del libro. Esta función gráfica en el espacio tridimensional un plano, lo cual es bastante útil para el ejercicio. Recibe como parámetros: un **punto en el plano** a graficar (vector con tres entradas), un **vector normal al plano** a graficar (vector con tres entradas), el **semiancho** de la porción de plano (escalar) y el **semialto** de la porción de plano (escalar).

Para determinar el vector normal a los planos consideremos la Fig. 5 donde se muestra el plano xz (el eje y sale y entra de la página).

Dado que el plano debe formar un ángulo de $\pi/6$ con el plano yz entonces podemos elegir un vector normal $\vec{N}$ tal que forme un ángulo de $\pi/6$ con el eje x y solo tenga componente en x y en z. Si además consideramos que sea unitario entonces este vector debe ser de la forma

$$\vec{N} = \left(\cos\left(\frac{\pi}{6}\right), 0, \sin\left(\frac{\pi}{6}\right)\right)$$

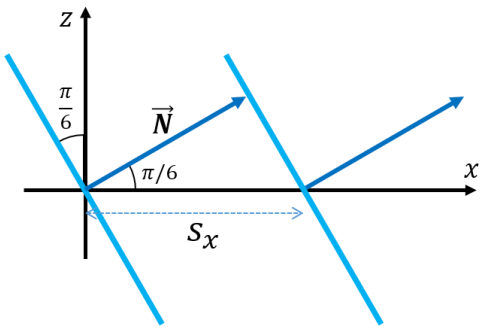


Fig. 5. Vector normal a los planos y separación entre ellos.

Notemos que $\|\vec{N}\| = 1$. Este vector es común para todos los planos a graficar pues todos deben cumplir la condición de formar $\pi/6$ con el plano yz.

Ahora, como además se pide que cada plano esté separado por una unidad, aprovechando nuestra definición de $\vec{N}$ podemos ver que, al considerar el triángulo rectángulo formado entre dos planos consecutivos, el eje x y el vector $\vec{N}$, la distancia s_x que separa cada plano sobre el eje x necesariamente debe verificar $\cos(\pi/6) = 1/s_x$, de donde obtenemos que

$$s_x = 1/\cos\left(\frac{\pi}{6}\right)$$

Por lo tanto, consideraremos que los puntos que necesariamente pertenecen a cada plano deben cumplir estar separados por la distancia s_x, es decir, para el $j-$ésimo plano tiene la forma

$$P = \left(\frac{(j-1)}{\cos\left(\frac{\pi}{6}\right)}, 0,0\right), j = 1, 2, \cdots, 8$$

Finalmente, se realizó el script correspondiente tomando en cuenta todos estos hechos para cada llamada a la función `plane`, obteniendo los 8 planos que cumplen estar centrados en el eje x, formar un ángulo de $\pi/6$ con el plano yz y además estar separados perpendicularmente en una unidad.

El resultado obtenido se muestra en la Fig. 6.

```
%La normal de los planos es un vector que forme un angulo
%de pi/6 con el plano YZ, se eligió el vector unitario
%en el plano ZX dado por (cos(pi/6),0,sin(pi/6))
normal=[cos(pi/6),0,sin(pi/6)];
%Un for para graficar los 8 planos
for i=1:8
%Sacamos la coordenada en X del punto inicial dentro del plano
%Recordemos que el primer plano está en el origen (0,0,0) y para
%que estén separados perpendicularmente por 1 unidad deben estar
%separados una distancia de 1/cos(pi/6) en el eje X
coord_x=(i-1)/cos(pi/6);
%Usamos la función Plane que propone el autor
plane([coord_x,0,0],normal,1,1);
hold on;
end
%Graficamos auxiliarmente los ejes coordenados
plot3([0,9],[0,0],[0,0],'b','LineWidth',3);
plot3([0,0],[-1,1],[0,0],'b','LineWidth',3);
plot3([0,0],[0,0],[-1,1],'b','LineWidth',3);
xlabel('x'); ylabel('y'); zlabel('z');
```

