NATALIA CRESPO BRAVO
Grado en Física
USC

March 16, 2024

ATRÉVETE — FOLLA 4

Enunciado

Figura 1: cruz celta que se atopa na igrexa de Santa Susana, na alameda de Santiago de Compostela

Demostra (debuxándoa) que se pode parametrizar coa seguinte traxectoria:

$$x(t) = 21cos(t) + 11.5cos(3t) - 11.5cos(9t) \tag{1-1}$$

$$y(t) = 21sin(t) - 11.5sin(3t) - 11.5sin(9t) \tag{1-2}$$

1. Que rango debe de tomar t para debuxar a traxectoria completa?
2. maxina unha formiga que percorra a traxectoria, indica o camiño que percorrería, en todo o rango de t.
3. Canta distancia percorrería en total? Asume que a traxectoria está definida en cm.
4. Se t é o tempo en s, canto tempo tardaría en percorrer a traxectoria completa?
5. En que momentos a formiga estaría paralela ao chan, e en cales perpendicular a este?

Solución

***O código en Python das representacións está ao final do pdf.

1. Apartado 1: Primeiro tentei de achar a expresión analítica da función coas súas compoñentes paramétricas, pero ista é moi complicada de achar.

$$x(t) = 21\cos(t) + 11.5\cos(3t) - 11.5\cos(9t)$$
$$y(t) = 21\,\mathrm{sen}(t) - 11.5\,\mathrm{sen}(3t) - 11.5\,\mathrm{sen}(9t)$$

$$\boxed{\mathrm{sen}^2 t + \cos^2 t = 1}$$

$$x^2+y^2 = \left[21\cos(t) + 11.5[\cos(3t) - \cos(9t)]\right]^2 + \left[21\,\mathrm{sen}(t) - 11.5(\mathrm{sen}(3t)+\mathrm{sen}(9t))\right]^2$$

$$x^2+y^2 = 21^2\cos^2(t) + 21^2\,\mathrm{sen}^2(t) + 11.5^2[\cos(3t)-\cos(9t)]^2 + 11.5^2[\mathrm{sen}(3t)+\mathrm{sen}(9t)]^2 + 2\cdot 21\cdot 11.5\cos(t)(\cos(3t)-\cos(9t)) - 2\cdot 21\cdot 11.5\,\mathrm{sen}(t)(\mathrm{sen}(3t)+\mathrm{sen}(9t))$$

$$x^2+y^2 = 21^2 + 11.5^2(\cos^2(3t)+\cos^2(9t) - 2\cos(3t)\cos(9t) + \mathrm{sen}^2(3t) + \mathrm{sen}^2(9t) + 2\,\mathrm{sen}(3t)\,\mathrm{sen}(9t)) + 2\cdot 21\cdot 11.5[\cos(t)(\cos(3t)-\cos(9t)) - \mathrm{sen}(t)(\mathrm{sen}(3t)+\mathrm{sen}(9t))]$$

$$x^2+y^2 = 21^2 + 11.5^2(1+1+2(\underbrace{\mathrm{sen}(3t)\,\mathrm{sen}(9t) - \cos(3t)\cos(9t)}_{-\cos(3t+9t)})) + 2\cdot 21\cdot 11.5[\underbrace{\cos(t)(\cos(3t)-\cos(9t)) - \mathrm{sen}(t)(\mathrm{sen}(3t)+\mathrm{sen}(9t))}_{\text{(WolframAlpha)}}]$$

$$x^2+y^2 = 21^2 + 11.5^2(2+2(-\cos(12t)) + 2\cdot 21\cdot 11.5[\cos(2t) - \cos(10t)]$$

$$x^2+y^2 = 21^2 + 11.5^2\cdot 2(1-\cos(12t)) + 2\cdot 21\cdot 11.5[\cos(2t)-\cos(10t)]$$

$$\frac{x^2+y^2}{2} = \frac{21^2}{2} + 11.5^2(1-\cos(12t)) + 21\cdot 11.5(\cos(2t)-\cos(10t))$$

$$\frac{x^2+y^2}{2\cdot(11.5)^2} = \frac{21^2}{2\cdot(11.5)^2} + (1-\cos(12t)) + \frac{21}{11.5}(\cos(2t)-\cos(10t))$$

$$\frac{x^2+y^2}{2(11.5)^2} - (1-\cos(12t)) - \frac{21}{11.5}(\cos(2t)-\cos(10t)) = \frac{21^2}{2\cdot(11.5)^2}$$

