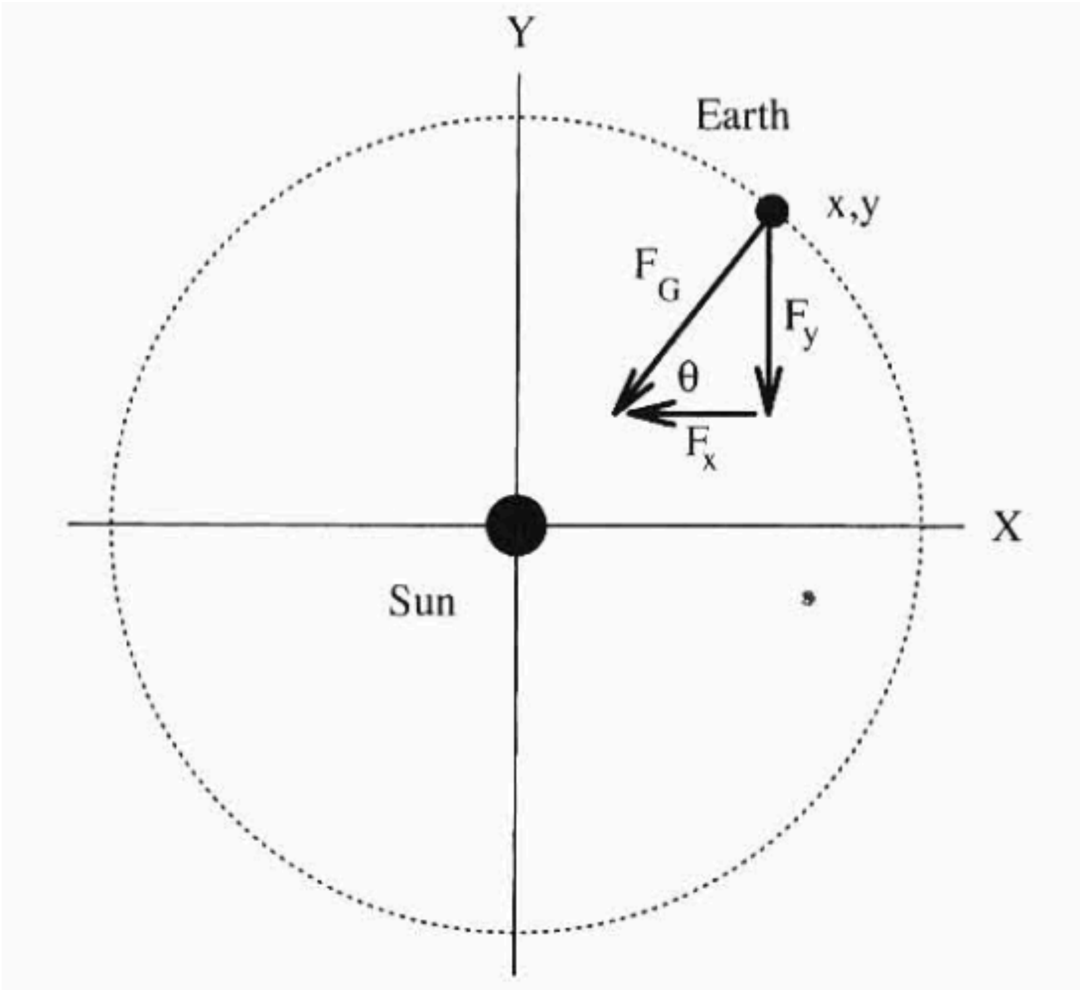
Tema 4: Ecuaciones diferenciales en 2 y 3 dimensiones

4.3 Cálculo de órbitas planetarias



4.3 Cálculo de órbitas planetarias

$$F_G = \frac{GM_S M_T}{r^2}$$

$$F_{G,x} = -\frac{GM_S M_T}{r^2}\cos\vartheta = -\frac{GM_S M_T x}{r^3}$$

$$F_{G,y} = -\frac{GM_S M_T y}{r^3}$$

Ecuaciones del movimiento:

$$\frac{dx}{dt} = v_x \qquad \frac{dy}{dt} = v_y$$

$$\frac{dv_x}{dt} = -\frac{GM_S x}{r^3} \qquad \frac{dv_y}{dt} = -\frac{GM_S y}{r^3}$$

4.3 Cálculo de órbitas planetarias

$$1\ UA \approx 1.5 \cdot 10^{11}\ m\ ; \qquad 1\ \text{año} \approx 3.2 \cdot 10^{7}\ s$$

Órbita circular:

$$F_G = \frac{GM_S M_T}{r^2} = \frac{M_T v^2}{r} \qquad \Rightarrow \qquad GM_S = v^2 r$$

$$v = \frac{2\pi r}{1\ \text{año}} = \frac{2\pi(1\ UA)}{1\ a.} = 2\pi$$

$$GM_s = 4\pi^2 \cdot (1\ UA) = 4\pi^2 \qquad (\text{en } UA \text{ y años})$$

Ecuaciones del movimiento:

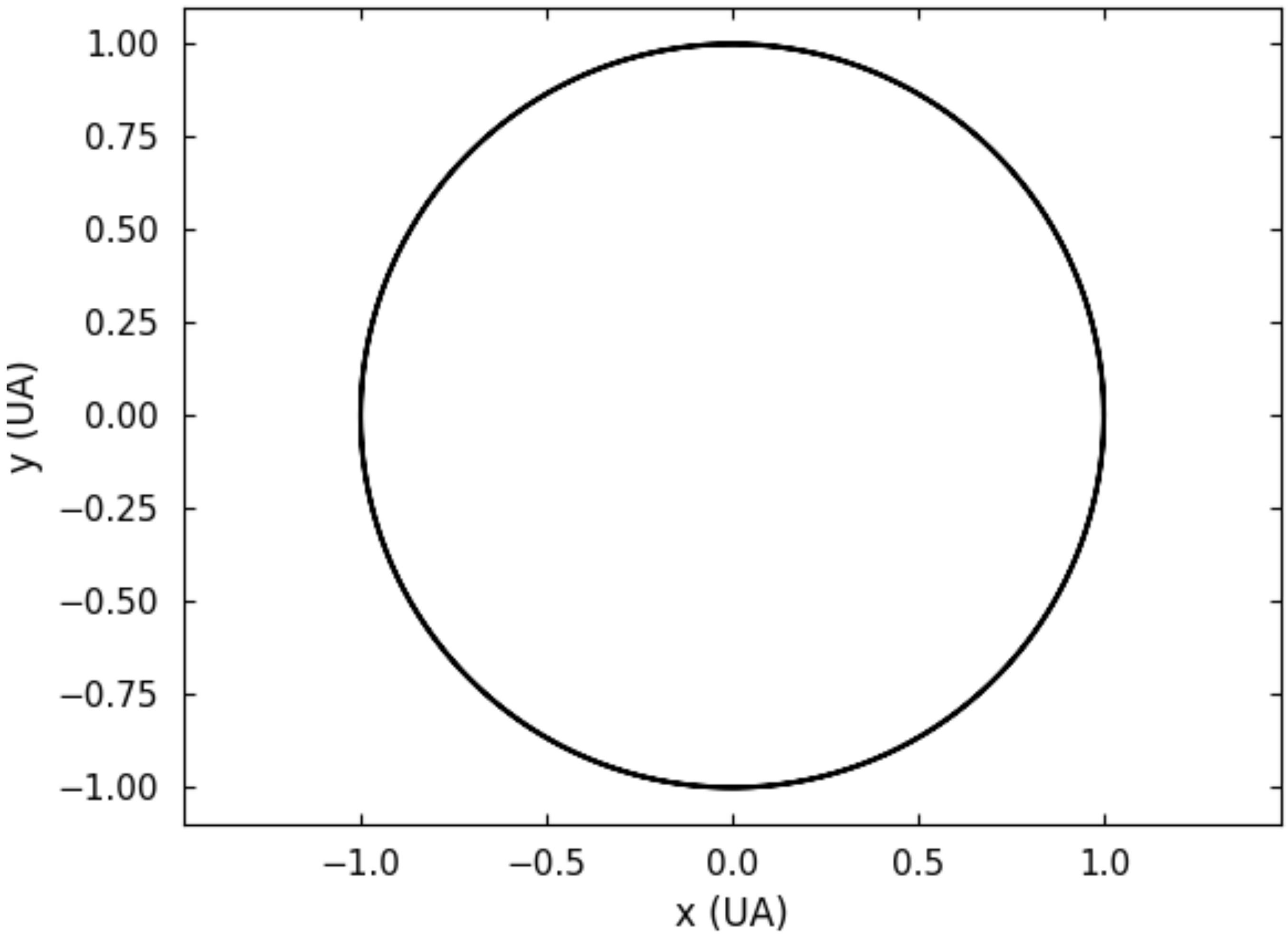
$$\frac{dx}{dt} = v_x \qquad \frac{dy}{dt} = v_y$$

$$\frac{dv_x}{dt} = -\frac{4\pi^2 x}{r^3} \qquad \frac{dv_y}{dt} = -\frac{4\pi^2 y}{r^3}$$

4.3 Cálculo de órbitas planetarias

$x_0 = 1;\ y_0 = 0,\ v_{x0} = 0,\ v_{y0} = 2\pi \qquad (2\pi r/t = 2\pi \cdot 1/1 = 2\pi)$

Órbita circular, v_0=2π

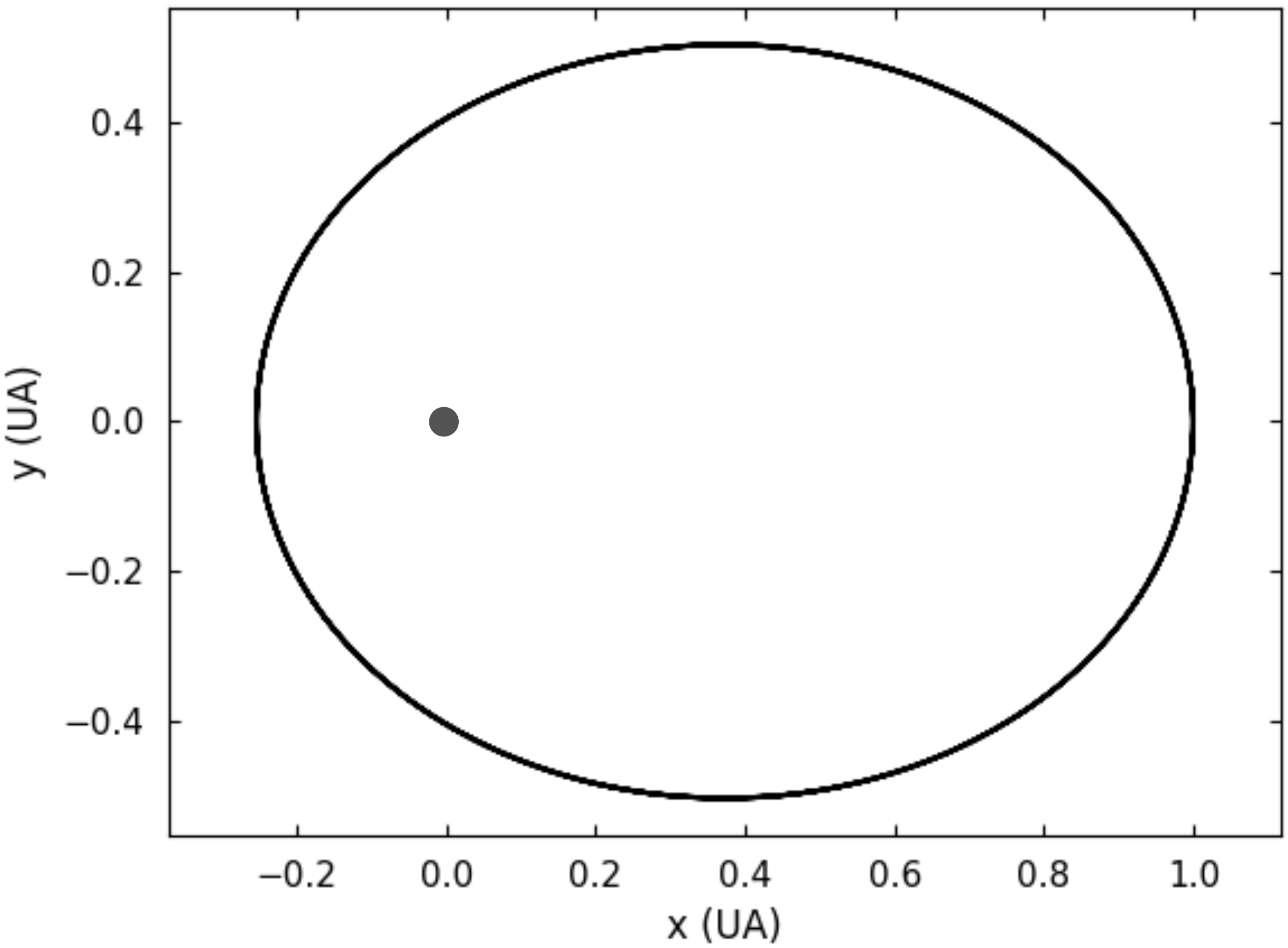


4.3 Cálculo de órbitas planetarias

Órbita circular, $v_0=2\pi$

4.3 Cálculo de órbitas planetarias

Órbita elíptica, v_0=4



4.3 Cálculo de órbitas planetarias

Órbita elíptica, v_0=4

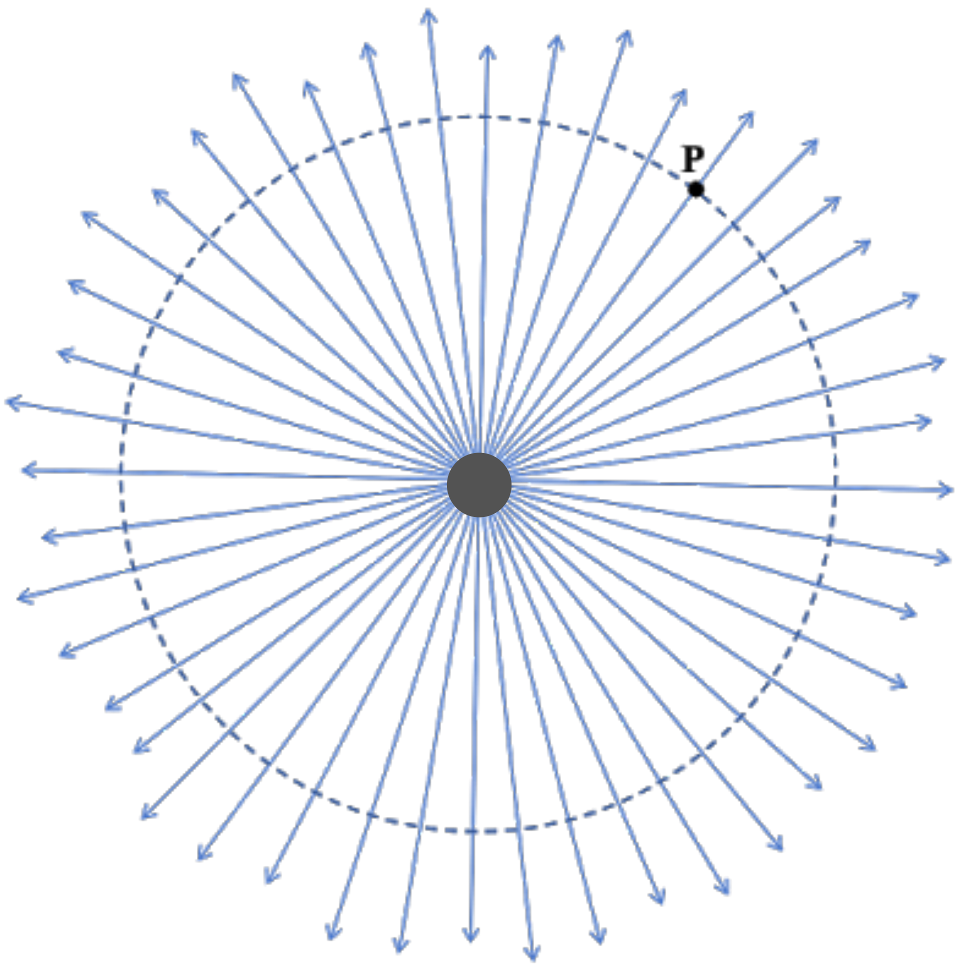
4.3 Cálculo de órbitas planetarias

Utilización de vpython para crear imagen en movimiento

```
import vpython as vp

[...]

# Inicializo parámetros para los gráficos 3D y pinto la posición inicial del Sol
# y la Tierra.

vp.scene.height=640                    # Para hacer la pantalla cuadrada
Sol = vp.sphere(pos=vp.vector(0,0,0), radius=0.1, color=vp.color.yellow)
Tierra = vp.sphere(pos=vp.vector(1,0,0), radius=0.02, color=vp.color.cyan)

# La siguiente línea crea una curva por donde pasa la Tierra.

Tierra.orbita = vp.curve(color=vp.vector(0.3,0.3,0.3))

[...]

# Comienzo el bucle. Dentro del mismo:

         vp.rate(300)                  # Retrasa la ejecución 1/300 de segundo.
         Tierra.pos=vp.vector(x[i],y[i],0)
         Tierra.orbita.append(pos=Tierra.pos)

[...]

# Al hacer la figura con matplotlib, antes del comando show():

# Para que las escalas en x y en y sean las mismas:

ax = gca()                    # Otra opción es sustituir estas dos líneas por
ax.axis("equal")              # una: gca().set_aspect(“equal”): gráfico cuadrado
```