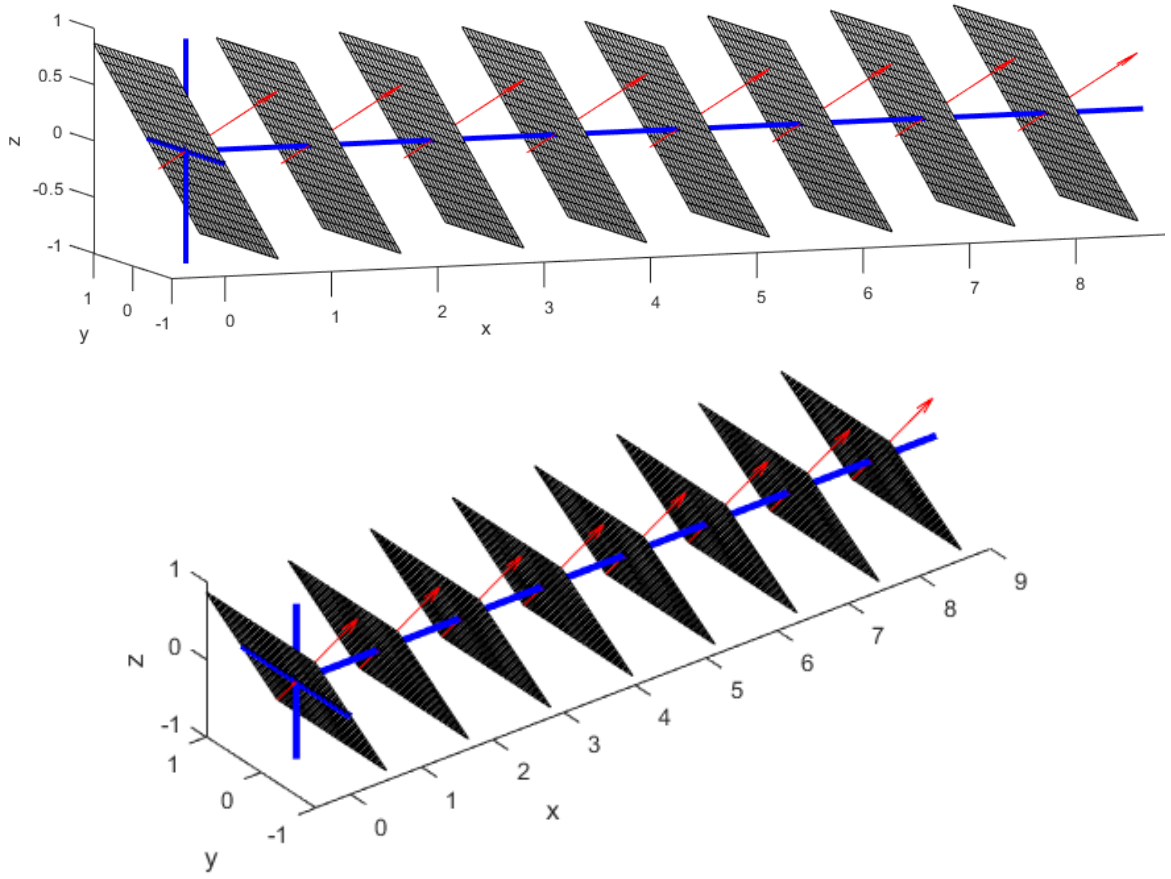


Fig. 6. Planos generados por `plane()` distanciados en una unidad y sobre el eje x.

a) Use the parameterization $x(t) = 2\cos(t)\,, y(t) = \sin(t)\,, 0 \le t \le 2\pi$, to graph the elllipse $x^2/4 + y^2 = 1$ in the x, y plane.
b) Calculate by hand the velocity, speed, and acceleration. Where is the speed greatest, and where is the speed the smallest?
c) Use the command arrow to attach the velocity vectors at the points $P(t)$ for the times $t = 0, \pi/2, \pi, 3\pi/2$.
d) Calculate the curvature by hand. Then write the speed and curvature as inline functions and graph them together on $[0,2\pi]$. Where are máximum and mínimum values of the speed attained? Same questions for curvature.
e) Use MATLAB to approximate the arc length using a polygonal approximation with 100 segments. Then use `quad8` to estimate the arc length integral of the speed. In both cases, check your method of calculation on a circle t osee if it is working correctly.

a) Se graficó la elipse paramétricamente utilizando `plot`. El resultado se muestra en la Fig. 7.

b) Consideremos el vector de posición de los puntos de la elipse $\boldsymbol{r}(t)$ parametrizado por t. Con sus primera y segunda derivadas obtenemos los vectores de velocidad y aceleración $\boldsymbol{v}(t)$, $\boldsymbol{a}(t)$ y tomando la magnitud de la velocidad obtenemos la rapidez $s(t)$.

$$\boldsymbol{r}(t) = (2\cos(t)\,, \sin(t))$$
$$\boldsymbol{v}(t) = \frac{d}{dt}\boldsymbol{r}(t) = (-2\sin(t)\,, \cos(t))$$
$$\boldsymbol{a}(t) = \frac{d}{dt}\boldsymbol{v}(t) = (-2\cos(t)\,, -\sin(t))$$
$$s(t) = \|\boldsymbol{v}(t)\| = \sqrt{4\sin^2 t + \cos^2 t}$$

Utilizando el criterio de la segunda derivada se determina que $t = 0, \pi, 2\pi$ son máximos y $t = \pi/2, 3\pi/2$ son mínimos de la rapidez $s(t)$.

c) Se utilizó `quiver` para graficar sobre la elipse los vectores de velocidad correspondientes a los máximos y mínimos encontrados en el inciso anterior, es decir, en $t = 0, \pi/2, \pi, 3\pi/2$. El resultado se muestra en la Fig. 7.

d) La curvatura κ es una función escalar definida para cada punto de la curva. En este caso

$$\kappa(t) = \frac{\boldsymbol{v} \times \boldsymbol{a}}{\|\boldsymbol{v} \times \boldsymbol{a}\|} = \frac{2}{\left(\sqrt{4\sin^2 t + \cos^2 t}\right)^3} = \frac{2}{s^3(t)}$$