Intento de ecuación de la curva sin terminar de despejar t

Figura 2: Intento de ecuación de la curva

Como non conseguín facelo ben, usei Python para facer unha representación da curva e comparala coa figura da foto.
Expresión coa que vou facer as representacións e resolucións:

$$\frac{x^2+y^2}{2*(11.5)^2} - (1-cos(12t)) - \frac{21}{11.5}(cos(2t) - cos(10t)) = \frac{21^2}{2*(11.5)^2} \tag{1-3}$$

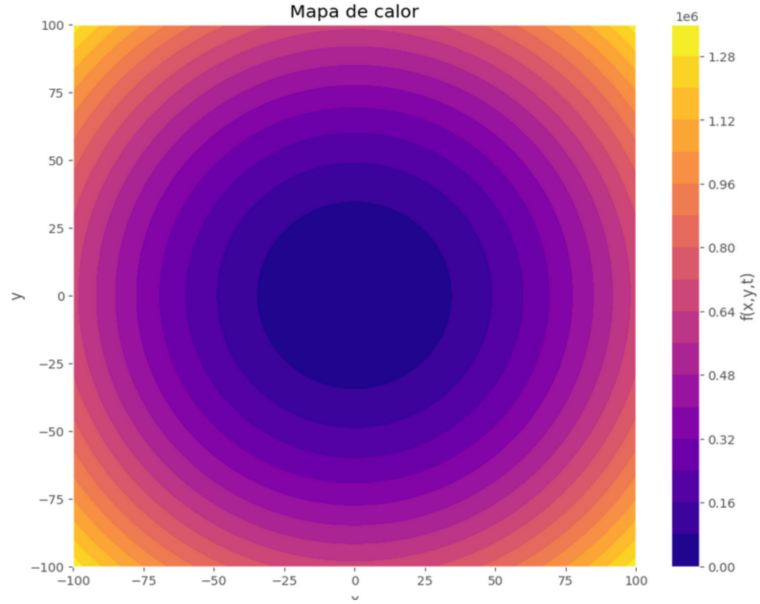

Figura 3: Mapa de calor de la función

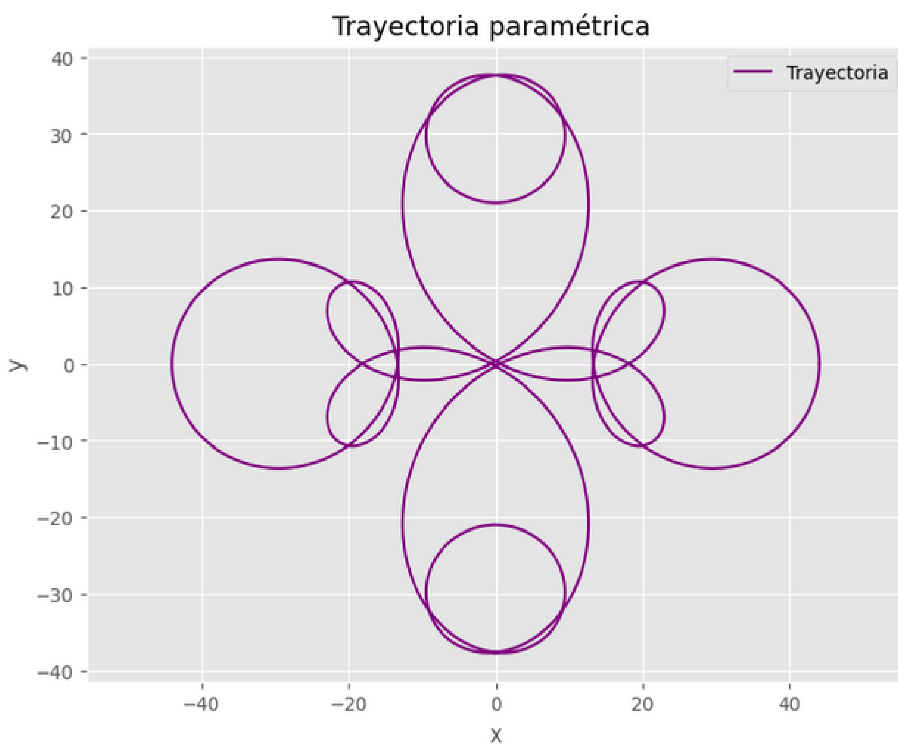

Figura 4: Representación da traxectoria paramétrica da función, onde podemos ver que a parametrización se corresponde coa da foto do enunciado

O rango de valores de t para obter a gráfica completa, ya que a función está composta de seos e cosenos periódicos e contida nunha circunferencia(mellor, nunha elipse(?) $\longrightarrow a(semieixomaior) = 57,92$**) co centro (0,0) e radio 57.92(cálculos en Python), será:** $t \in [0, 2\pi)$**, é dicir, unha volta completa.**

2. Apartado 2:

   Para determinar o percorrido da hormiga necesitamos determinar cal será o sentido positivo do movemento que describirá, que vén dado pola traxectoria da función.