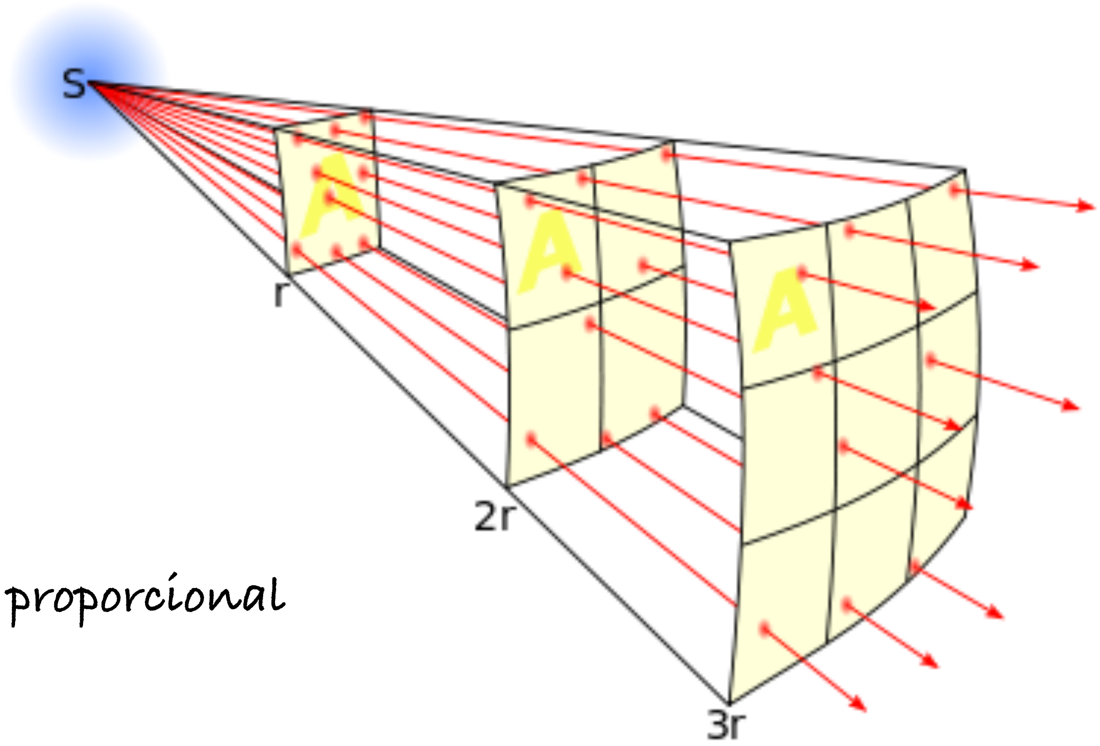
4.3 Cálculo de órbitas planetarias

Líneas de campo



4.3 Cálculo de órbitas planetarias

Líneas de campo



N líneas,
densidad proporcional
a N/r^2

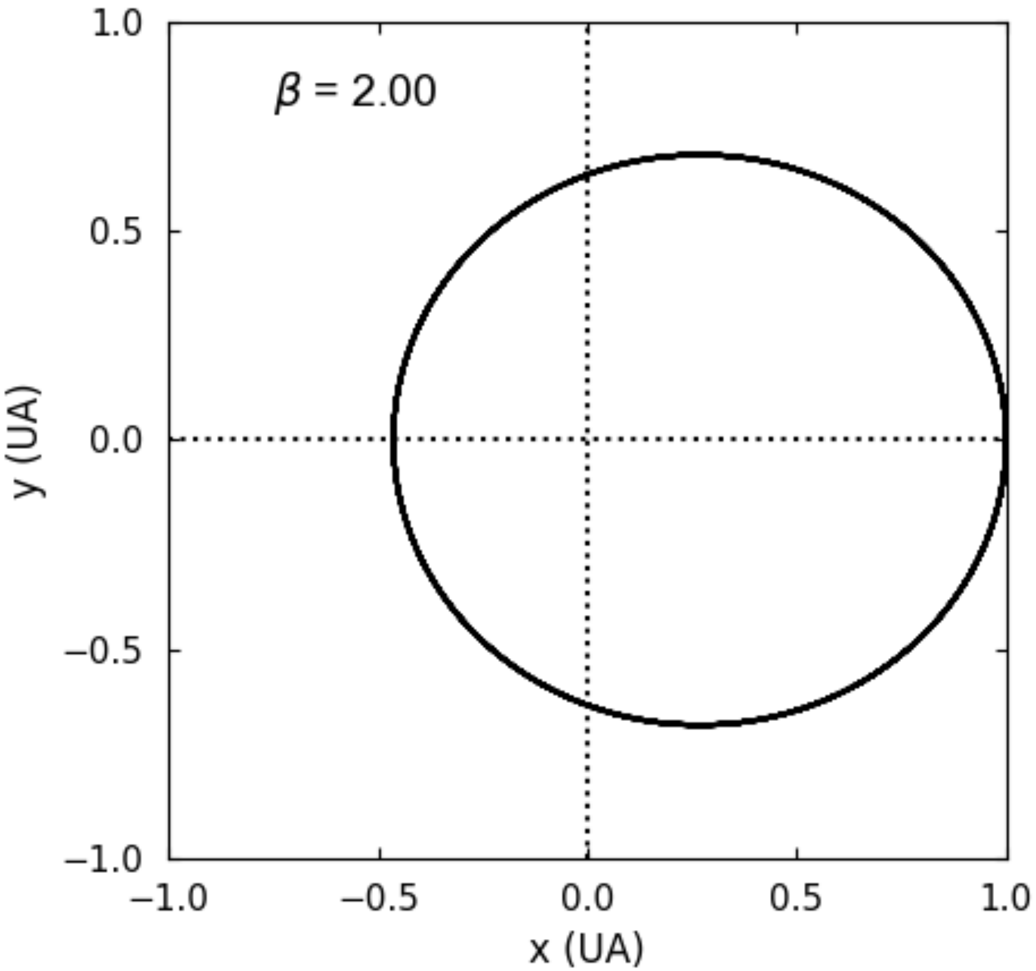
4.3 Cálculo de órbitas planetarias

Variación del exponente de r en la fuerza gravitatoria

$$F_G = \frac{G M_S M_T}{r^{\beta}}$$

4.3 Cálculo de órbitas planetarias

t_f = 2 años, dt = 0.001 años, x_0 = 1 UA, y_0 = 0, v_{x0} = 0, v_{y0} = 5 UA/año, vp.rate(300)

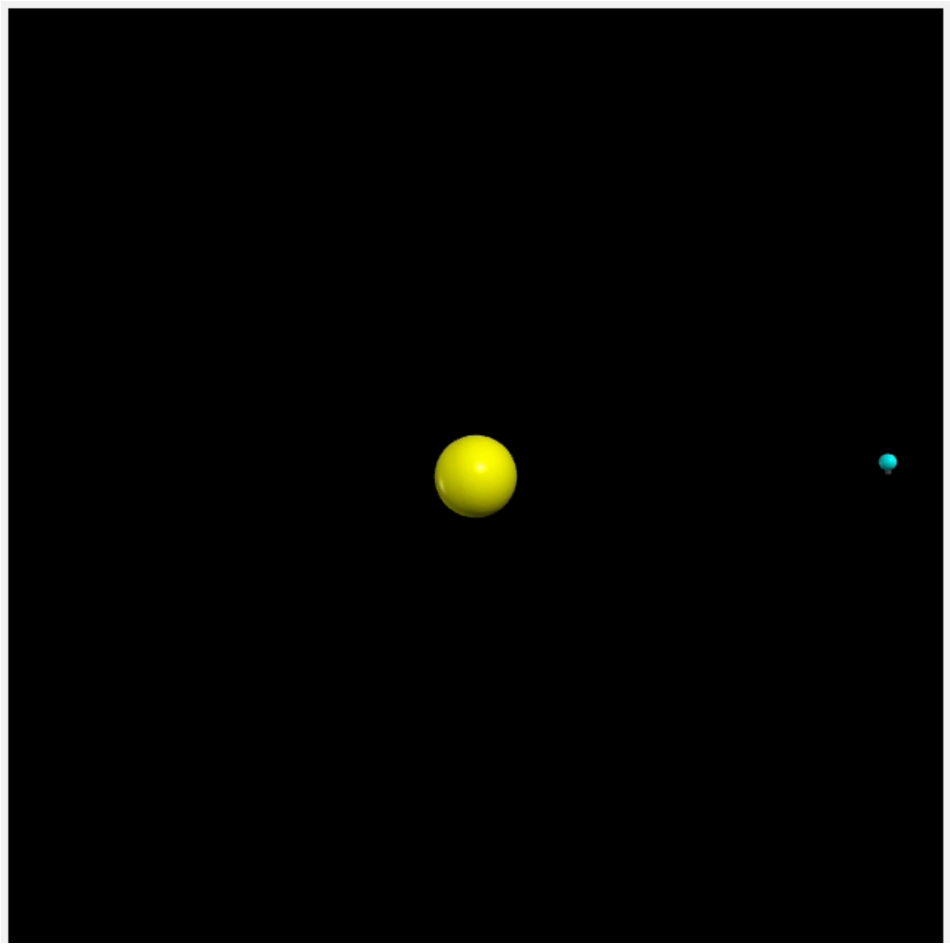


4.3 Cálculo de órbitas planetarias

t_f = 2 años, dt = 0.001 años, x_0 = 1 UA, y_0 = 0, v_{x0} = 0, v_{y0} = 5 UA/año, vpython.rate(300)

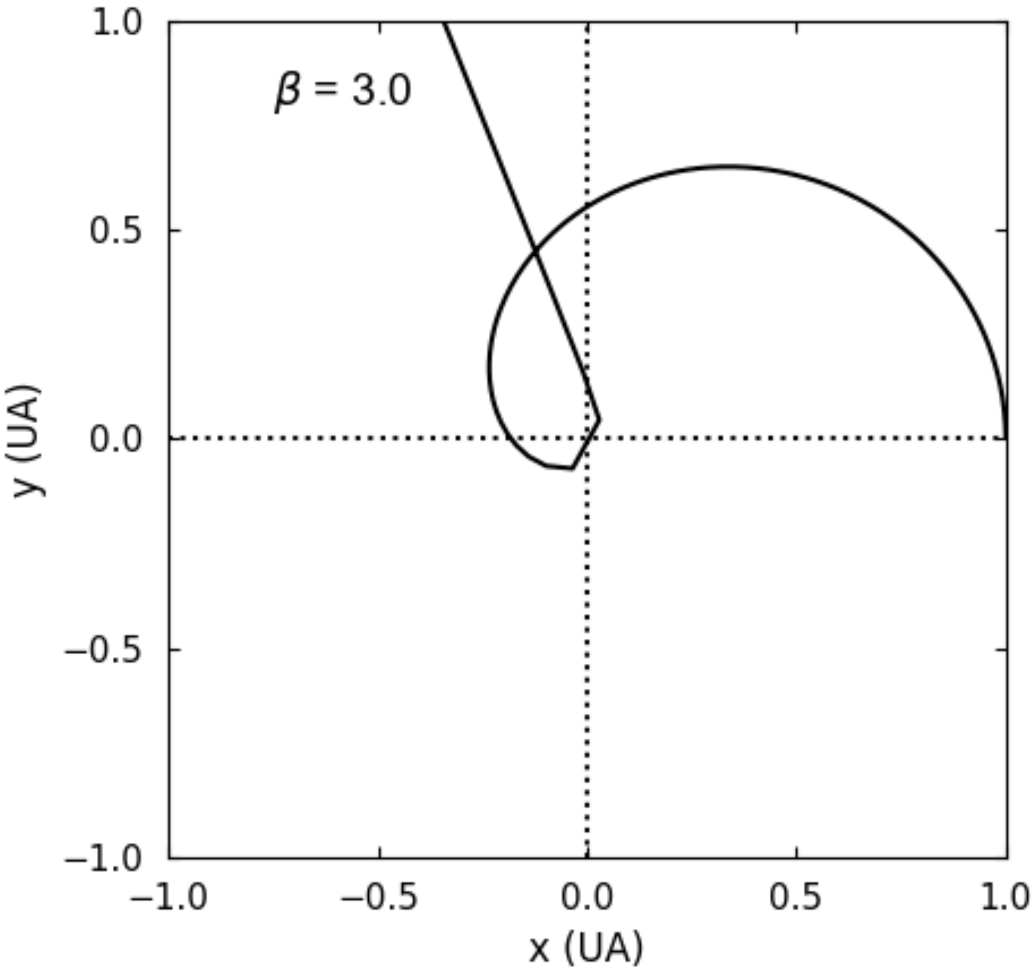
4.3 Cálculo de órbitas planetarias

β = 3.0, vp.rate(100)



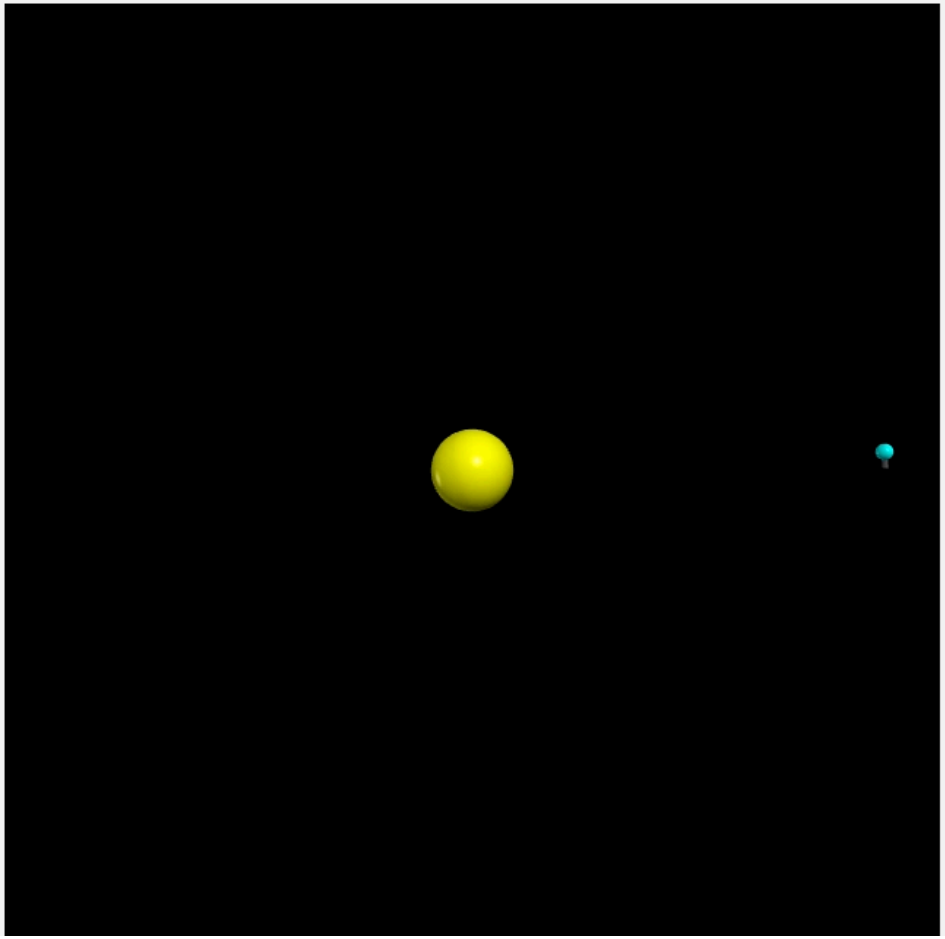
4.3 Cálculo de órbitas planetarias

β = 3.0, vp.rate(100)