Y la rapidez fue calculada en el inciso b), $s(t) = \|\boldsymbol{v}(t)\| = \sqrt{4\sin^2 t + \cos^2 t}$. Usando `plot` se graficaron en función de t en el intervalo $[0,2\pi]$, el resultado se muestra en la Fig. 8.
De las expresiones de la curvatura y la rapidez podemos ver que, los máximos de la rapidez $s(t)$ que son $t = 0, \pi, 2\pi$ corresponden con los mínimos de la curvatura; así mismo los máximos de la rapidez $t = \pi/2, 3\pi/2$ corresponden con los máximos de la curvatura. Esto se puede corroborar fácilmente viendo las gráficas obtenidas en la Fig. 8.

e) Dado que hemos empleado un vector para representar t de 0 a 2π con incrementos de $2\pi/100$, la aproximación de la longitud de arco de la elipse por aproximación poligonal se realizó con el vector $\boldsymbol{r}(t)$ que tiene 100 puntos que generan la elipse. La idea es tomar cada segmento entre dos puntos, tomar su norma y sumarlo a la aproximación. Se obtuvo un valor para la longitud de arco de

$$l_{poligono} = 9.686854613337783$$

También, la longitud de arco de la elipse viene dada por

$$l_{integral} = \int_0^{2\pi} \|\boldsymbol{v}(t)\|\, dt = \int_0^{2\pi} \sqrt{4\sin^2 t + \cos^2 t}\; dt$$

Se integró esta expresión numéricamente con la función `integral` (se pide usar `quad8` pero Matlab propone sustituirla con integral para un mejor funcionamiento), obteniendo un valor más aproximado a la longitud de arco real

$$l_{integral} = 9.688448220547675$$

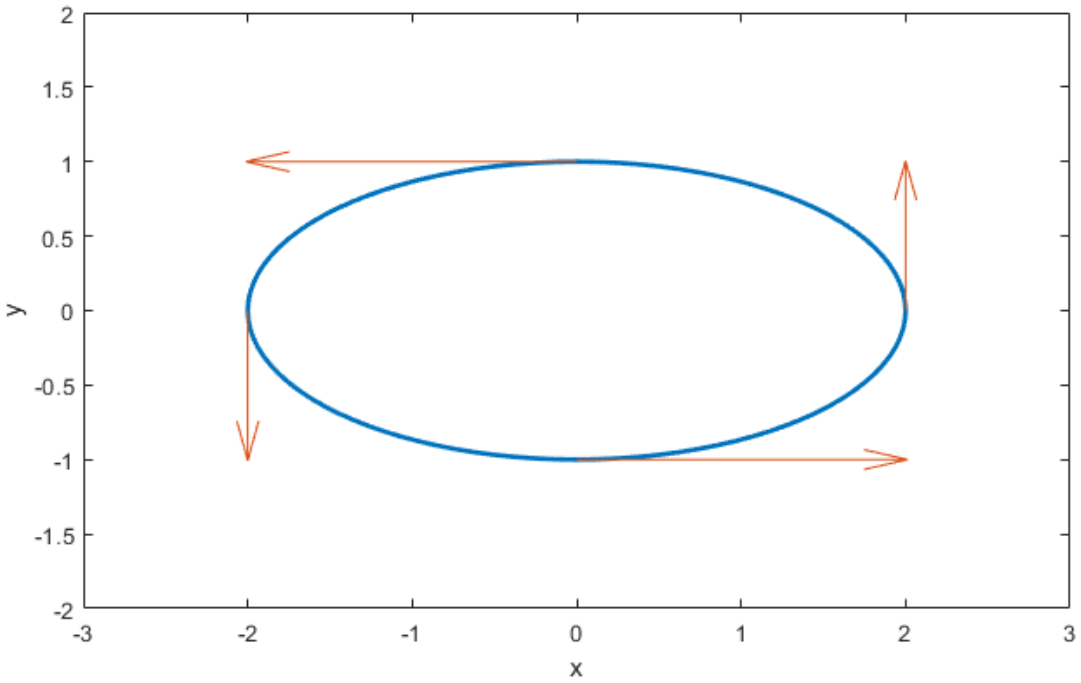


Fig. 7. Elipse parametrizada y vectores velocidad en sus máximos y mínimos.