   Gradiente da función: Non ten sentido facelo a man porque non temos a ecuación da curva en función de x e y, pero daríanos a dirección na que o campo varía mais rápidamente, que son as frechas moradas da representación de mais abaixo.

   Dando valores a función:

   Para as equis:

$$x(0) = 21cos(0) + 11.5cos(0) - 11.5cos(0) = 21 \tag{1-4}$$

$$x(\frac{\pi}{2}) = 21cos(\frac{\pi}{2}) + 11.5(cos(\frac{3\pi}{2})) - 11.5(cos(\frac{9\pi}{2})) = 0 \tag{1-5}$$

$$x(\pi) = 21cos(\pi) + 11.5cos(3\pi) - 11.5cos(9\pi) = -21 \tag{1-6}$$

   Para o eixo y:

$$y(0) = 21sen(0) - 11.5sen(0) - 11.5sen(0) = 0 \tag{1-7}$$

$$y(\frac{\pi}{2}) = 21sen(\frac{\pi}{2}) - 11.5(sen(\frac{3\pi}{2})) - 11.5(sen(\frac{9\pi}{2})) = 21 + 11.5 - 11.5 = 21 \tag{1-8}$$

$$y(\pi) = 21sen(\pi) - 11.5sen(3\pi) - 11.5sen(9\pi) = 21 \tag{1-9}$$

   ... así ata completar a volta completa. Como t representa o tempo, supoñemos que nos movemos en el sentido do tempo crecente, é dicir, antihorario, porque o tempo decrecente non ten moito sentido físico.