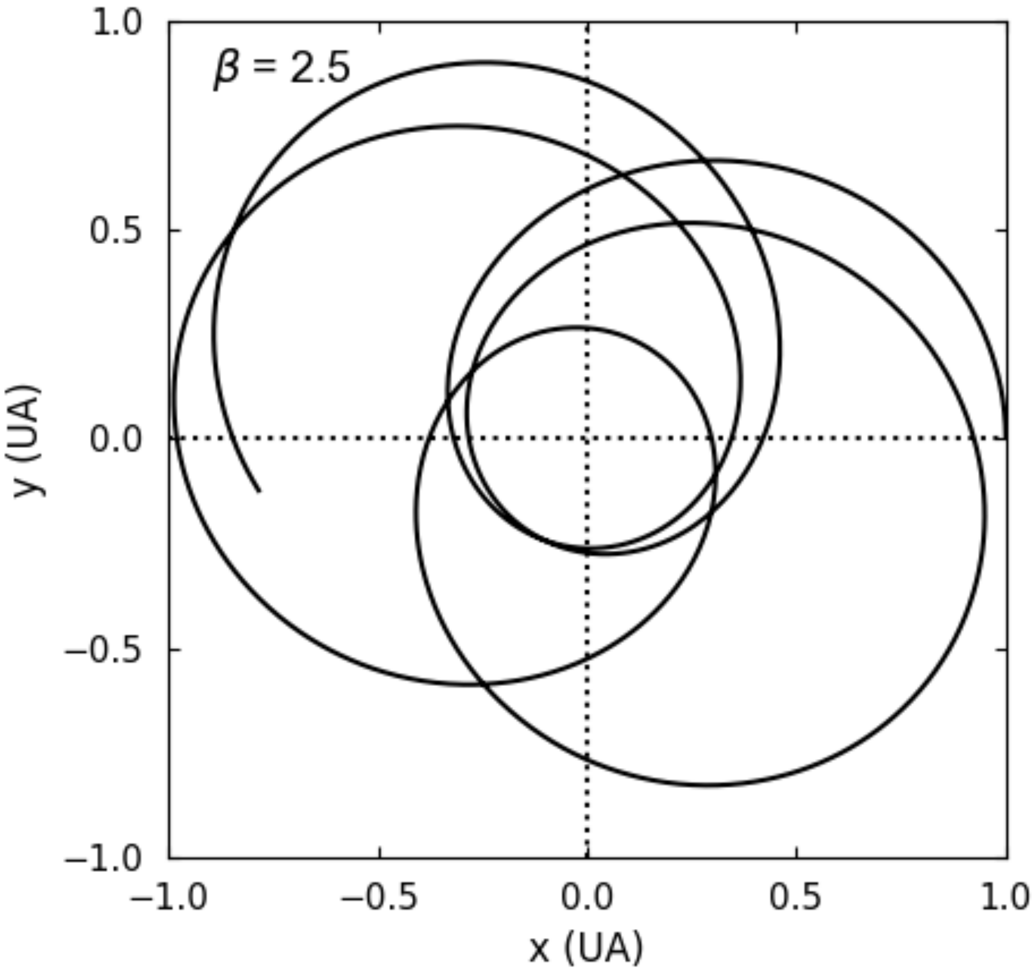
4.3 Cálculo de órbitas planetarias

$\beta = 2.5$



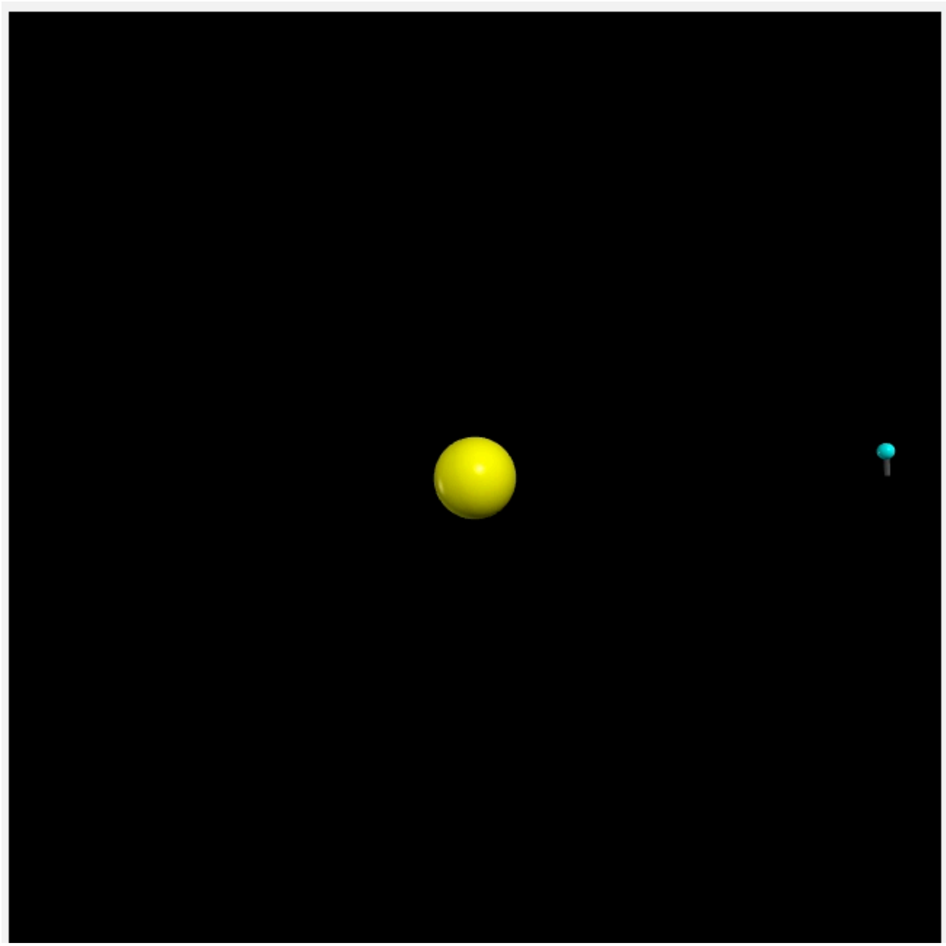
4.3 Cálculo de órbitas planetarias

β = 2.5



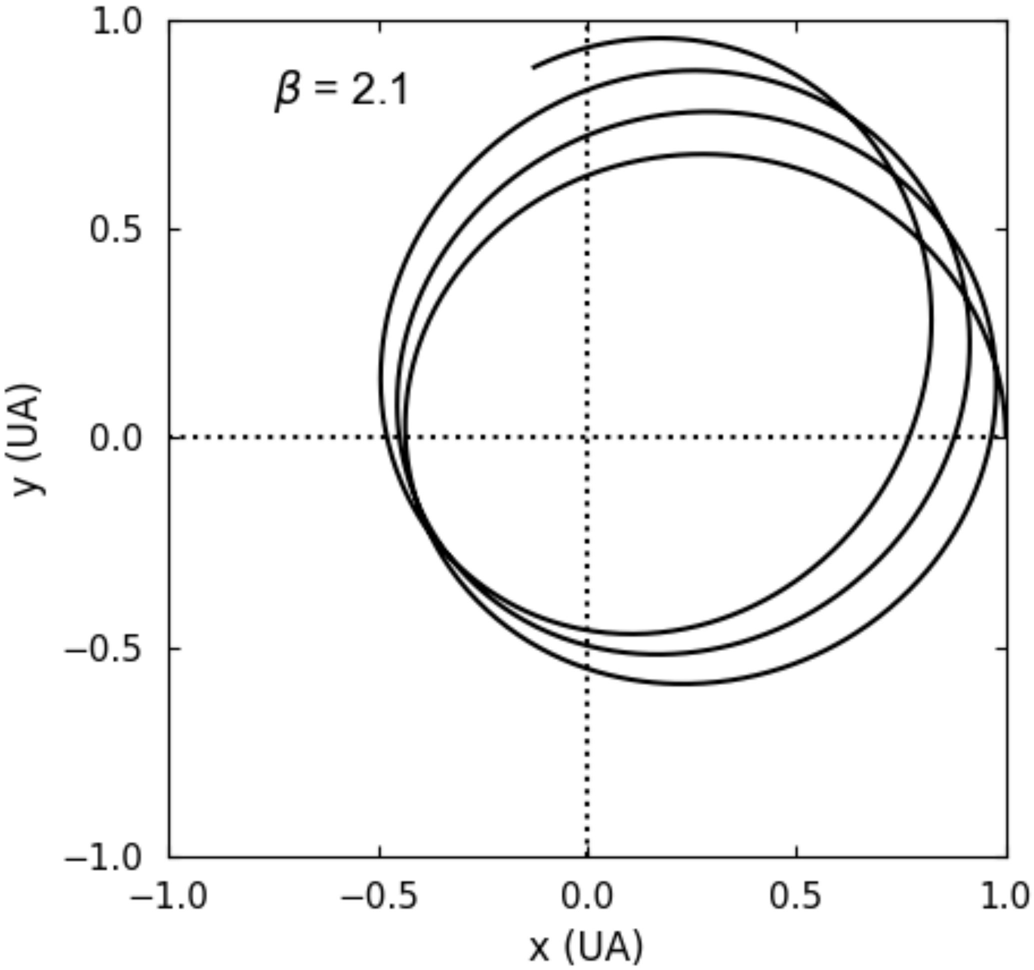
4.3 Cálculo de órbitas planetarias

$\beta = 2.1$



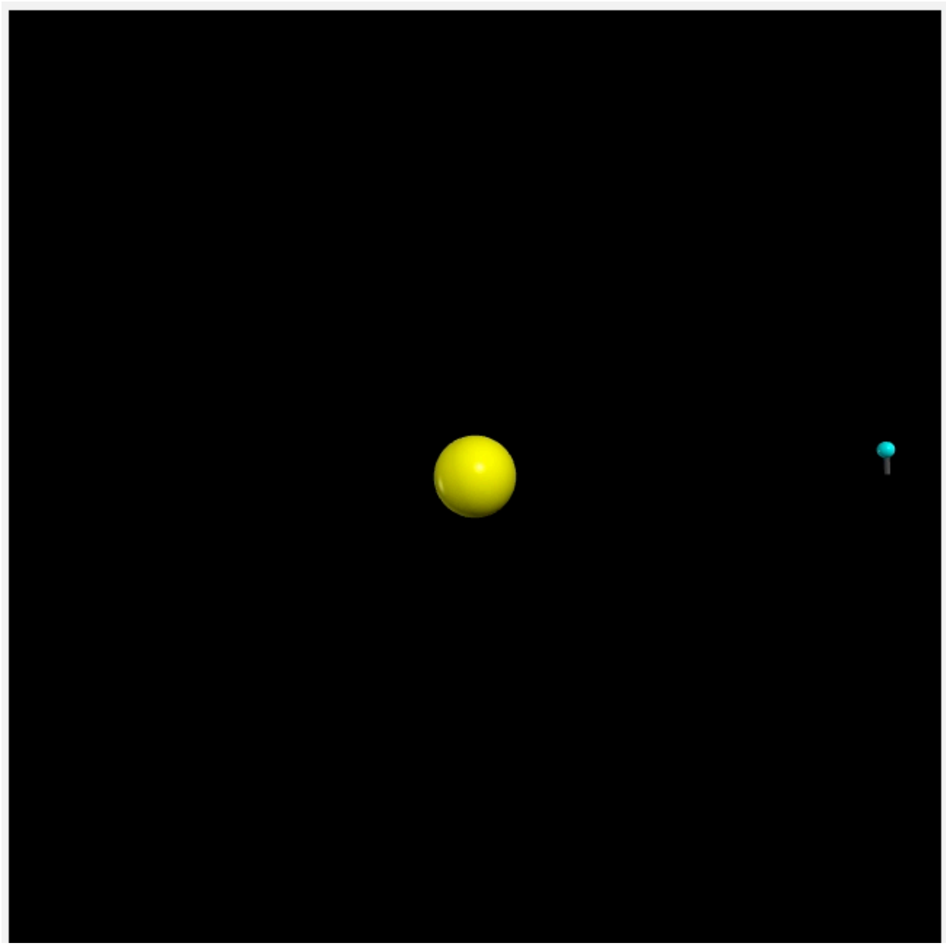
4.3 Cálculo de órbitas planetarias

β = 2.1



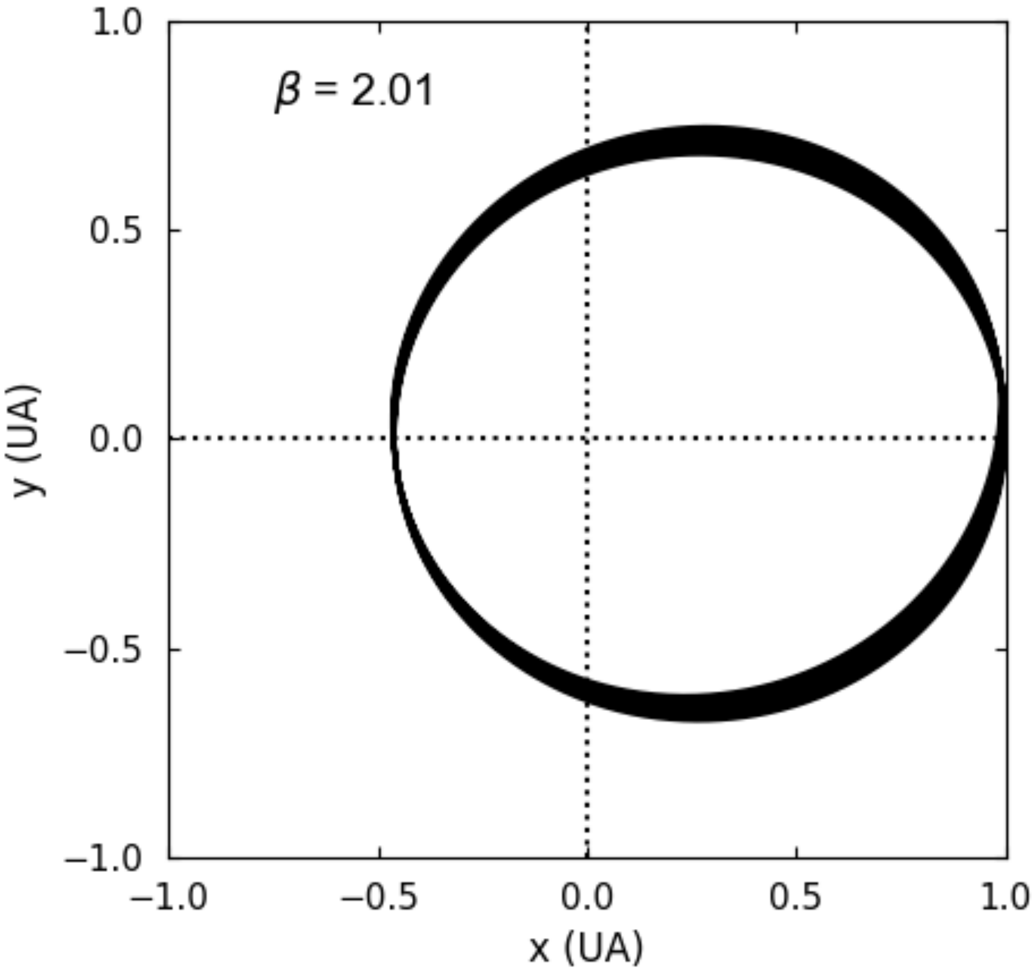
4.3 Cálculo de órbitas planetarias

$\beta = 2.01$, $t_f = 5$, vp.rate(300)



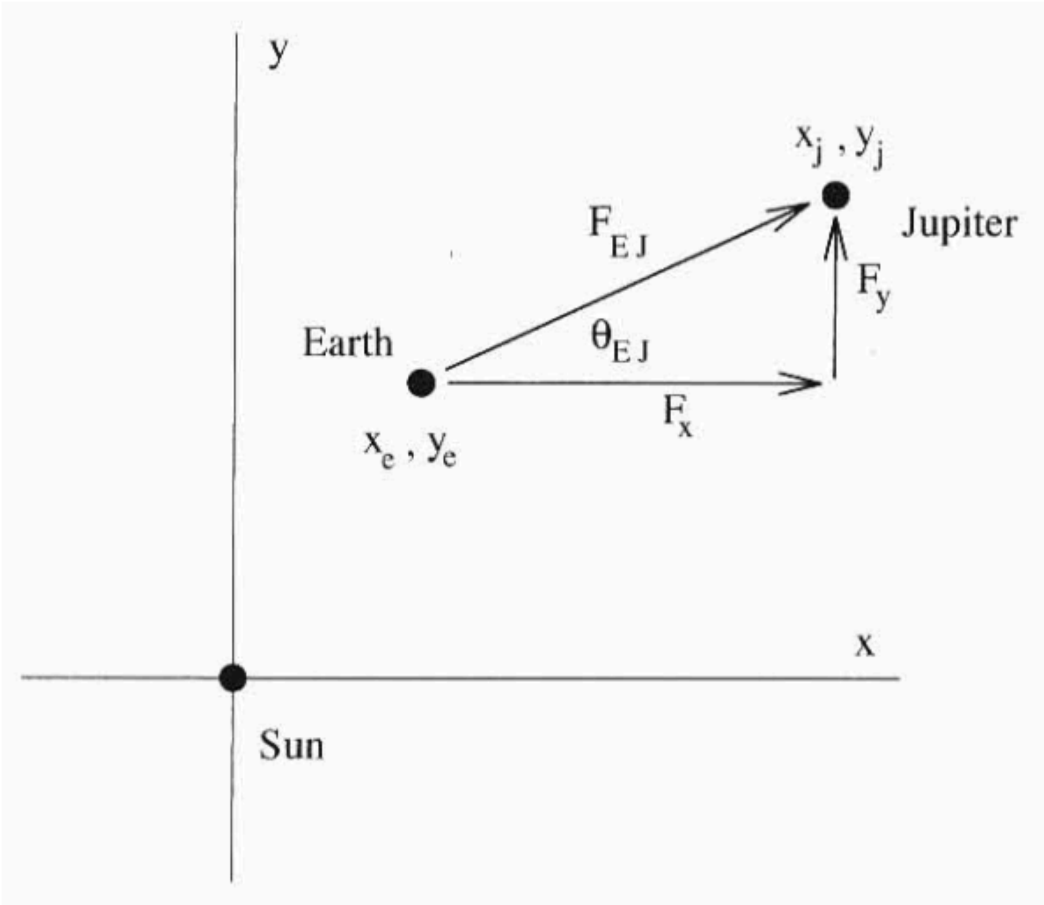
4.3 Cálculo de órbitas planetarias

β = 2.01, t_f = 5, vp.rate(300)