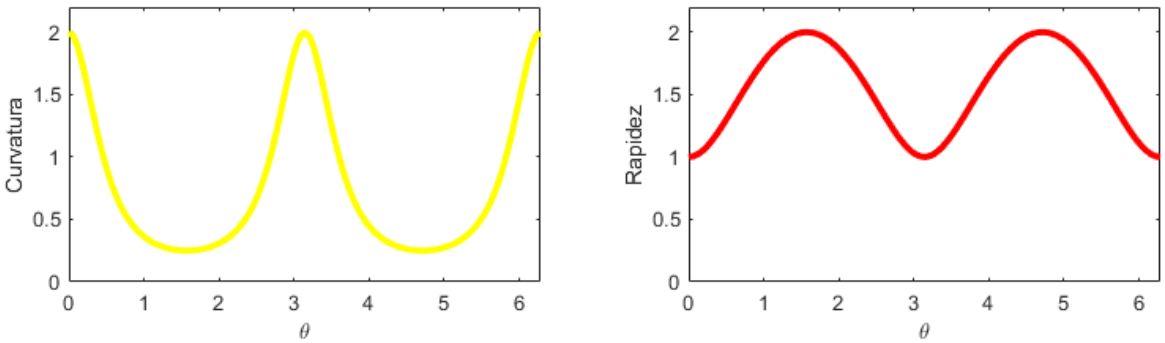


Fig. 8. Gráficas de la curvatura κ y la rapidez s en función de t.

```
%Creamos el vector t de 0 a 2pi
t=0:2*pi/100:2*pi;
%Creamos los puntos de la elipse parametricamente
x_elip=2*cos(t);
y_elip=sin(t);
%Graficamos en la Figure 1 el elipse
figure(1);
plot(x_elip,y_elip,'LineWidth',2);
xlabel('x'); ylabel('y');
axis([-3,3,-2,2]);
hold on;

%Graficamos los vectores de velocidad en t=0,pi/2,pi,3pi/2
t_aux=[0,pi/2,pi,3*pi/2];
p_x=2*cos(t_aux); p_y=sin(t_aux);
v_x=-2*sin(t_aux); v_y=cos(t_aux);
quiver(p_x,p_y,v_x,v_y);

%Graficamos en la Figure 2 las graficas de la curvatura y la rapidez
figure(2);
curvatura=2./((sqrt(4.*sin(t).*sin(t)+cos(t).*cos(t))).^3);
subplot(1,2,1);
plot(t,curvatura,'y','LineWidth',3);
xlabel('\theta'); ylabel('Curvatura'); axis([0,2*pi,0,2.2]);

rapidez=sqrt(4.*sin(t).*sin(t)+cos(t).*cos(t));
subplot(1,2,2);
plot(t,rapidez,'r','LineWidth',3);
xlabel('\theta'); ylabel('Rapidez'); axis([0,2*pi,0,2.2]);

%Calculamos la longitud de arco del elipse
format long;
%Método de poligonalización a 100 segmentos
long_segmentos=0;
for i=1:100
    dx=x_elip(i+1)-x_elip(i);
    dy=y_elip(i+1)-y_elip(i);
    dr=[dx,dy];
    long_segmentos=long_segmentos+norm(dr);
end
disp('Longitud del elipse por segmentos: ');
disp(long_segmentos);
%Método de integración numérica de la longitud de arco
integrando=@(o) sqrt(4.*sin(o).*sin(o)+cos(o).*cos(o));
long_integral=integral(integrando,0,2*pi);
disp('Longitud del elipse por integral númerica: ');
disp(long_integral);
```

A projectile with mass $m = 1$ is fired at an angle θ from the horizontal with a speed v_0. The components of its motion are

$$\boldsymbol{r}(t) = [v_0 t \cos\theta\,, v_0 t \sin\theta - g t^2/2]$$

We are using units of feet and seconds, so $g = 32\ ft/\mathrm{s}^2$. Let $v_0 = 50\ ft/s$.

a) Let t=0:0.1:5. Use the two-dimensional plotting command plot to plot the trajectories for $\theta = 10,20,30,40,50,60,70,80$ degrees. Plot all these curves on the same graph using the command hold on. Use the command axis([0 80 0 50]) to cut them off when they hit the ground.

b) By making further experiments, find value of θ that yields the máximum range. Then make an analytic calculation to confirm your result. What is the máximum range? Find a formula for the máximum range in terms of the initial speed v_0.

a) Se graficaron las trayectorias de los proyectiles a los ángulos pedidos utilizando `plot` y la parametrización de su posición en función de t. El resultado se muestra en la Fig. 9.

b) El máximo alcance horizontal se logra a un ángulo de 45 grados ($\pi/2\ rad$) con respecto a la horizontal. Este hecho puede demostrarse a partir de que el tiempo de vuelo (tiempo en que se alcanza el eje X, logrando un alcance horizontal) es

$$t = 2v_0 \sin\theta\ /g$$

Entonces para v_0 constante, el alcance horizontal en función de θ es

$$x(\theta) = \frac{2v_0^2}{g}\sin\theta\cos\theta$$

Aplicando el criterio de la segunda derivada se determina que en el intervalo de $[0, \pi/2]$ hay un máximo en $\theta = \pi/4$, por lo tanto, a este ángulo el alcance horizontal es máximo. Esta trayectoria se graficó en conjunto con las otras órbitas mostradas en la Fig. 9.
Explícitamente el alcance horizontal máximo es

$$x_{max} = x\left(\frac{\pi}{4}\right) = \frac{2v_0^2}{g}\sin\left(\frac{\pi}{4}\right)\cos\left(\frac{\pi}{4}\right) = \frac{2v_0^2}{g}\left(\frac{\sqrt{2}}{2}\right)\left(\frac{\sqrt{2}}{2}\right) = \frac{v_0^2}{g}$$