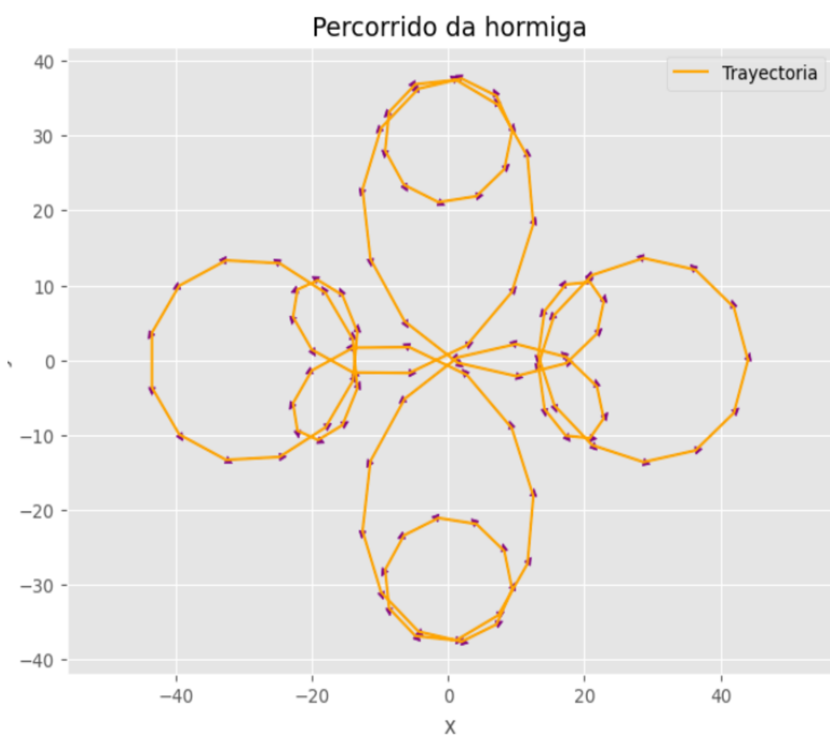

Figura 5: As frechas moradas representan o gradiente da función, que é a pendiente da recta tanxente nesos puntos.

3. Apartado 3: Distancia total percorrida Para calcular a distancia total que percorre a hormiga ao longo da curva, temos que achar a lonxitude do arco da curva (en cm).

   a) derivadas das compoñentes:

$$\frac{d(x(t))}{dt} = -21sen(t) - 11.5 * 3sen(3t) + 11.5 * 9sen(9t) \tag{1-10}$$

$$\frac{d(y(t))}{dt} = 21cos(t) - 11.5 * 3cos(3t) - 11.5 * 9cos(9t) \tag{1-11}$$

   b) Expresión da lonxitude da curva por definición entre 0 y 2pi, que ven dada pola definición de distancia euclídea entre os puntos que escollemos da curva:

$$L = \int_0^{2\pi} \sqrt{(\frac{d(x(t))}{dt})^2 + (\frac{d(y(t))}{dt})^2} \mathrm{d}t \tag{1-12}$$

   noso caso sería:

$$\frac{d(x(t))}{dt} = -21\,sen(t) - 11.5 \cdot 3\,sen(3t) + 11.5 \cdot 9\,sen(9t)$$

$$\frac{d(y(t))}{dt} = 21\,cos(t) - 11.5 \cdot 3\,cos(3t) - 11.5 \cdot 9\,cos(9t)$$

$$L = \int_0^{2\pi} \sqrt{\left(\frac{d(x(t))}{dt}\right)^2 + \left(\frac{d(y(t))}{dt}\right)^2}\, dt = \int_0^{2\pi} \sqrt{\left(-21sen(t) + 11.5(9sen(9t) - 3sen(3t))\right)^2 + \left(21cos(t) - 11.5(3cos(3t) + 9cos(9t))\right)^2}\, dt$$

Figura 6: Cálculo da lonxitude da curva a man

   Non é nada fácil achar a solución analítica desta integral e por iso imos facela co Python:

```
####Lonxitude da curva

#genero valores de t para 0-2pi
t_values = np.linspace(0, 2*np.pi, 1000000)#pongo muchos puntos para reducir el error
#almaceno los valores de las x e y para cada t en listas:
x_values = []
y_values = []

for t_val in t_values:
    x_val = x(t_val)
    y_val = y(t_val)
    x_values.append(x_val)
    y_values.append(y_val)

#calculo la longitud de la curva
curva= 0
for i in range(1, len(x_values)):
    dx = x_values[i] - x_values[i-1]
    dy = y_values[i] - y_values[i-1]
    curva+= np.sqrt(dx**2 + dy**2)

print("La longitud de la curva es:", curva)
```

Figura 7: Código en Python para el cálculo de la longitud de la curva, **LONGITUD= 675.4170624331058 cm ≈ 675.42 cm**