4.3 Cálculo de órbitas planetarias

Problema de tres cuerpos

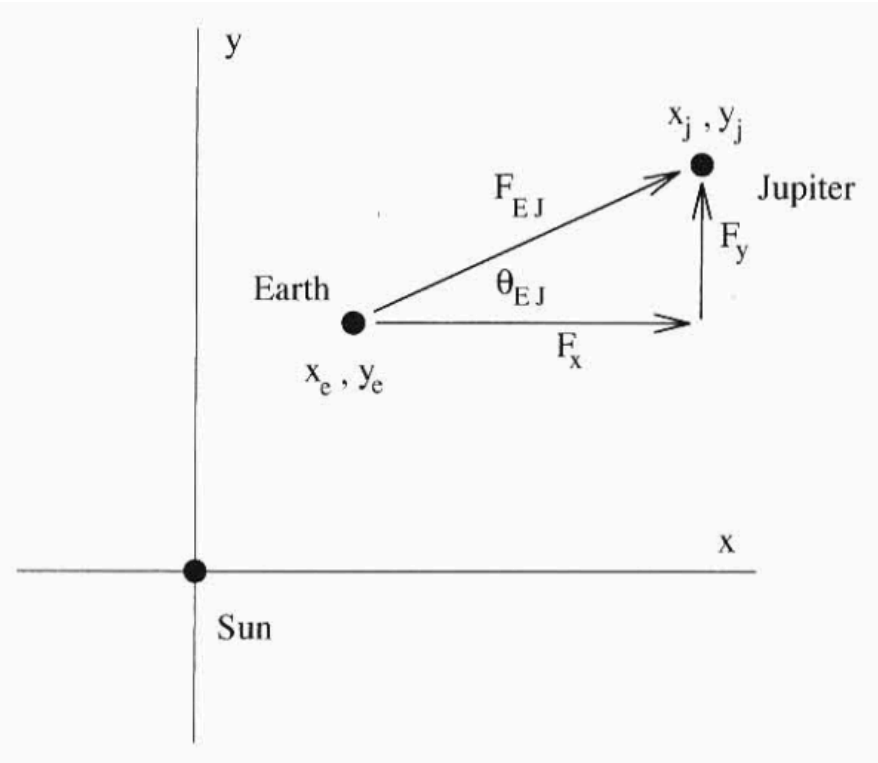


Tema 4: Ecuaciones diferenciales en 2 y 3 dimensiones

4.3 Cálculo de órbitas planetarias

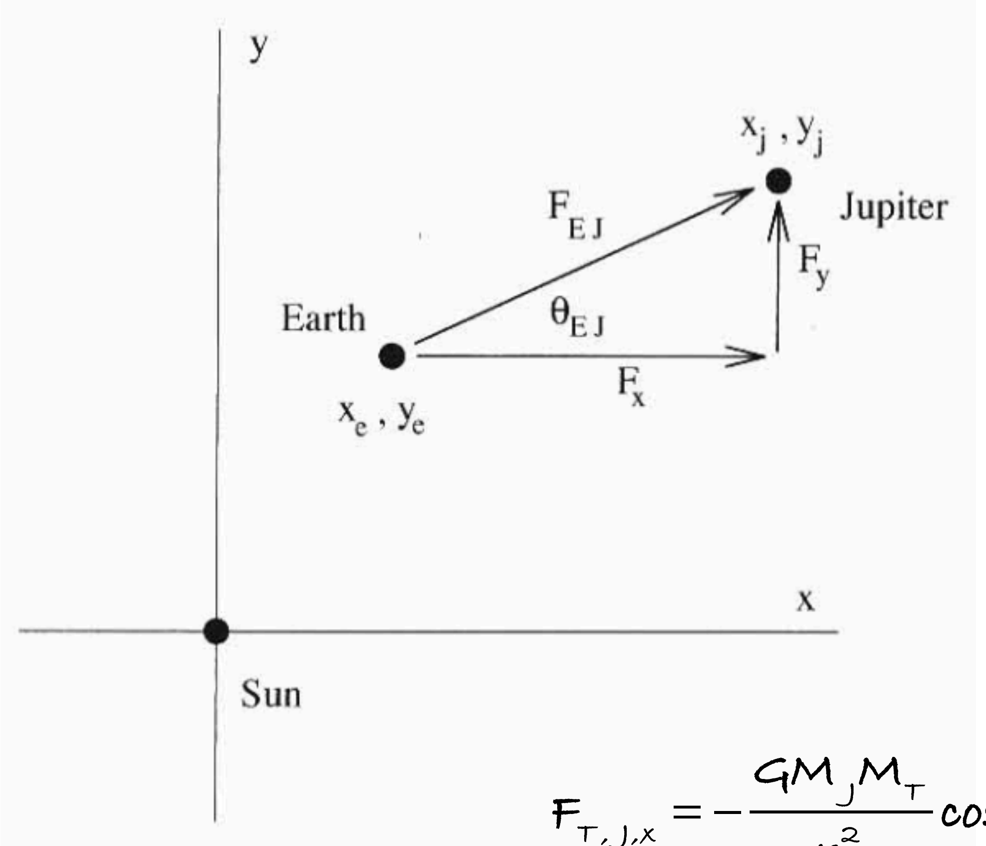
Fuerza ejercida por Júpiter sobre la Tierra:

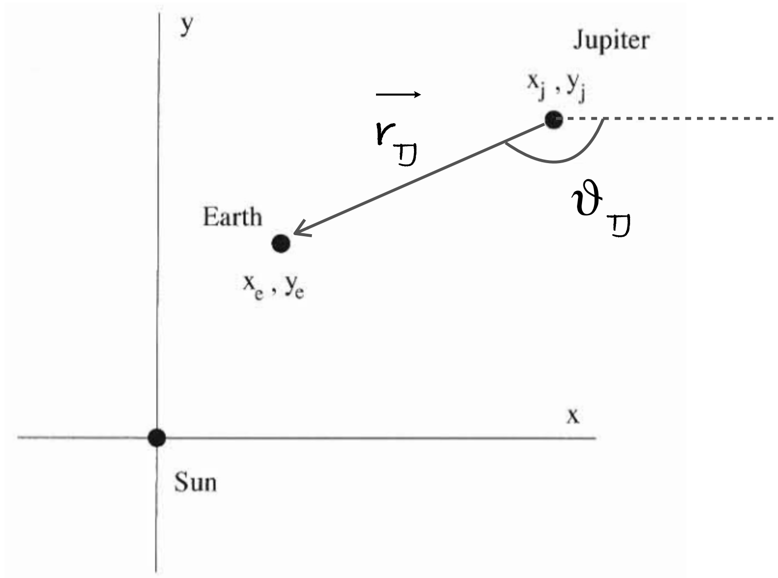
$$F_{T,J} = \frac{GM_J M_T}{r_{TJ}^2}$$



$$F_{T,J,x} = -\frac{GM_J M_T}{r_{TJ}^2} \cos\vartheta_{TJ} = -\frac{GM_J M_T \left(x_T - x_J\right)}{r_{TJ}^3}$$

4.3 Cálculo de órbitas planetarias





$$F_{T,J,x} = -\frac{GM_J M_T}{r_{TJ}^2}\cos\vartheta_{TJ} = -\frac{GM_J M_T\left(x_T - x_J\right)}{r_{TJ}^3}$$

$$F_{T,J,y} = -\frac{GM_J M_T}{r_{TJ}^2}\operatorname{sen}\vartheta_{TJ} = -\frac{GM_J M_T\left(y_T - y_J\right)}{r_{TJ}^3}$$

4.3 Cálculo de órbitas planetarias

$$F_{T,J,x} = -\frac{GM_J M_T}{r_{TJ}^2}\cos\vartheta_{TJ} = -\frac{GM_J M_T\left(x_T - x_J\right)}{r_{TJ}^3}$$

$$F_{T,J,y} = -\frac{GM_J M_T}{r_{TJ}^2}\operatorname{sen}\vartheta_{TJ} = -\frac{GM_J M_T\left(y_T - y_J\right)}{r_{TJ}^3}$$

$$GM_J = GM_S \cdot \left(\frac{M_J}{M_S}\right) = 4\pi^2\left(\frac{M_J}{M_S}\right)$$

$$M_S = 2.0 \cdot 10^{30}\ \text{kg};\quad M_J = 1.9 \cdot 10^{27}$$

$$GM_J = 3.8 \cdot \pi^2 \cdot 10^{-3}$$

4.3 Cálculo de órbitas planetarias

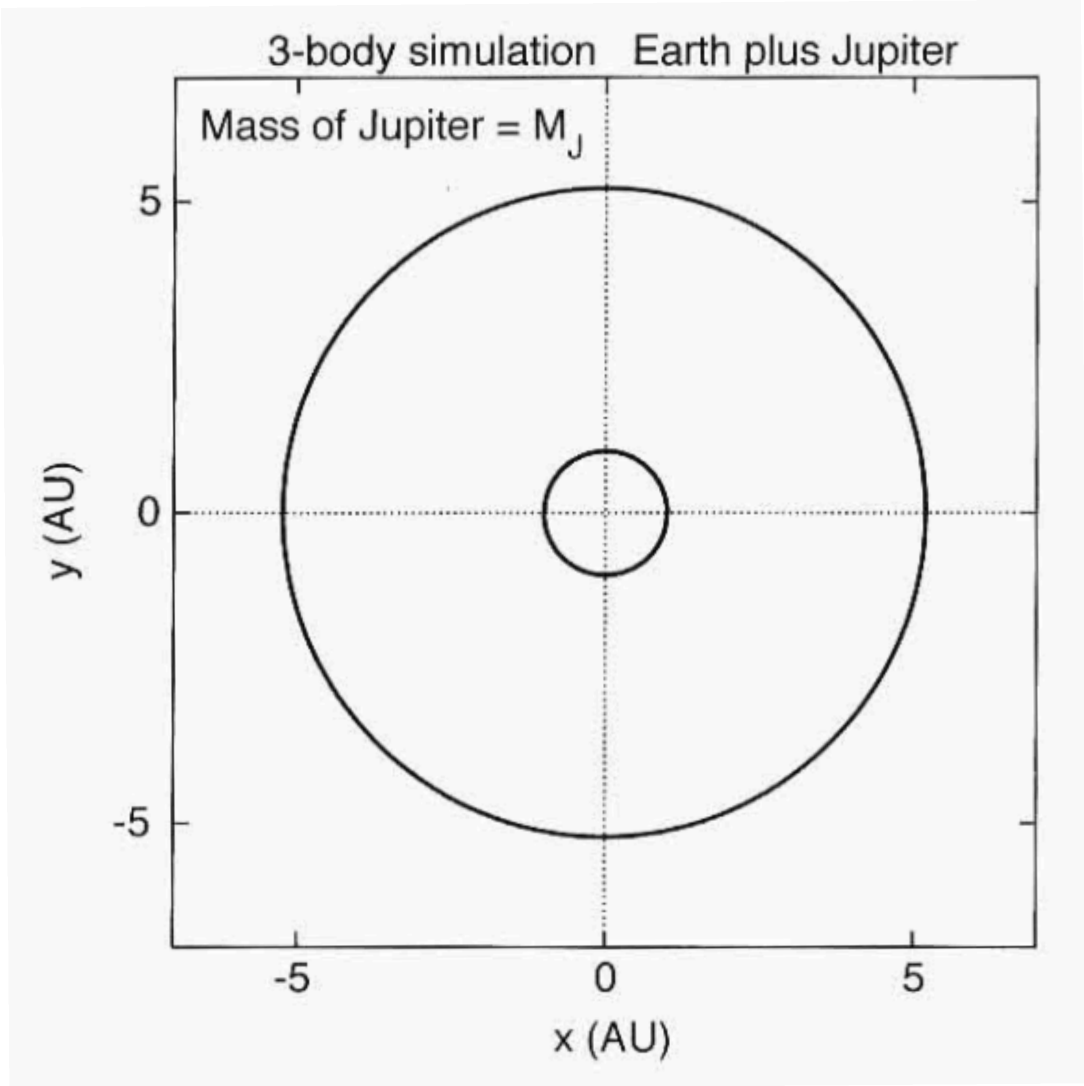
$$\frac{dx_J}{dt} = v_{J,x} \quad ; \quad \frac{dv_{J,x}}{dt} = \frac{-4\pi^2 x_J}{r_J^3} - \frac{3.8 \cdot \pi^2 \cdot 10^{-3}\left(x_J - x_T\right)}{r_{T,J}^3}$$

$$\frac{dy_J}{dt} = v_{J,y} \quad ; \quad \frac{dv_{J,y}}{dt} = \frac{-4\pi^2 y_J}{r_J^3} - \frac{3.8 \cdot \pi^2 \cdot 10^{-3}\left(y_J - y_T\right)}{r_{T,J}^3}$$

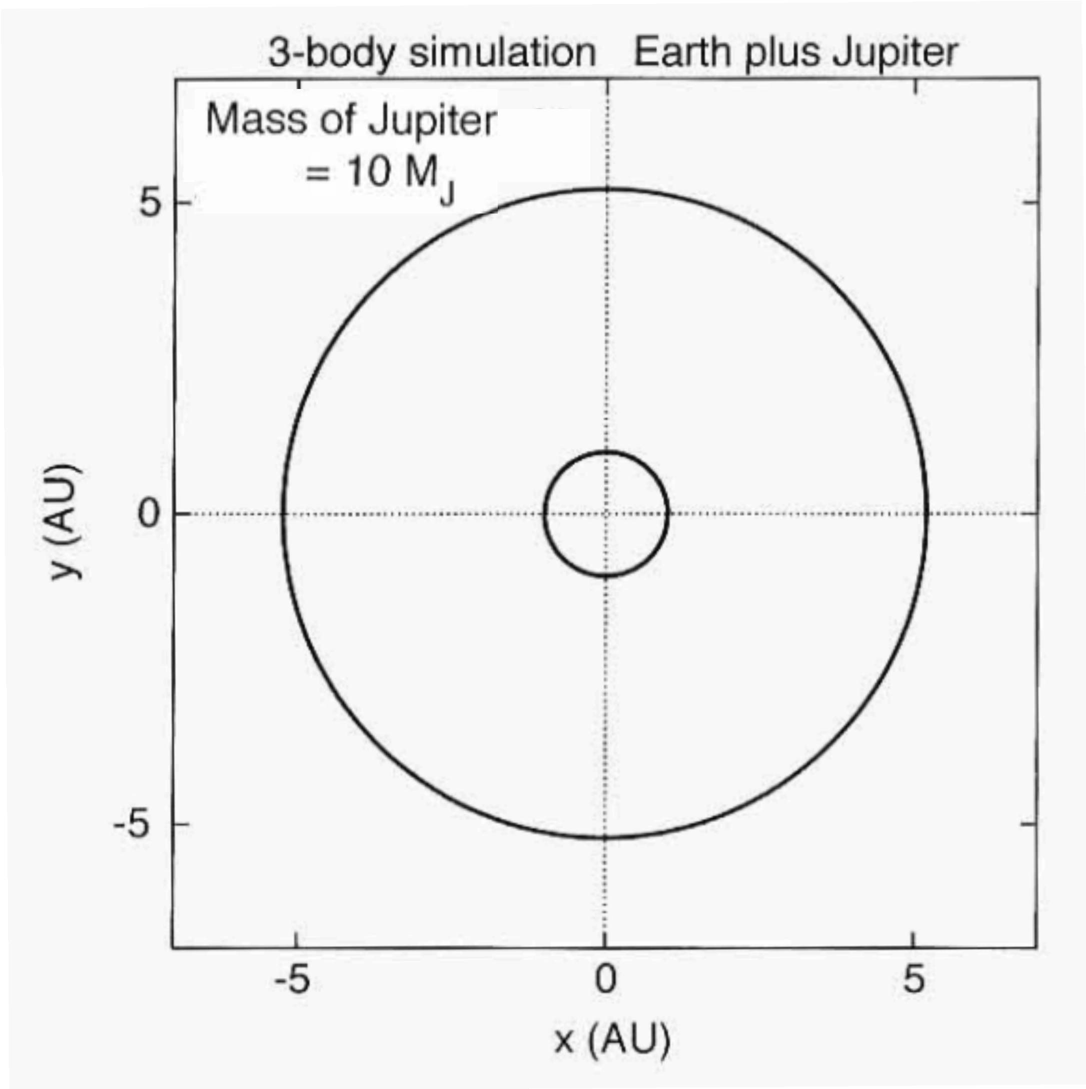
$$\frac{dx_T}{dt} = v_{T,x} \quad ; \quad \frac{dv_{T,x}}{dt} = \frac{-4\pi^2 x_T}{r_T^3} - \frac{3.8 \cdot \pi^2 \cdot 10^{-3}\left(x_T - x_J\right)}{r_{T,J}^3}$$

$$\frac{dy_T}{dt} = v_{T,y} \quad ; \quad \frac{dv_{T,y}}{dt} = \frac{-4\pi^2 y_T}{r_T^3} - \frac{3.8 \cdot \pi^2 \cdot 10^{-3}\left(y_T - y_J\right)}{r_{T,J}^3}$$