La ecuación $x(\theta)$ funciona para una v_0 constante. Si hacemos ahora θ constante y v_0 una variable, el alcance horizontal es ahora

$$x(v_0) = \frac{2\sin\theta\cos\theta}{g}v_0^2$$

Que es la expresión requerida del alcance horizontal en función de la velocidad inicial.

```
%Parámetros del tiro
masa=1;
g=32;
v_ini=50;
%Vector t requerido
t=0:0.01:5;
%Graficación de la trayectoria con mayor alcance horizontal
x=v_ini*cos(pi/4)*t;
y=v_ini*sin(pi/4)*t-0.5*g*t.^2;
plot(x,y,'k','Linewidth',3);
hold on;
axis([0 80 0 40]);
xlabel('x');
ylabel('y');
%Graficación de todas las trayectorias requeridas
for theta_grad=10:10:80
    theta=(theta_grad/180)*pi;
    x=v_ini*cos(theta)*t;
    y=v_ini*sin(theta)*t-0.5*g*t.^2;
    plot(x,y,'--');
    hold on;
end
```

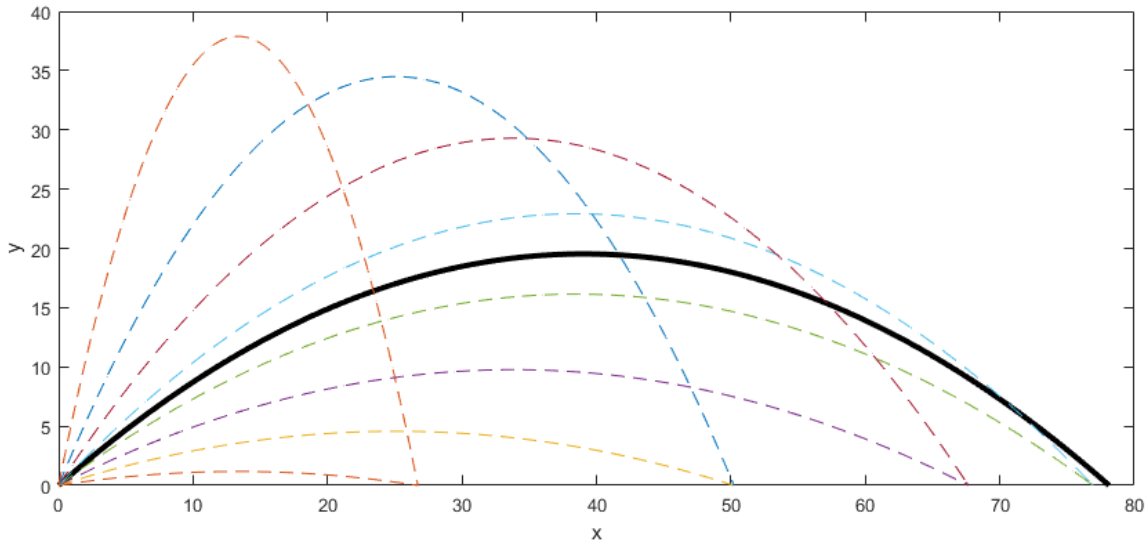


Fig. 9. Trayectorias parabólicas de un proyectil a distintos ángulos de la velocidad inicial con respecto a la horizontal. Se remarca en negro la trayectoria de máximo alcance.

Métodos numéricos Diego Emilio Dominguez Tableros
Hw02

In this exercise, we shall construct the orbit of a moon circling a planet. The planet in turn follows the elliptical orbit parameterized by $\boldsymbol{r_0}(t) = [3\cos t\,, 2\sin t\,, 0]$.

a) Calculate the normal $\boldsymbol{N}$ of the planet's orbit by hand, and show that the binormal $\boldsymbol{B} = [\boldsymbol{0}, \boldsymbol{0}, \boldsymbol{1}]$.

b) The moon circles the planet in an orbit with radius ρ. It makes 20 complete orbits of the planet in the time the planet completes one elliptical orbit. We use the normal and binormal vectors to describe the moon's position relative to the planet.