4. Apartado 4: Tempo que tarda a hormiga en percorrer os 675.42 cm Para ter completada a traxectoria, t debe tomar valores dende 0 ata 2π e, tendo en conta que o tempo non pode decrecer, o momento en que a traxectoria dáse por terminada e cando $t = 2\pi$, e, por tanto, o tempo que tarda a hormiga en facer o percorrido (en segundos) é:

$$t = 2 * \pi \approx 6.2831853071795864 \approx \mathbf{6.28\ segundos} \tag{1-13}$$

5. Apartado 5: Momentos nos que a hormiga está paralela e perpendicular ao chan:

   a) **Paralela:** A hormiga estará paralela ao chan, é dicir, ao eixo x, nos momentos nos que a derivada respecto de y sexa nula.

   $$\frac{d(y(t))}{dt} = 0 \longrightarrow t = \frac{\pi}{2} \tag{1-14}$$

   e

   $$t = \pm\frac{3\pi}{2} \tag{1-15}$$

   b) **Perpendicular:** Da misma maneira, a hormiga estará perpedicular ao chan, cando a derivada respecto do eixe x sexa nula.

   $$\frac{d(x(t))}{dt} = 0 \longrightarrow t = 0 \tag{1-16}$$

   e

   $$t = \pi \tag{1-17}$$

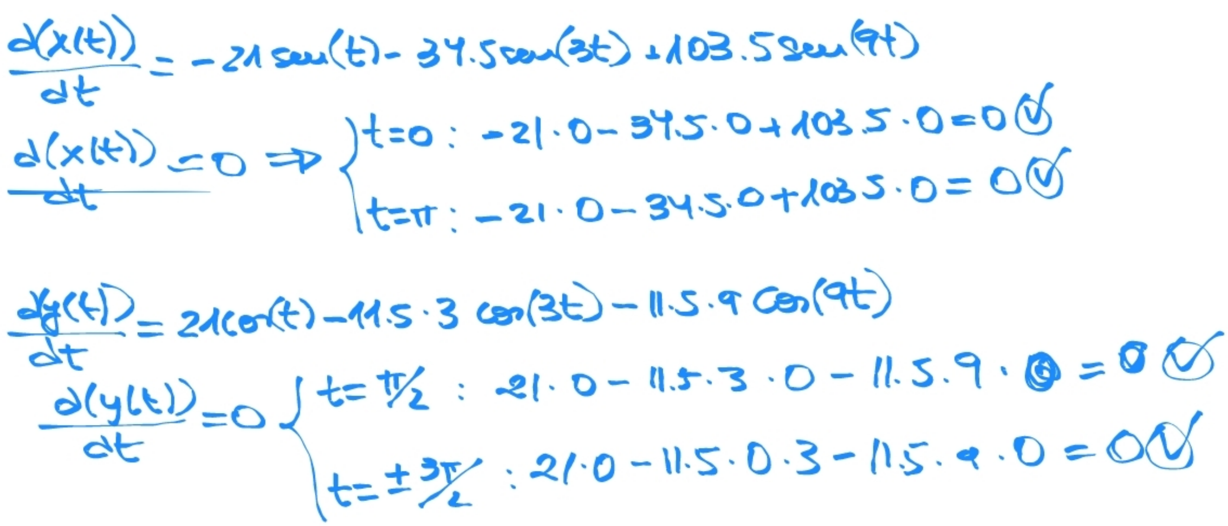

Figura 8: Cálculo de valores de t para los que se anulan las derivadas

1. Apéndice: Código en Python de las representaciones gráficas y cálculos

```
###### Plot de la trayectoria


# Funciones paramétricas:
def x(t):
    x=21 * np.cos(t) + 11.5 * np.cos(3*t) + 11.5 * np.cos(9*t)
    return x

def y(t):
    y=21 * np.sin(t) - 11.5 * np.sin(3*t) - 11.5 * np.sin(9*t)
    return y

# Doy valores a t:
t_values = np.linspace(0, 2*np.pi, 1000)

# Calculo x e y
x_values = x(t_values)
y_values = y(t_values)

# Trazar las funciones x(t) e y(t)
plt.figure(figsize=(8, 6))
plt.plot(x_values, y_values, label='Trayectoria', color='purple')
plt.xlabel('x')
plt.ylabel('y')
plt.title('Trayectoria paramétrica ')
plt.axis('equal')
plt.grid(True)
plt.legend()
plt.show()
```