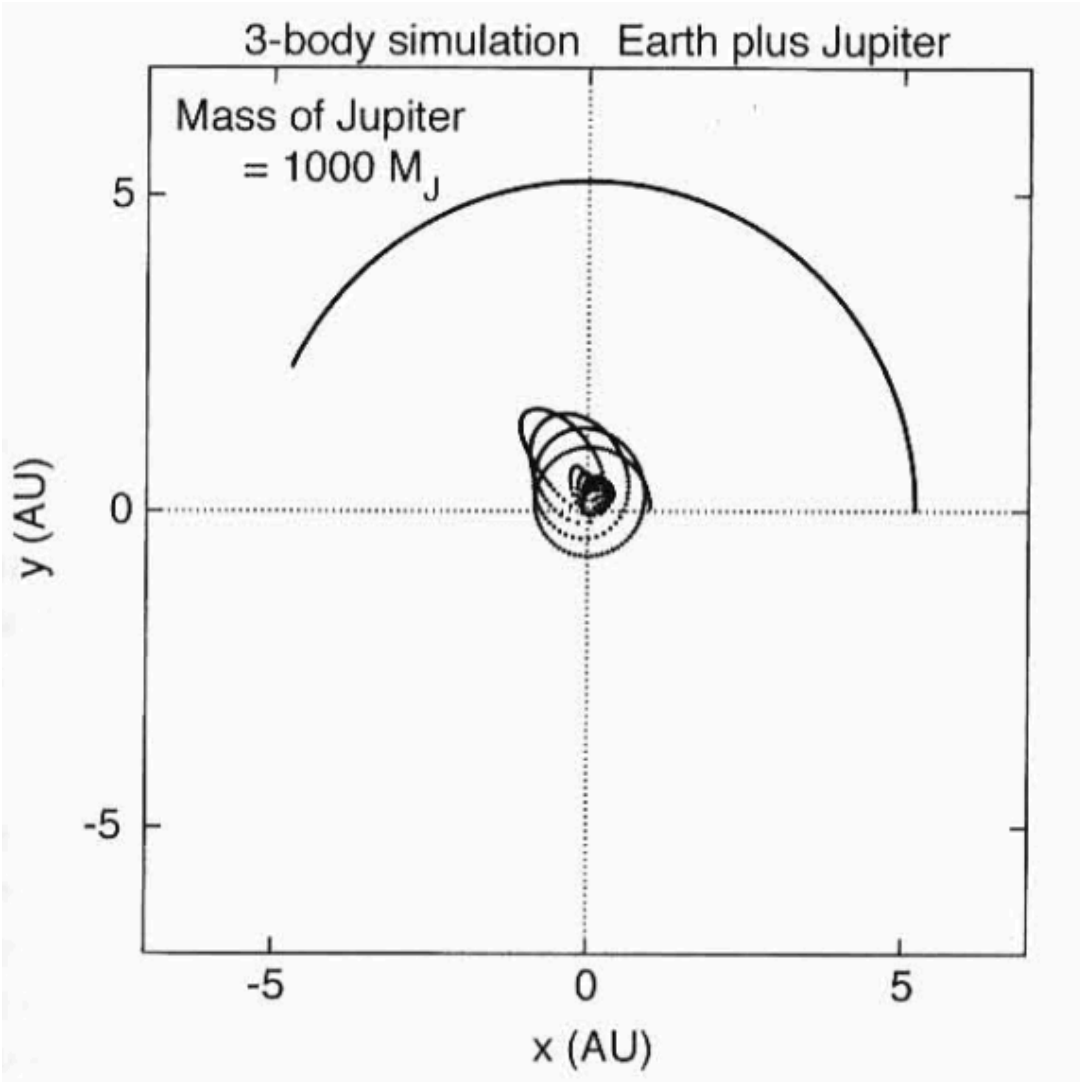
4.3 Cálculo de órbitas planetarias



4.3 Cálculo de órbitas planetarias

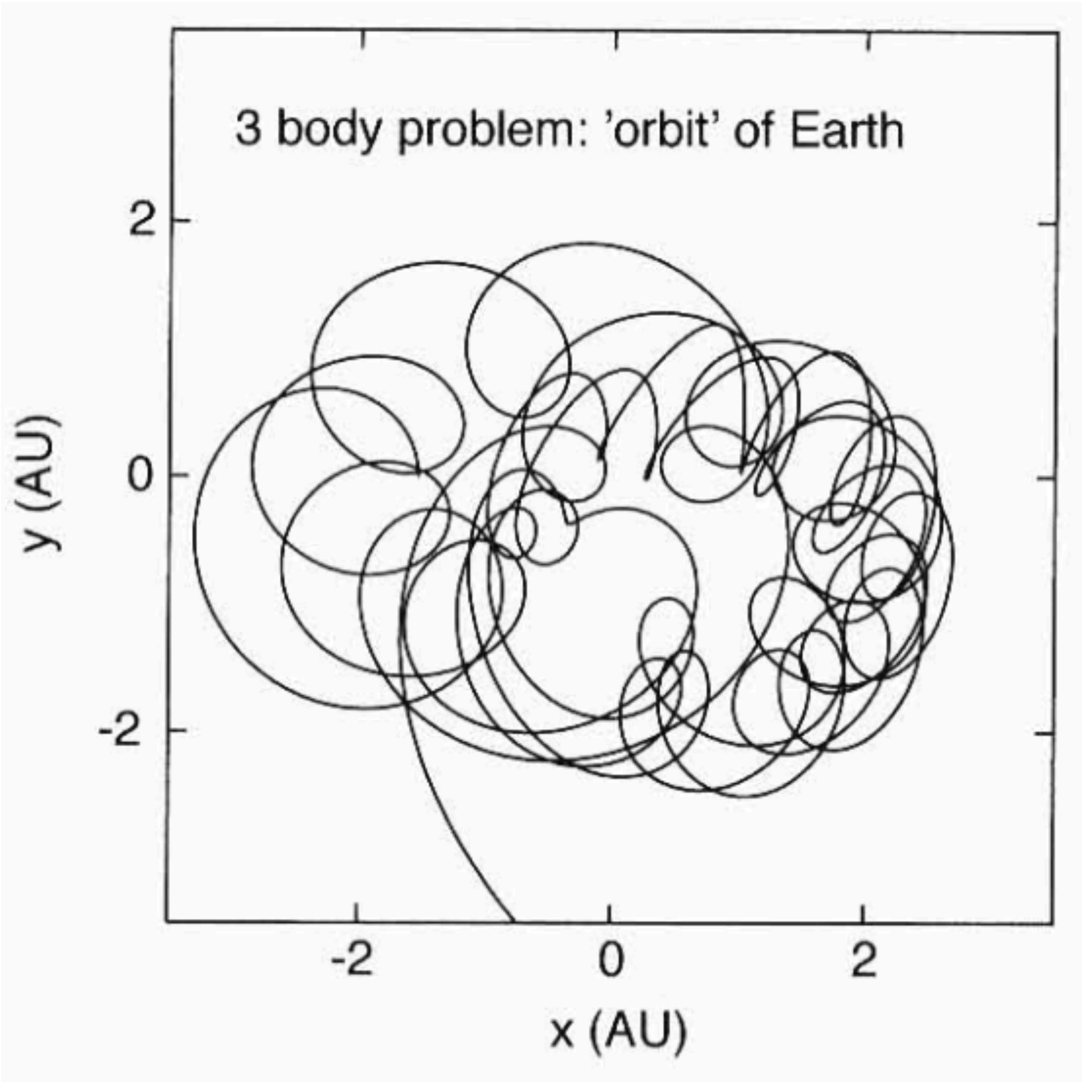


4.3 Cálculo de órbitas planetarias



Tema 4: Ecuaciones diferenciales en 2 y 3 dimensiones

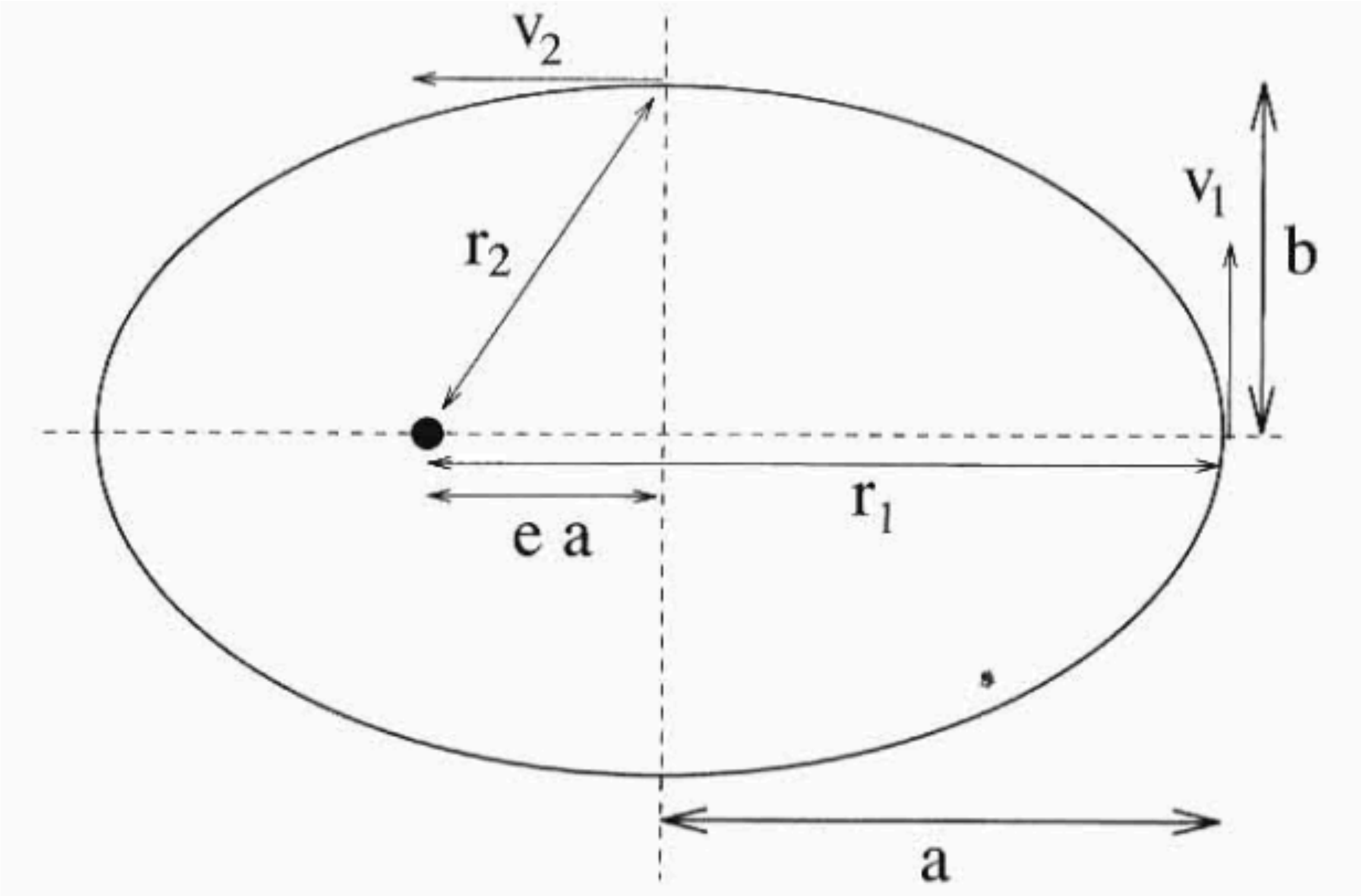
4.3 Cálculo de órbitas planetarias



4.3 Cálculo de órbitas planetarias

planet	mass (kg)	radius (AU)	eccentricity
Mercury	2.4×10^{23}	0.39	0.206
Venus	4.9×10^{24}	0.72	0.007
Earth	6.0×10^{24}	1.00	0.017
Mars	6.6×10^{23}	1.52	0.093
Jupiter	1.9×10^{27}	5.20	0.048
Saturn	5.7×10^{26}	9.54	0.056
Uranus	8.8×10^{25}	19.19	0.046
Neptune	1.03×10^{26}	30.06	0.010
Pluto	$\sim 6.0 \times 10^{24}$	39.53	0.248