For convenience, let $\boldsymbol{n} = -\boldsymbol{N}$. $\boldsymbol{n}$ is the normal vector to the ellipse that points to the exterior. Assume that at time $t = 0$, the moon starts at the position $\boldsymbol{r} = \boldsymbol{r_0}(0) + \rho\boldsymbol{n}$. From the point of view of someone on the planet, the motion of the moon is described by the rotation

$$\boldsymbol{r_1}(t) = \rho\cos(20t)\boldsymbol{n} + \rho\sin(20t)\boldsymbol{B}$$

Write out by hand the components $x(t), y(t), z(t)$ of the combined motion

$$\boldsymbol{r}(t) = \boldsymbol{r_0}(t) + \boldsymbol{r_1}(t)$$

c) Write a script to graph the orbit of the planet and of the moon with $\rho = 0.2$.

a) Primero se determinaron los vectores de velocidad, aceleración y tangente unitaria del vector de posición de la tierra, $\boldsymbol{r_0}(t) = (3\cos t\,, 2\sin t\,, 0)$, obteniendo:

$$\boldsymbol{v} = \boldsymbol{r_0'} = (-3\sin t\,, 2\cos t\,, 0)$$
$$\boldsymbol{a} = \boldsymbol{v_0'} = (-3\cos t\,, -2\sin t\,, 0)$$
$$\boldsymbol{T} = \frac{\boldsymbol{v_0}}{\|\boldsymbol{v_0}\|} = \frac{(-3\sin t\,, 2\cos t\,, 0)}{\sqrt{9\sin^2 t + 4\cos^2 t}}$$

Luego, el vector normal $\boldsymbol{N}$ es

$$\boldsymbol{N} = \frac{\boldsymbol{T'}}{\|\boldsymbol{T'}\|} = \frac{\boldsymbol{a} - (\boldsymbol{a}\cdot\boldsymbol{T})\boldsymbol{T}}{\|\boldsymbol{a} - (\boldsymbol{a}\cdot\boldsymbol{T})\boldsymbol{T}\|} = \frac{(-2\cos t\,, -3\sin t\,, 0)}{\sqrt{9\sin^2 t + 4\cos^2 t}}$$

Y el vector binormal $\boldsymbol{B}$ es

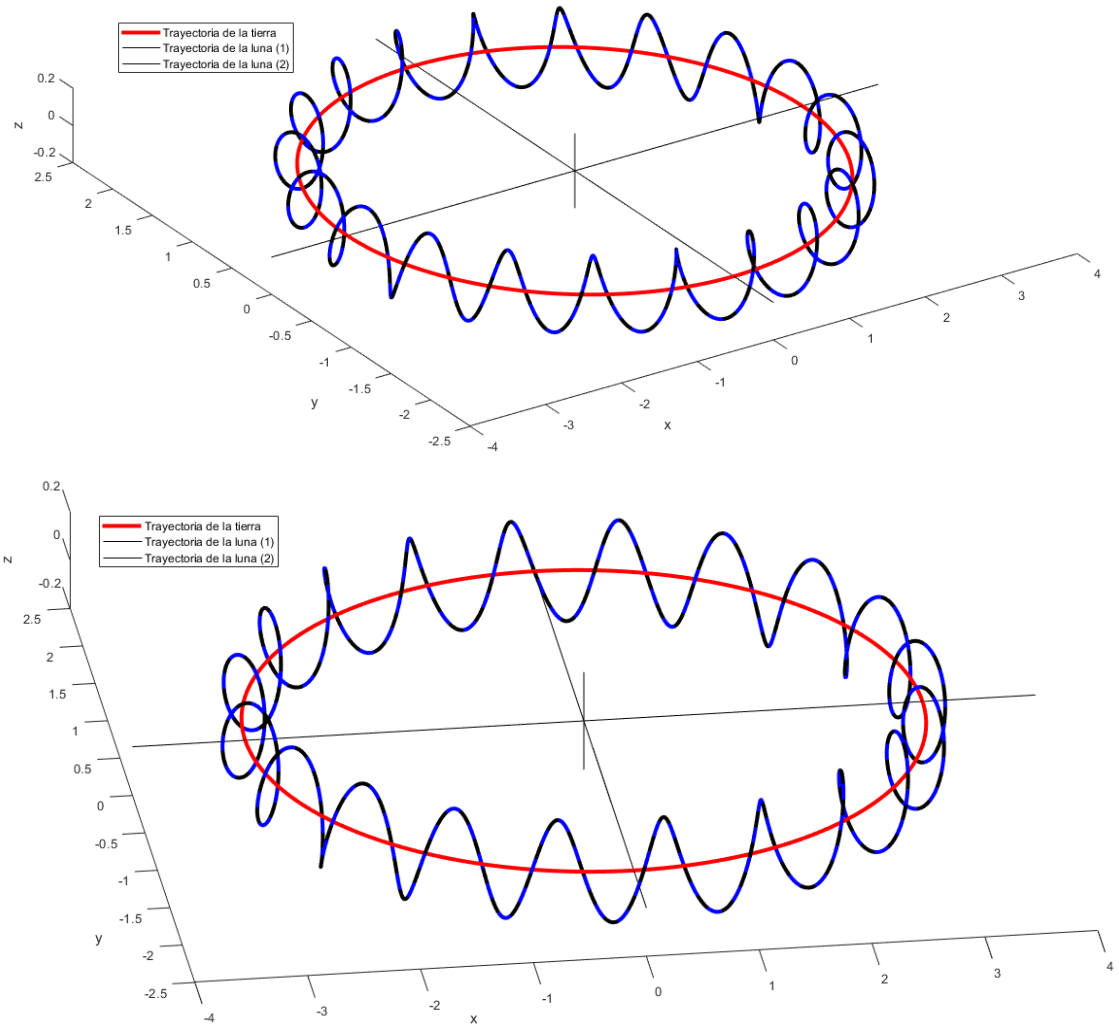
$$\boldsymbol{B} = \boldsymbol{T}\times\boldsymbol{N} = \frac{\boldsymbol{v}\times\boldsymbol{a}}{\|\boldsymbol{v}\times\boldsymbol{a}\|} = \frac{\begin{vmatrix} \hat{\imath} & \hat{\jmath} & \hat{k} \\ -3\sin t & 2\cos t & 0 \\ -3\cos t & -2\sin t & 0 \end{vmatrix}}{\|\boldsymbol{v}\times\boldsymbol{a}\|} = \frac{[0,0,6]}{\|[0,0,6]\|} = [0,0,1]$$