Figura 9: Código para la representación de la trayectoria paramétrica

```python
# general imports
# general imports
%matplotlib inline
%reload_ext autoreload
%autoreload 2

# numpy and matplotlib
import numpy as np
import matplotlib
import matplotlib.pyplot as plt
from mpl_toolkits.mplot3d import Axes3D
matplotlib.style.use('ggplot')
import graph_utils as gf

figsize = 6, 3.8
cmap    = 'hot'

########## ATREVETE 4

def f(x, y, t):
    funcion = ((x**2 + y**2) / 2*(11.5)**2) - (1- np.cos(12*t)) -(21/11.5)*(np.cos(2*t)-np.cos(10*t))
    funcion += ((21**2)/(2*(11.5)**2))
    return funcion

#valores de x,y e t :
x_values = np.linspace(-100, 100, 100)
y_values = np.linspace(-100, 100, 100)
t_values = np.linspace(0, 2*np.pi, 100)

# matrices de coordenadas:
X, Y, T = np.meshgrid(x_values, y_values, t_values)

# Función:
Z = f(X, Y, T)

#Representación:
plt.figure(figsize=(10, 8))
plt.contourf(X[:,:,0], Y[:,:,0], Z[:,:,0], levels=20, cmap='plasma')
plt.colorbar(label='f(x,y,t)')
plt.xlabel('x')
plt.ylabel('y')
plt.title('Mapa de calor')
plt.grid(True)
plt.show()
```

Figura 10: Código para la representación del mapa de calor

Submitted by NATALIA CRESPO BRAVO on 16 de marzo de 2024.

```python
# Definir funciones x(t) y y(t)
def x(t):
    return 21 * np.cos(t) + 11.5 * np.cos(3*t) + 11.5 * np.cos(9*t)

def y(t):
    return 21 * np.sin(t) - 11.5 * np.sin(3*t) - 11.5 * np.sin(9*t)

# Generar valores de t
t_values = np.linspace(0, 2*np.pi, 100)

# Calcular los valores de x(t) e y(t)
x_values = x(t_values)
y_values = y(t_values)

# Trazar las funciones x(t) e y(t) con flechas que indican la dirección
plt.figure(figsize=(8, 6))
plt.plot(x_values, y_values, color='orange', label='Trayectoria')

# Dibujar flechas que indican la dirección
for i in range(len(x_values)-1, 0, -1):
    dx = x_values[i] - x_values[i-1]
    dy = y_values[i] - y_values[i-1]
    plt.arrow(x_values[i], y_values[i], -dx, -dy, head_width=1, head_length=1, fc='purple', ec='purple')

plt.xlabel('x')
plt.ylabel('y')
plt.title('Percorrido da hormiga')
plt.axis('equal')
plt.grid(True)
plt.legend()
plt.show()
```

Figura 11: Código para la representación del recorrido de la hormiga

```python
#Cálculo del radio de la circunferencia que contiene a la función
# Calculo los valores máximos y mínimos de x e y
x_max = np.max(x_values)
x_min = np.min(x_values)
y_max = np.max(y_values)
y_min = np.min(y_values)

# Calculo la distancia desde el origen hasta el punto más alejado
max_distance = np.sqrt(x_max**2 + y_max**2)
min_distance = np.sqrt(x_min**2 + y_min**2)
radius = max(max_distance, min_distance)

print("El radio de la circunferencia que contiene a la función completa es:", radius)
```

Figura 12: Código para el cálculo del radio de la circunferencia, o para el cálculo del semieje mayor de la elipse