4.3 Cálculo de órbitas planetarias

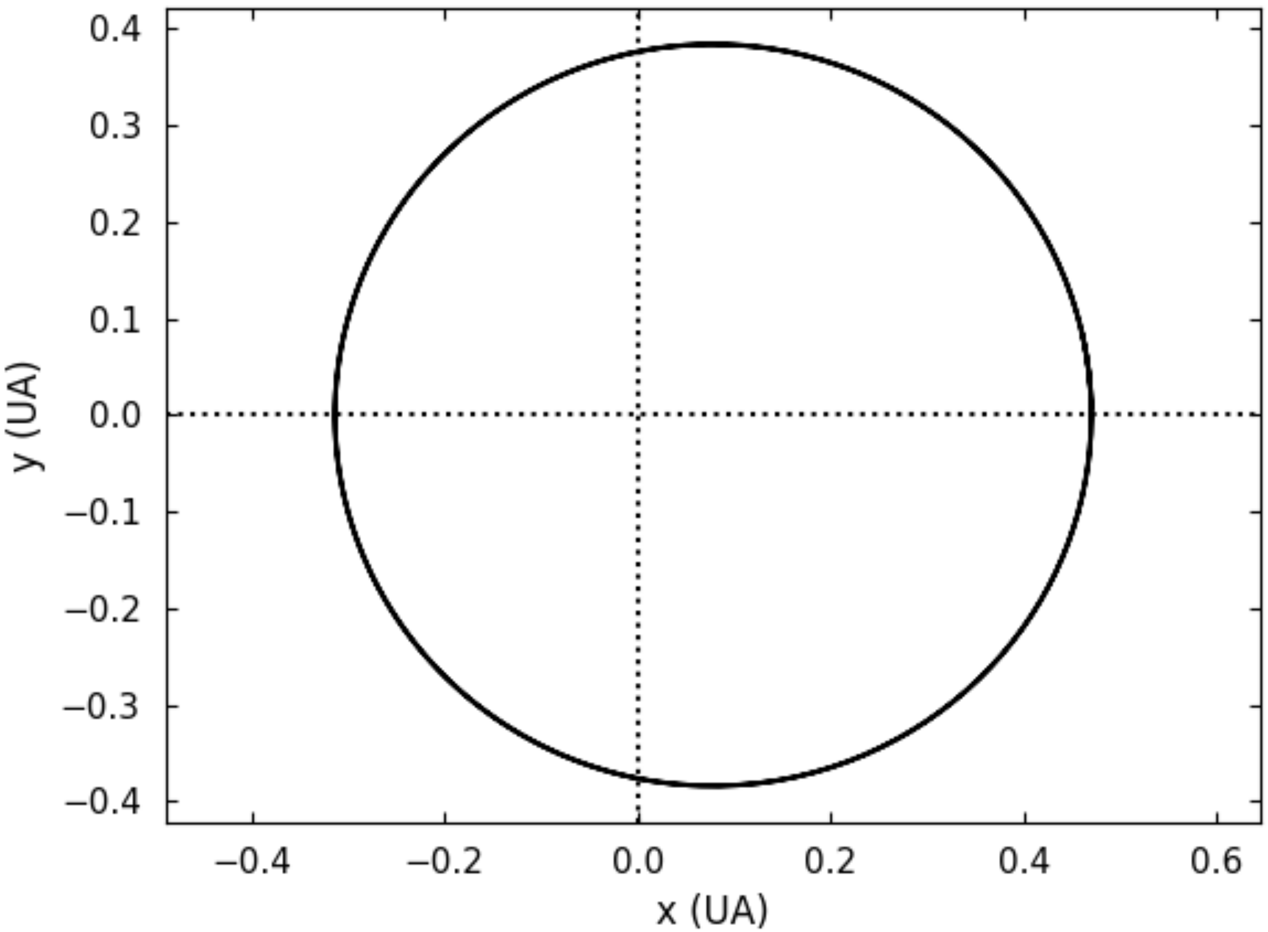


4.3 Cálculo de órbitas planetarias

Órbita de Mercurio

4.3 Cálculo de órbitas planetarias

Órbita de Mercurio



4.3 Cálculo de órbitas planetarias

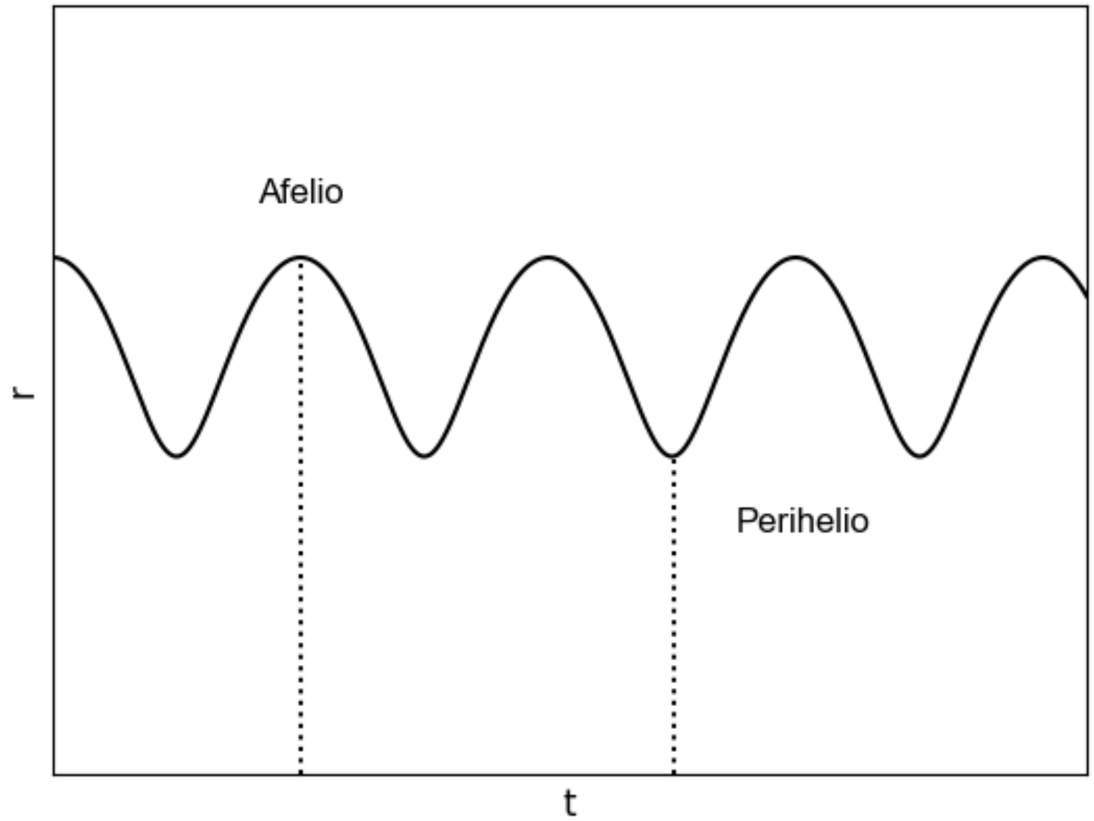
Procedimiento para calcular la precesión del perihelio de Mercurio debida a efectos relativistas:

1) Modificar el código que calcula la órbita planetaria para introducir la ecuación con los efectos relativistas.

$$F_G \approx \frac{GM_S M_M}{r^2}\left(1+\frac{\alpha}{r^2}\right)$$

2) Para cada valor de α ejecutar el programa de forma que el planeta describa varias veces la órbita
3) Detectar para cada giro el momento en que el planeta pasa por el afelio (punto más alejado del Sol) y guardar el ángulo que forma este eje con el eje x. Para ello, detectar el momento en que dr/dt=0 pasando de valores positivos a negativos.

4.3 Cálculo de órbitas planetarias



4.3 Cálculo de órbitas planetarias

Procedimiento para calcular la precesión del perihelio de Mercurio debida a efectos relativistas:

1) Modificar el código que calcula la órbita planetaria para introducir la ecuación con los efectos relativistas.

$$F_G \approx \frac{GM_S M_M}{r^2}\left(1+\frac{\alpha}{r^2}\right)$$

2) Para cada valor de α ejecutar el programa de forma que el planeta describa varias veces la órbita
3) Detectar para cada giro el momento en que el planeta pasa por el afelio (punto más alejado del Sol) y guardar el ángulo que forma este eje con el eje x. Para ello, detectar el momento en que dr/dt=0 pasando de valores positivos a negativos.
4) Representa θ frente a t. Se obtendrá una recta cuya pendiente da el ritmo de giro para el valor de α que estamos usando. Para obtener la pendiente se puede usar las herramientas de ajustes por mínimos cuadrados de numpy.

4.3 Cálculo de órbitas planetarias

Ajustes lineales con numpy

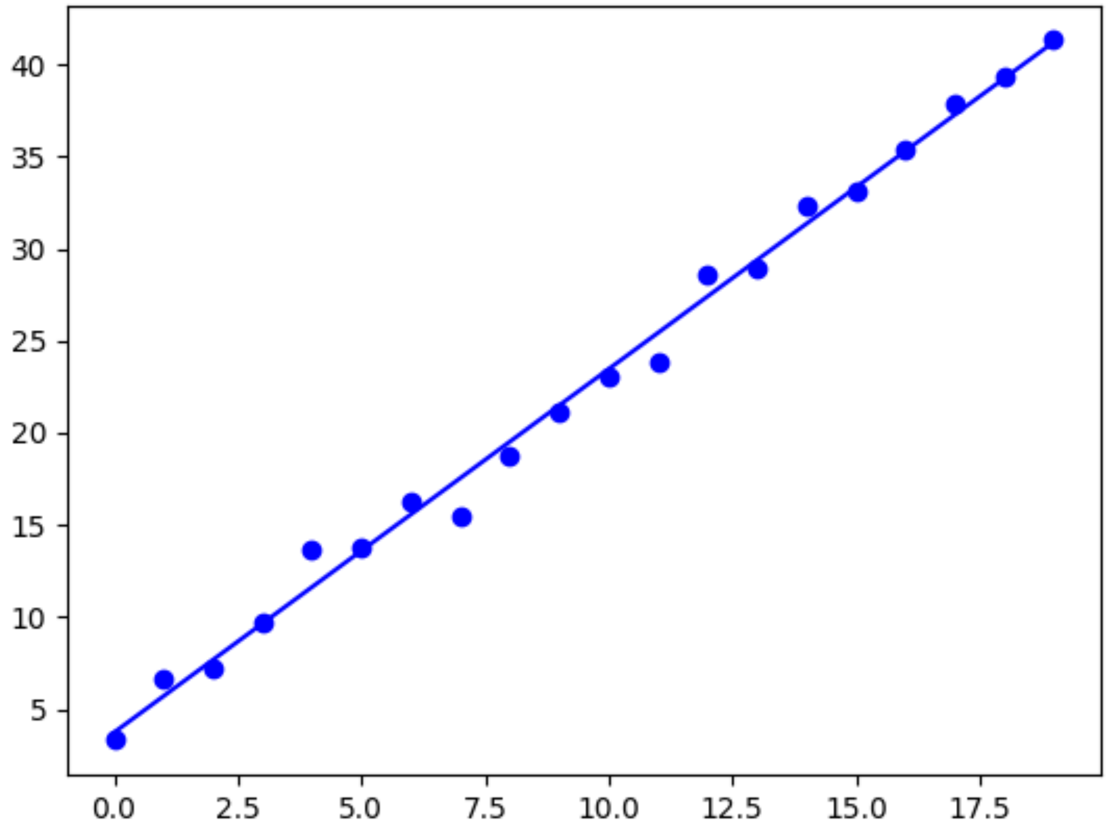
```
import numpy as np
import matplotlib.pyplot as plt

# Creación de datos con cierta dispersión:

npoints = 20
slope = 2
offset = 3
x = np.arange(npoints)
y = slope * x + offset + np.random.normal(size=npoints)

# Método 1, mediante la función polyfit

p = np.polyfit(x,y,1)           # El último argumento es el grado del polinomio
print(p)
f = p[0]*x + p[1]
plt.plot(x, y, 'bo', label="Data")
plt.plot(x, f, 'b-',label="Polyfit")
plt.show()
```

Salida:

```
>>> [1.97347829 3.72691205]
```

4.3 Cálculo de órbitas planetarias

Ajustes lineales con numpy



4.3 Cálculo de órbitas planetarias

Ajustes lineales con numpy

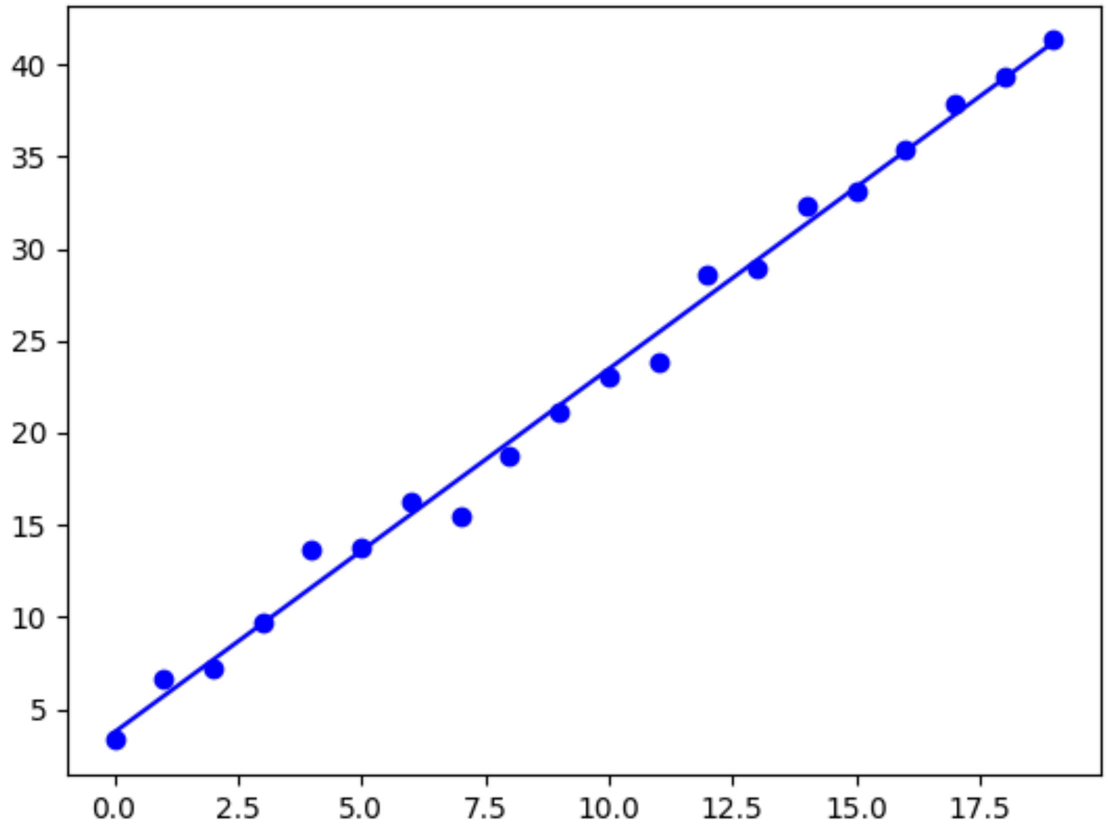
```
import numpy as np
import matplotlib.pyplot as plt

# Creación de datos con cierta dispersión:

npoints = 20
slope = 2
offset = 3
x = np.arange(npoints)
y = slope * x + offset + np.random.normal(size=npoints)

# Método 2, mediante la función lstsq

A=np.vstack([x, np.ones(len(x))]).T   # Crea una matriz de dos columnas [[x],[1]]
m, c = np.linalg.lstsq(A, y, rcond=None)[0]
print(m, c)
f = m*x + c
plt.plot(x, y, 'bo', label="Data")
plt.plot(x, f, 'b-',label="Polyfit")
plt.show()
```

Salida:

```
>>> 1.9734782865967457 3.7269120544997665
```

4.3 Cálculo de órbitas planetarias

Ajustes lineales con numpy



4.3 Cálculo de órbitas planetarias

Ajustes lineales con numpy

```
import numpy as np
import matplotlib.pyplot as plt

# Creación de datos con cierta dispersión:

npoints = 20
slope = 2
offset = 3
x = np.arange(npoints)
y = slope * x + offset + np.random.normal(size=npoints)

# Método 3, mediante la función lstsq, forzando el paso por el origen:

m = np.linalg.lstsq(x.reshape(-1,1), y, rcond=None)[0][0]
print(m)
f = m*x
plt.plot(x, y, 'bo', label="Data")
plt.plot(x, f, 'b-',label="Polyfit")
plt.show()
```

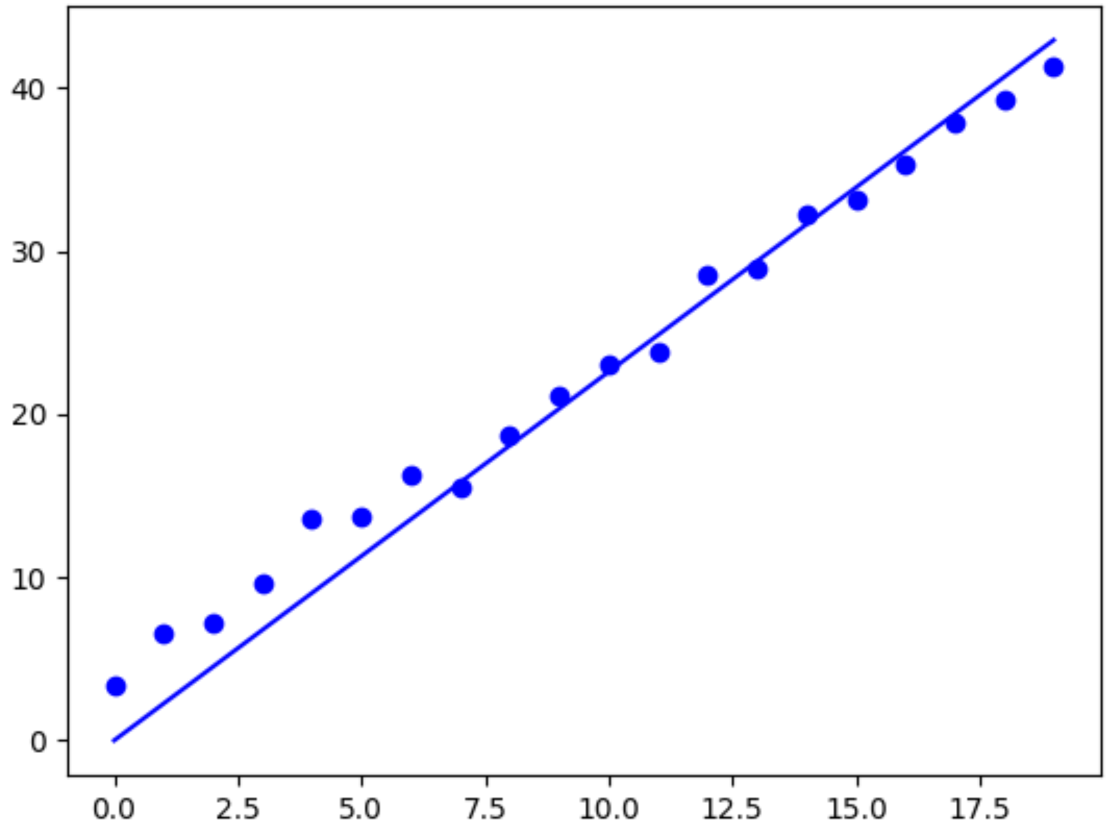
Salida:

```
>>> 2.260163829250574
```

Tema 4: Ecuaciones diferenciales en 2 y 3 dimensiones

4.3 Cálculo de órbitas planetarias

Ajustes lineales con numpy



4.3 Cálculo de órbitas planetarias

Procedimiento para calcular la precesión del perihelio de Mercurio debida a efectos relativistas:

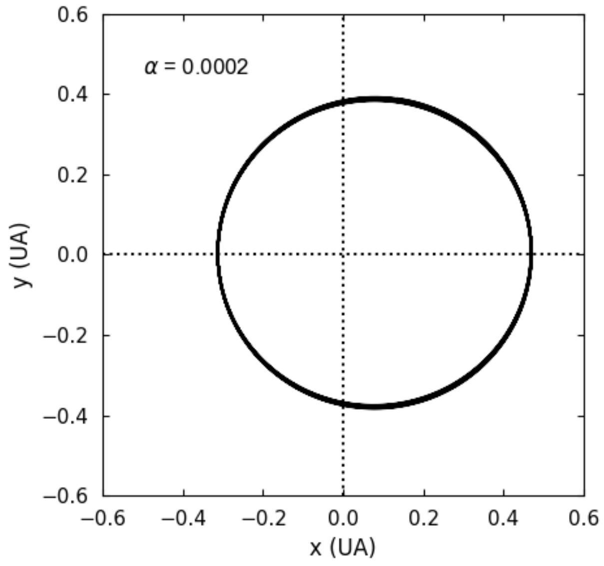
1) Modificar el código que calcula la órbita planetaria para introducir la ecuación con los efectos relativistas.