b) Consideremos que $\boldsymbol{n} = -\boldsymbol{N}$. Para encontrar el vector posición de la luna con respecto al sistema de referencia de la tierra $\boldsymbol{r}(t) = \boldsymbol{r_0}(t) + \boldsymbol{r_1}(t)$ tenemos que

$$\boldsymbol{r}(t) = \boldsymbol{r_0}(t) + \boldsymbol{r_1}(t) = (3\cos t\,, 2\sin t\,, 0) + \rho\cos(20t)\boldsymbol{n} + \rho\sin(20t)\boldsymbol{B}$$
$$= (3\cos(t)\,, 2\sin(t)\,, 0) + \frac{\rho(2\cos(t)\cos(20t)\,, 3\sin(t)\cos(20t), 0)}{\sqrt{9\sin^2 t + 4\cos^2 t}} + (0,0,\rho\sin(20t))$$

$$\therefore \boldsymbol{r}(t) = \left(3\cos t + \frac{2\rho\cos t\cos 20t}{\sqrt{9\sin^2 t + 4\cos t}}, 2\sin t + \frac{3\rho\sin t\cos 20t}{\sqrt{9\sin^2 t + 4\cos^2 t}}, \rho\sin(20t)\right)$$

c) Por último, para graficar la trayectoria de la tierra simplemente se tomó $\boldsymbol{r_0}(t)$ que parametriza una elipse tridimensional y se usó `plot3`. Se graficaron los ejes coordenados para mayor claridad. El resultado se muestra en la Fig. 10.
Para el movimiento de la luna se decidió realizar todos los pasos de los incisos anteriores, pero utilizando solamente los valores numéricos de la trayectoria de la tierra, de su velocidad, tangente unitaria, normal, etc. Hacerlo de esta manera es interesante porque se pudo construir la trayectoria de la luna $\boldsymbol{r}(t)$ mediante combinaciones de estas matrices y la suma vectorial del movimiento de la tierra y la luna desde sus propios marcos de referencia, obteniendo una gráfica del movimiento de la luna orbitando la tierra mientras la tierra gira elípticamente. El resultado se muestra en la Fig. 10.
Como último comentario también se generó el vector $\boldsymbol{r}(t)$ según lo obtenido en el inciso anterior utilizando un ciclo `for`, obteniendo la misma trayectoria. Hacerlo de esta forma permite una obtención más rápida y menos propensa a errores.