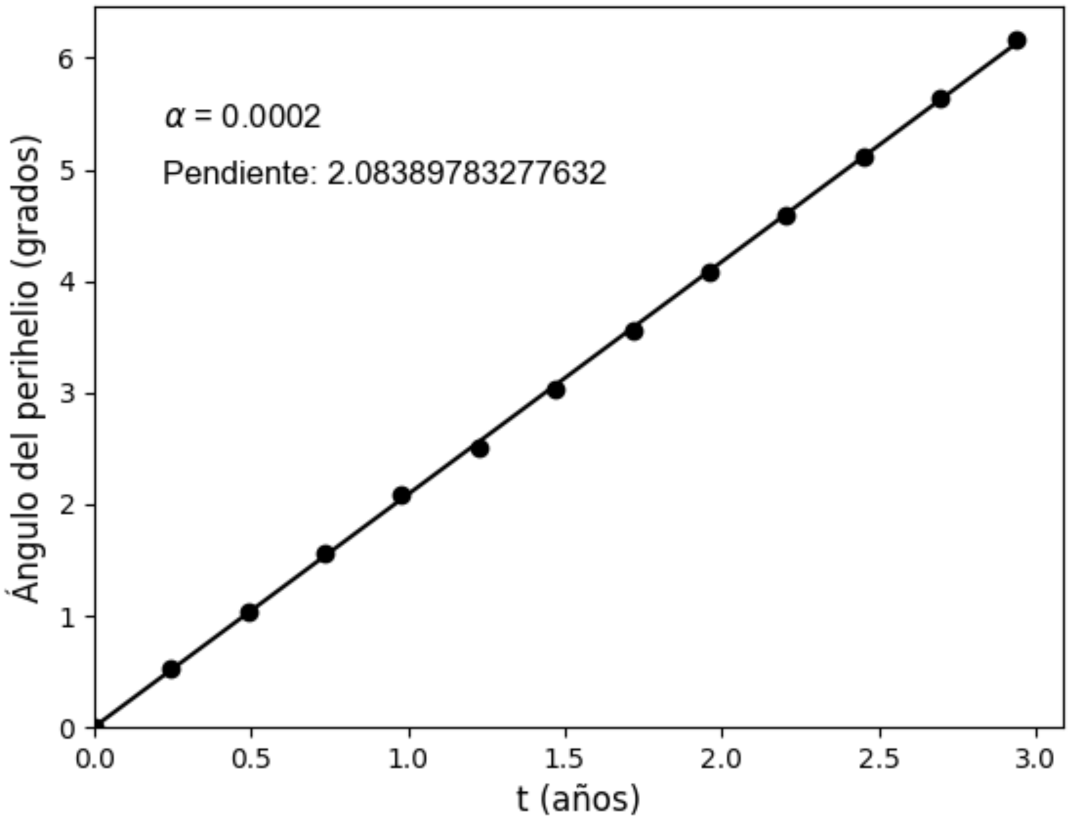
$$F_G \approx \frac{GM_S M_M}{r^2}\left(1+\frac{\alpha}{r^2}\right)$$

2) Para cada valor de α ejecutar el programa de forma que el planeta describa varias veces la órbita
3) Detectar para cada giro el momento en que el planeta pasa por el afelio (punto más alejado del Sol) y guardar el ángulo que forma este eje con el eje x. Para ello, detectar el momento en que dr/dt=0 pasando de valores positivos a negativos.
4) Representa θ frente a t. Se obtendrá una recta cuya pendiente da el ritmo de giro para el valor de α que estamos usando. Para obtener la pendiente se puede usar las herramientas de ajustes por mínimos cuadrados de numpy.
5) Tras hacer lo anterior para varios valores de α, representar dθ/dt frente a α. Debería obtenerse una recta que pase por el origen, $d\theta/dt = C\cdot\alpha$.
6) Hacer un ajuste lineal para obtener C. Sustituir el valor real de α ($\alpha=1.1\times10^{-8}$) y obtener de esta manera el valor de dθ/dt obtenido (valor real: 0.43 arcseg/siglo.)