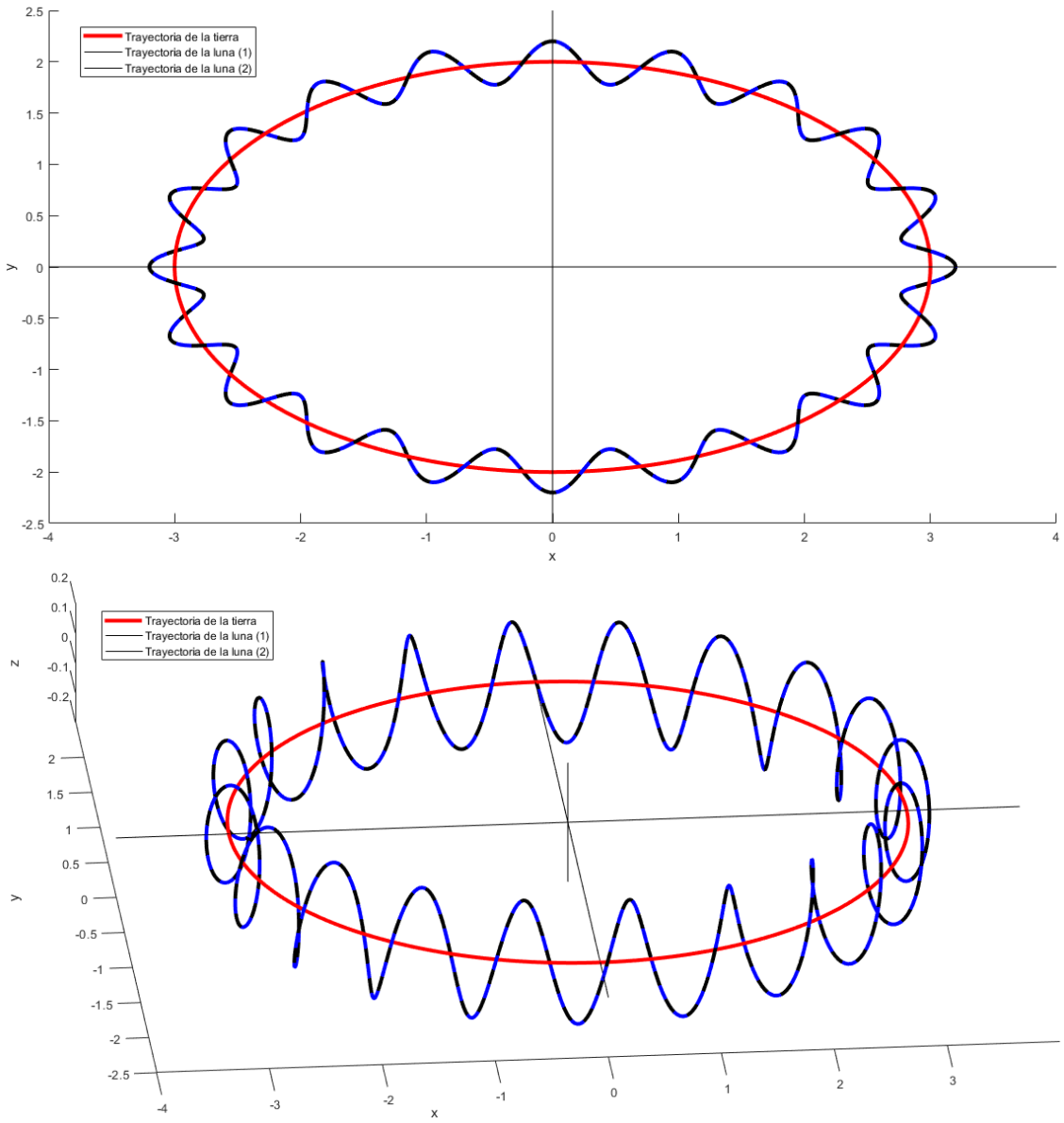


Fig. 10. Trayectorias de la tierra y de la luna.

```
%delta divide el intervalo [0,2pi] en delta intervalos pequeños
delta=1000;
%Creamos el intervalo [0,2pi] con sus divisiones
t=0:2*pi/delta:2*pi;
%Vectores posicion, velocidad y aceleracion de la tierra
r_0=[3*cos(t);2*sin(t);0*t];
v_0=[-3*sin(t);2*cos(t);0*t];
a_0=[-3*cos(t);-2*sin(t);0*t];
%Graficar la orbita de la tierra
plot3(r_0(1,:),r_0(2,:),r_0(3,:),'r','linewidth',3);
%Graficación de ejes coordenados
hold on; axis([-4,4,-2.5,2.5,-0.2,0.2]);
plot3([-4,4],[0,0],[0,0],'k'); xlabel('x');
plot3([0,0],[-2.5,2.5],[0,0],'k'); ylabel('y');
plot3([0,0],[0,0],[-0.2,0.2],'k'); zlabel('z');

%Vectores tangente, normal y binormal
rho=0.2;
tangente=v_0./sqrt(9.*sin(t).*sin(t)+4.*cos(t).*cos(t));
normal=[-tangente(2,:);tangente(1,:);0*t];
binormal=[0*t;0*t;ones(1,delta+1)];
normal_neg=-normal;

%Posición de la luna con respecto al sistema [T,N,B] de la tierra
r_1=rho.*binormal.*sin(20.*t)+rho.*normal_neg.*cos(20.*t);
%Posición de la luna con respecto al marco de referencia de la tierra
r=r_0+r_1;
%Graficar la orbita de la luna alrededor de la tierra mientras gira
plot3(r(1,:),r(2,:),r(3,:),'b','linewidth',3);

%%%OPCIONAL. Uso del resultado del inciso c)
%%%Generación desde cero del movimiento de la luna, r(t)

r_x=zeros(1,delta+1);
r_y=zeros(1,delta+1);
r_z=zeros(1,delta+1);
for i=1:delta+1
    n=t(i);
    raiz=sqrt(9*sin(n)*sin(n)+4*cos(n)*cos(n));
    %Obtención de las coordenadas de la luna para cada t
    r_x(i)=3*cos(n)+(2*rho*cos(n)*cos(20*n))/raiz;
    r_y(i)=2*sin(n)+(3*rho*sin(n)*cos(20*n))/raiz;
    r_z(i)=rho*sin(20*n);
end
%Graficar la orbita de la luna alrededor de la tierra mientras gira
plot3(r_x,r_y,r_z,'--g','linewidth',3);
legend('Trayectoria de la tierra', 'Trayectoria de la luna (1)',...
    'Trayectoria de la luna (2)');
```

Métodos numéricos
Hw02

Diego Emilio Dominguez Tableros