4.3 Cálculo de órbitas planetarias

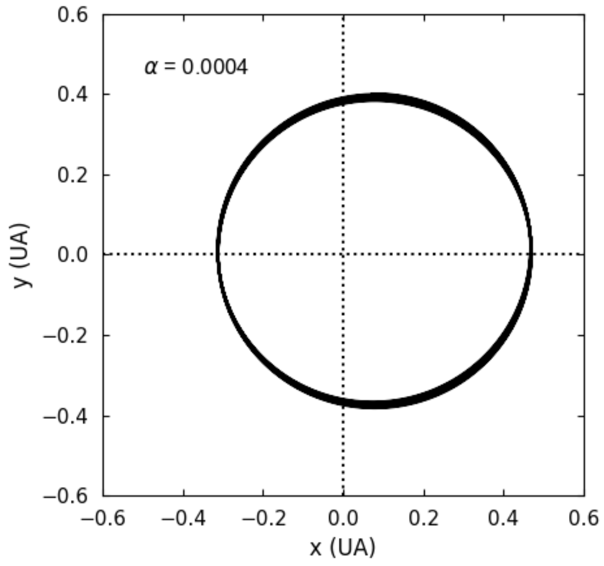
Resultados

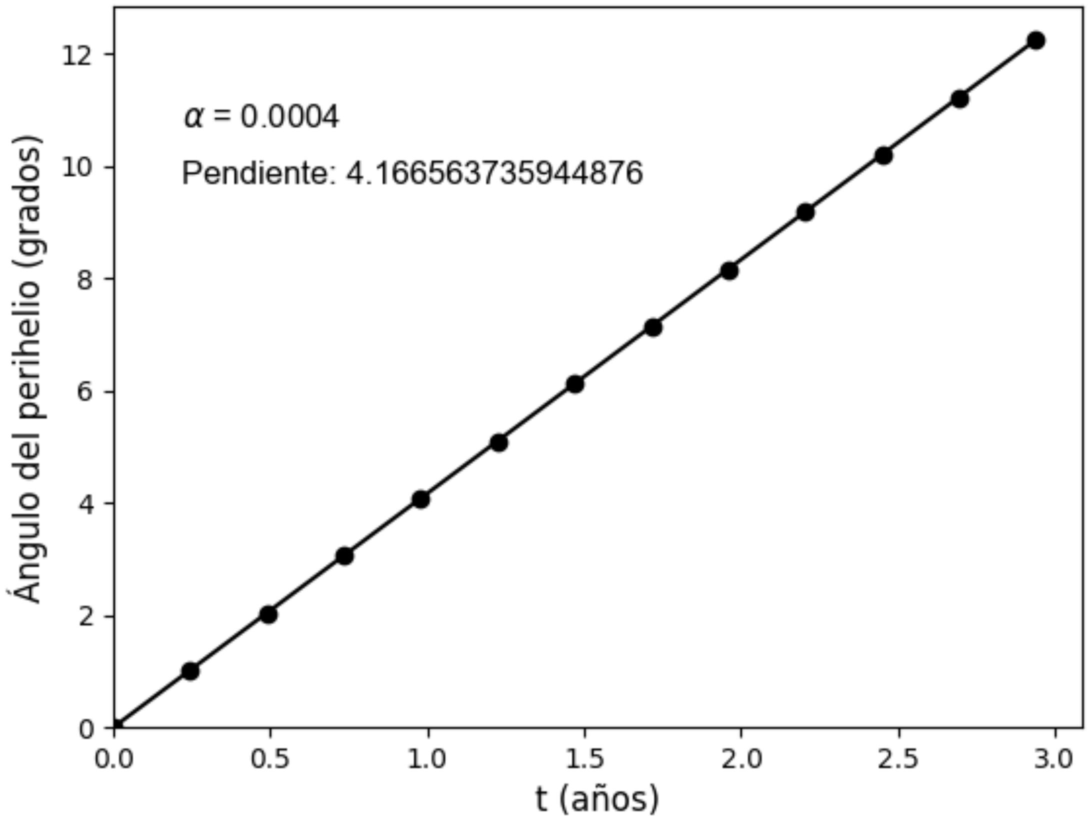




4.3 Cálculo de órbitas planetarias

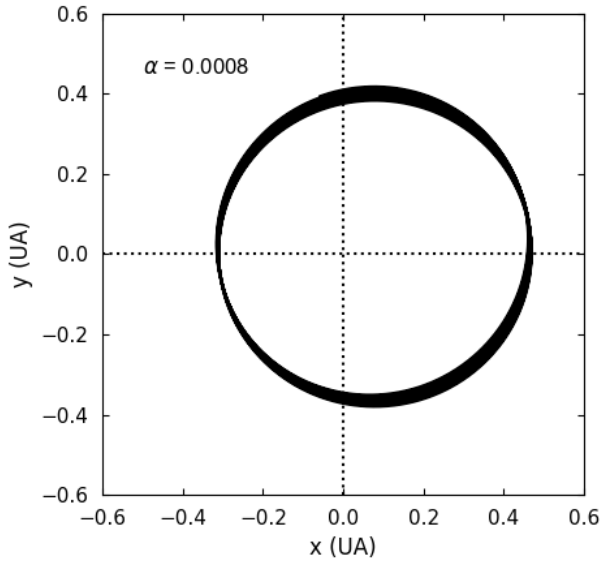
Resultados

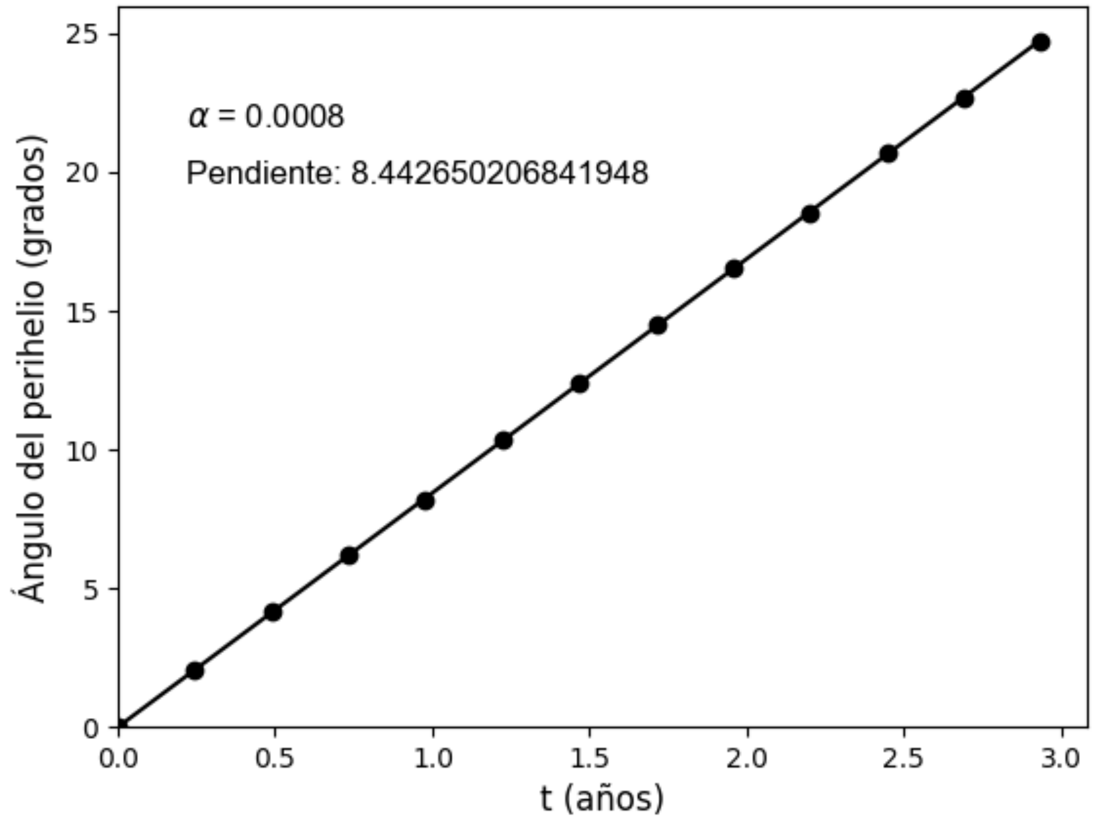




4.3 Cálculo de órbitas planetarias

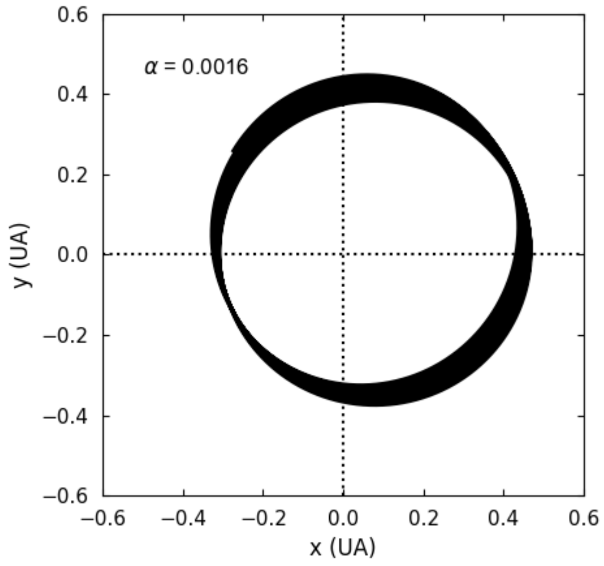
Resultados

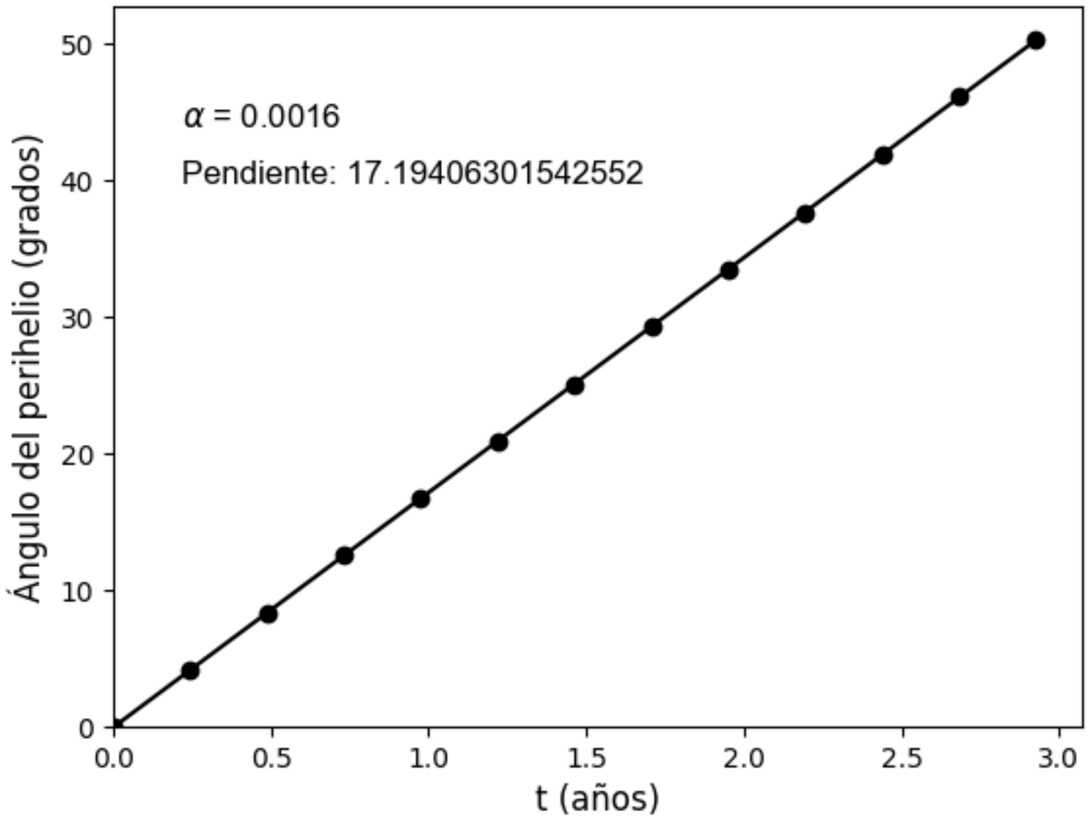




4.3 Cálculo de órbitas planetarias

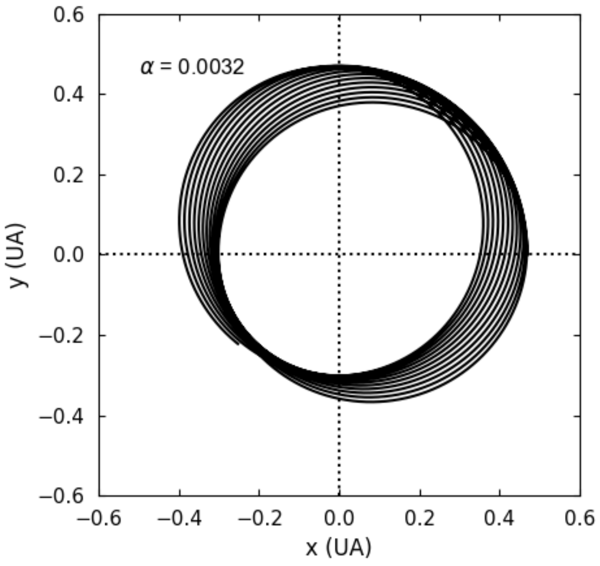
Resultados

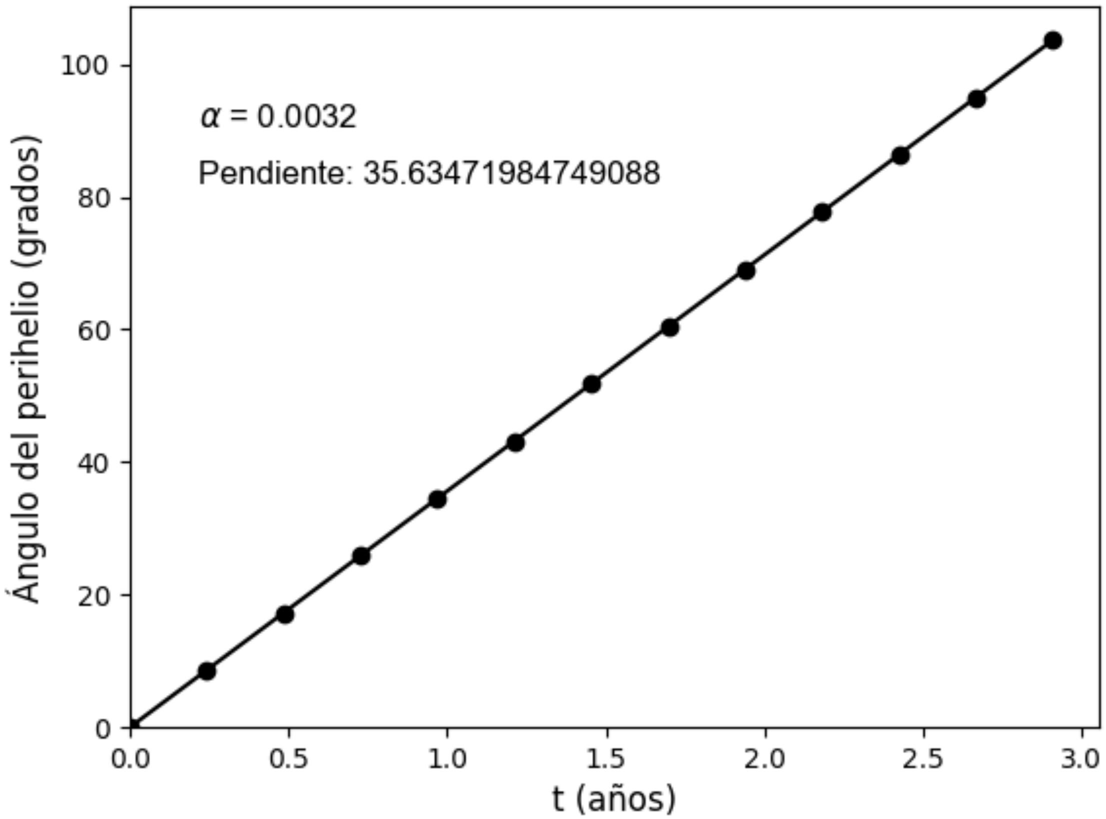




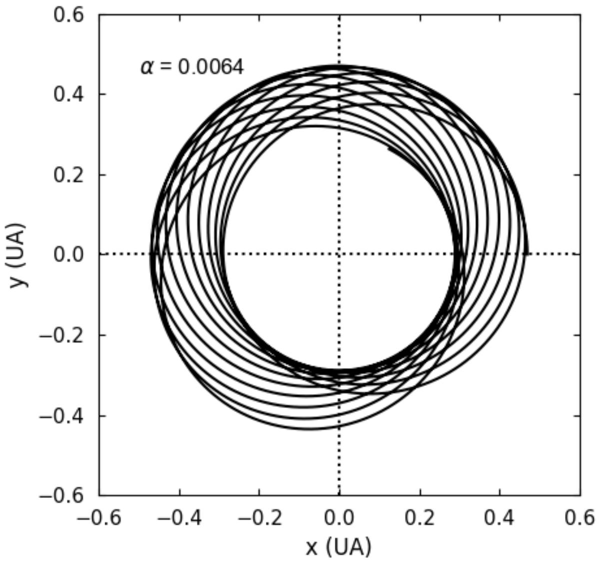
4.3 Cálculo de órbitas planetarias

Resultados

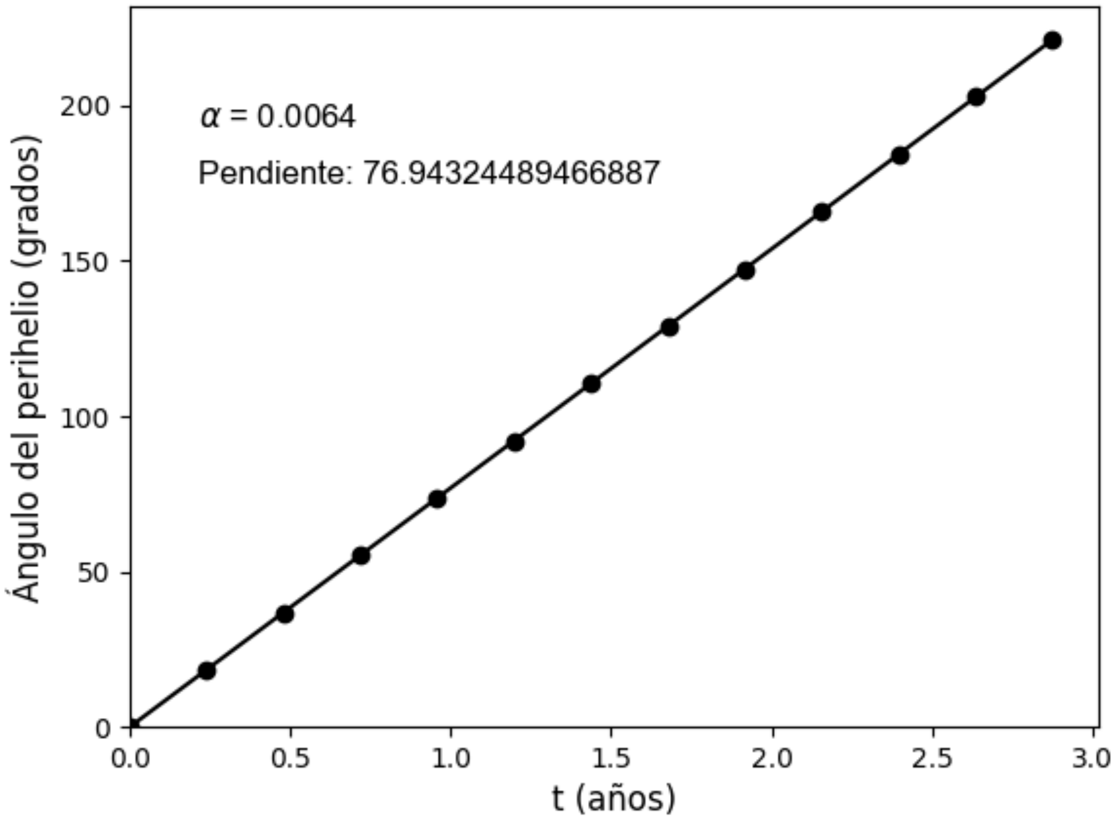




Resultados



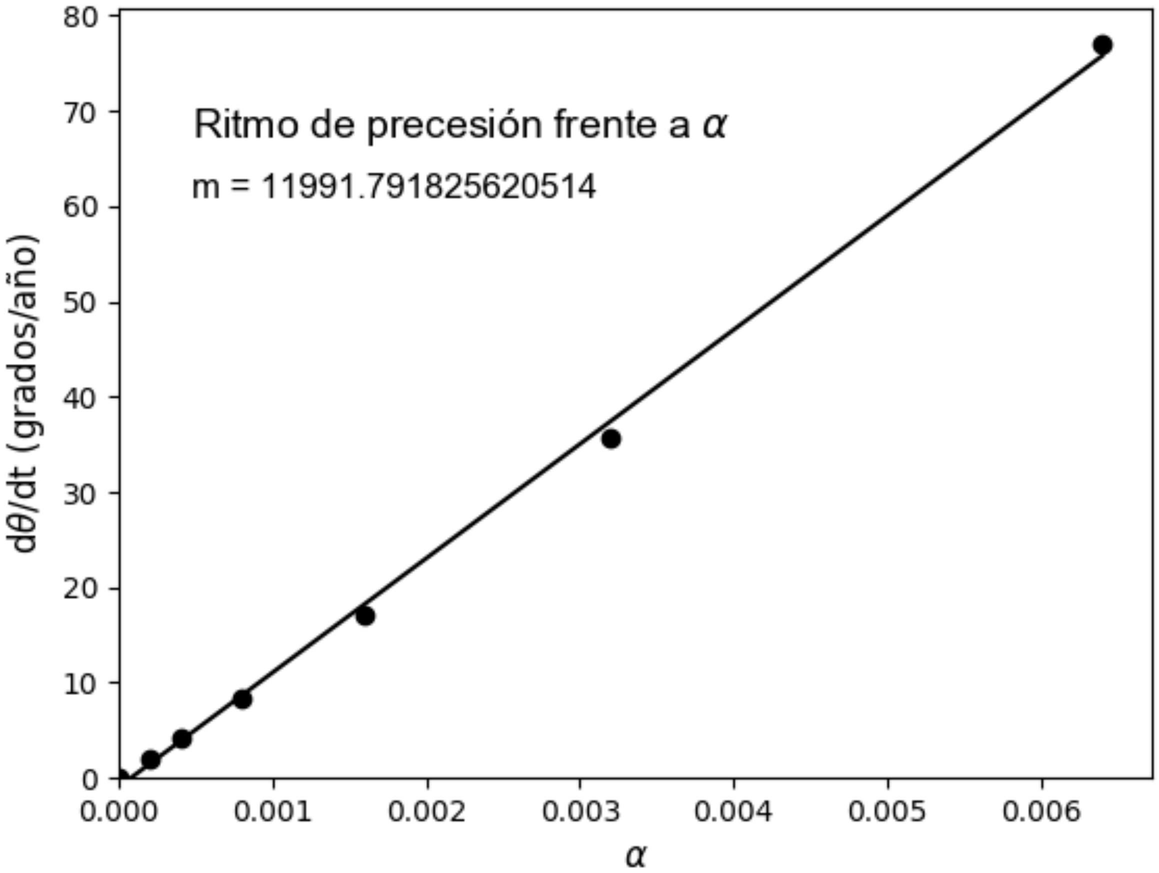
α = 0.0064
0.6
0.4
0.2
0.0
−0.2
−0.4
−0.6
y (UA)
x (UA)



α = 0.0064
Pendiente: 76.94324489466887
Ángulo del perihelio (grados)
t (años)
0
50
100
150
200
0.0
0.5
1.0
1.5
2.0
2.5
3.0

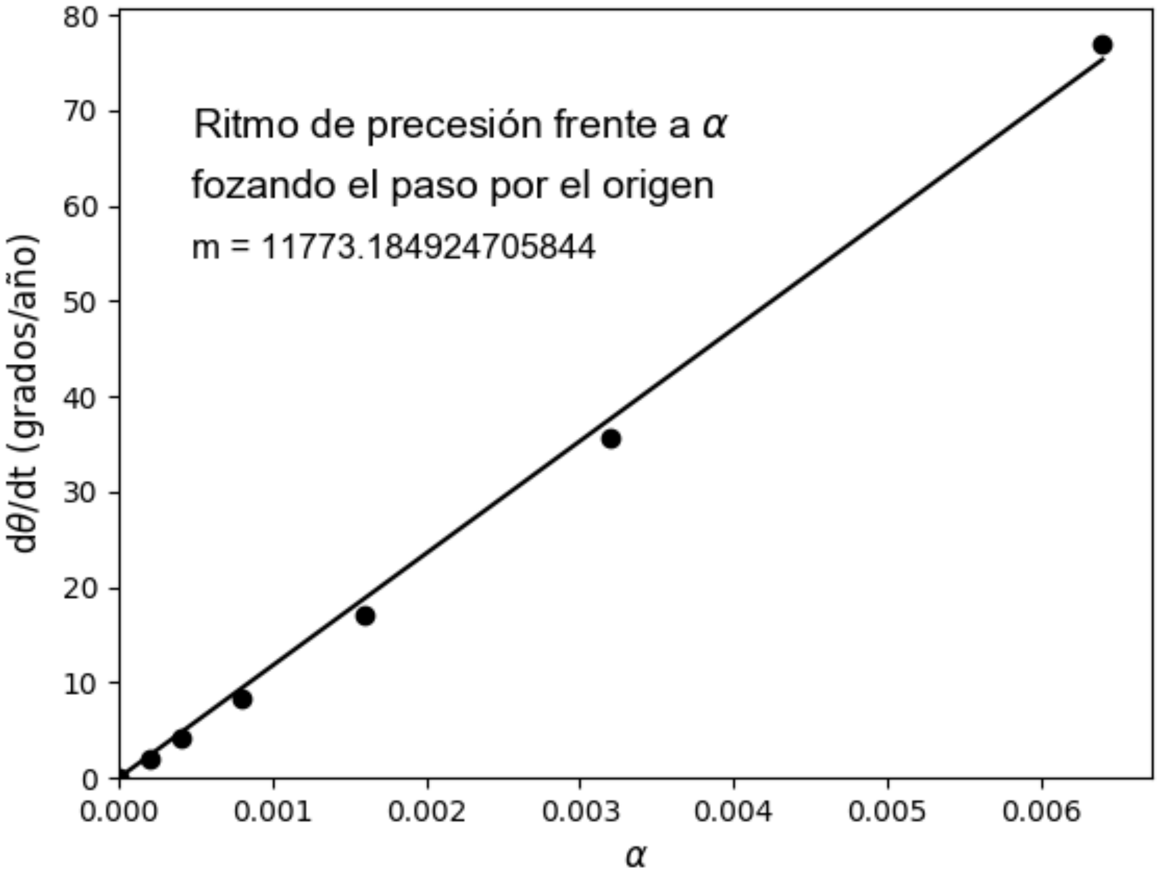
4.3 Cálculo de órbitas planetarias

Resultados



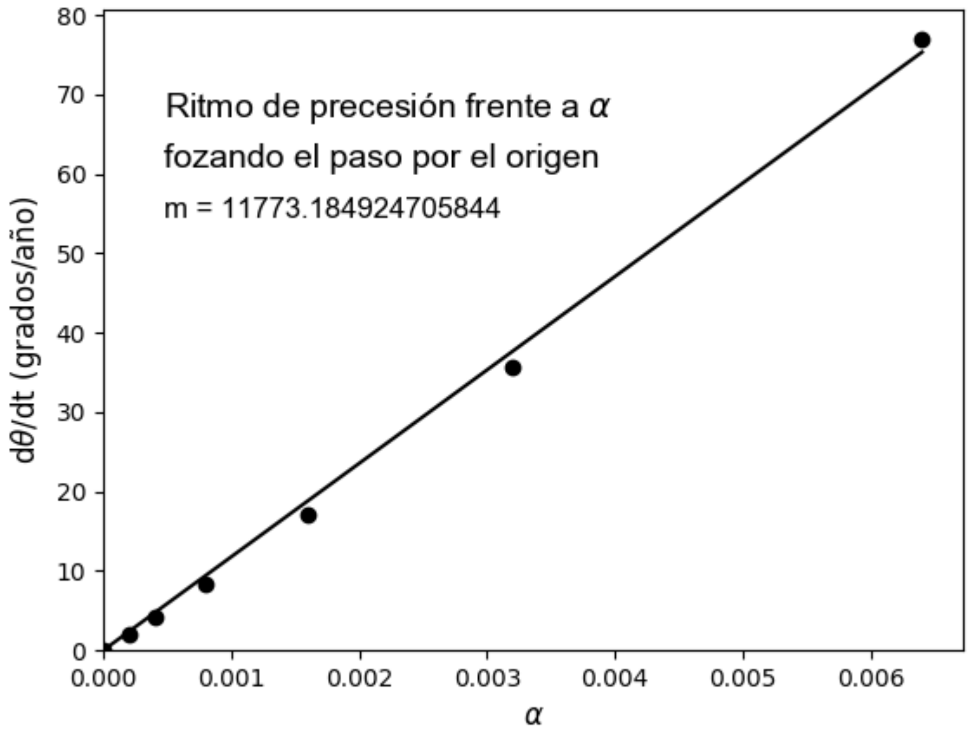
4.3 Cálculo de órbitas planetarias

Resultados



4.3 Cálculo de órbitas planetarias

Resultados



Ritmo de precesión para $\alpha = 1.1 \cdot 10^{-8}$ UA^2: 0.0001295 grados/año

Ritmo de precesión en arcosegundos/siglo: 46.62