Universidad de Chile
Facultad de Ciencias Físicas y Matemáticas
Departamento de Ingeniería Matemática

Laboratorio 4

Problema Lineal Cuadrático

Estudiantes: Maximiliano S. Lioi
Isidora Reyes
Profesor: Héctor Ramírez C.
Auxiliar: Matías V. Vera
Ayudante de laboratorio: S. Adrián Arellano
Santiago de Chile

Índice de Contenidos

Índice de Figuras

Índice de Códigos

1. Contexto

Consideramos una partícula en el espacio, la cuál está sometida a las fuerzas dadas por $F_x = x + u$ y $F_y = y + v$, donde u y v son los controles del sistema. Además de lo anterior, la velocidad del eje z depende de la posición en los ejes x e y de la forma x + y. El sistema queda entonces descrito por la ecuación diferencial:

$$\ddot{x} = x + v \ ; \ \ddot{y} = y + v \ ; \ \dot{z} = x + y$$

La partícula tiene masa unitaria, y en el instante inicial, se encuentra en reposo en la posición $(1, 1, 0)$.

Luego el sistema controlado se representa de forma matricial considerando

$$X(t) = \begin{bmatrix} x(t) \\ \dot{x}(t) \\ y(t) \\ \dot{y}(t) \\ z(t) \end{bmatrix}$$

$$\dot{X}(t) = \begin{bmatrix} \dot{x}(t) \\ x(t) + u(t) \\ \dot{y}(t) \\ y(t) + v(t) \\ x(t) + y(t) \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 \end{bmatrix} X(t) + \begin{bmatrix} 0 & 0 \\ 1 & 0 \\ 0 & 0 \\ 0 & 1 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} u(t) \\ v(t) \end{bmatrix}$$

De ahora en adelante, consideraremos $t_f = 5$ controles irrestrictos.

El problema presentado ahora, consiste en minimizar la fuerza total ejercida, en conjunto con la energía cinética al final del tramo, en particular, se debe minimizar el funcional:

$$J(u(\cdot), v(\cdot)) := K(t_f) + \int_0^{t_f} \|(u(t), v(t))\|^2 dt,$$

donde $K(t) := \frac{1}{2} m(t) \mathrm{V}(t)^2$ es la energía cinética, $m(t)$ es la masa y $\mathrm{V}(t)$ es la velocidad, para nuestro modelo se tienen los datos

$$V(t)^2 = \dot{x}^2(t) + \dot{y}^2(t) + \dot{z}^2(t) = \dot{x}^2(t) + \dot{y}^2(t) + (x(t) + y(t))^2$$

$$m(t) = 1$$

2. Ejercicio 1

Se modela la situación como un problema lineal cuadrático, identificando las matrices Q(·), W (·), U (·), nos interesa ver si se cumplen las condiciones vistas en cátedra para asegurar existencia y unicidad del nuevo problema lineal cuadrático para los controles óptimos $(u(t), v(t))$.

Queremos minimizar el funcional

$$J(u(\cdot), v(\cdot)) = \frac{m(t_f)V(t_f)^2}{2} + \int_0^{t_f} \|(u(t), v(t))\|^2 dt$$

Para ello, buscamos escribir el funcional de la forma

$$J(u(\cdot)) = X(t_f)^T Q X(t_f) + \int_0^{t_f} X(t)^T W(t) X(t) + u(t)^T U(t) u(t) dt$$

Tenemos masa unitaria y suponemos que la masa no varía a lo largo del tiempo, luego

$$m(t_f) = 1$$

Además, podemos expresar la velocidad de la partícula como

$$V(t)^2 = \dot{x}^2(t) + \dot{y}^2(t) + \dot{z}^2(t)$$

La dinámica del sistema nos dice que $\dot{z} = x + y$, por lo que

$$\dot{z}^2(t_f) = x^2(t_f) + 2x(t_f)y(t_f) + y^2(t_f)$$

Entonces, el criterio de minimización tiene la forma explícita

$$J(u(\cdot), v(\cdot)) = \frac{1}{2}(\dot{x}(t_f)^2 + \dot{y}(t_f)^2 + x^2(t_f) + 2x(t_f)y(t_f) + y^2(t_f)) + \int_0^{t_f} u(t)^2 + v(t)^2 dt$$

Tenemos para una matriz Q que la forma cuadrática

$$x^T Q x = \sum_{i=1}^{n} \sum_{j=1}^{n} x_i Q_{ij} x_j$$

Por lo que necesitamos que

$$Q_{22} = 1, Q_{44} = 1, Q_{11} = 1, Q_{33} = 1, Q_{13} = 1, Q_{31} = 1$$

Para obtener $\dot{x}(t_f)^2 + \dot{y}(t_f)^2 + x^2(t_f) + 2x(t_f)y(t_f) + y^2(t_f)$, luego la matriz Q debe ser

$$Q = \frac{1}{2}\begin{bmatrix} 1 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

Por lo demás, los valores propios de Q son

$$\lambda_1 = 0, \lambda_2 = 0, \lambda_3 = 1, \lambda_4 = 1/2, \lambda_5 = 1/2$$

Y por lo tanto $Q \in S^n_+$, por lo demás, tomamos $W(t) = 0_n \in S^n_+$ y $U(t) = I_n \in S^n_{++}$, de donde concluimos que se cumplen las hipótesis que asumimos en cátedra, a partir de estas condiciones y usando que $U(t) = I$, que implica $||u||_{L^2} = ||u||_U$, se tiene la buena posición del problema lineal cuadrático.

3. Ejercicio 2

Usando el Principio del Máximo de Pontryagin deseamos probar que el control óptimo esta dado por $(u^*(t), v^*(t)) = (p_2(t), p_4(t))$, con $p(\cdot)$ solución de:

$$\dot{p}(t) = -\begin{pmatrix} 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix} p(t); \quad p\left(t_f\right) = -\frac{1}{2}\begin{pmatrix} x\left(t_f\right) + y\left(t_f\right) \\ \dot{x}\left(t_f\right) \\ x\left(t_f\right) + y\left(t_f\right) \\ \dot{y}\left(t_f\right) \end{pmatrix}$$

Tenemos del principio del máximo de Pontryagin para el Problema Lineal Cuadrático que existe una función absolutamente continua $p(\cdot)$ tal que

$$\dot{p}(t) = -A(t)^T + W(t)x(t); p(t_f) = -Qx(t_f)$$

Para $x(t)$ la trayectoria óptima asociada al problema, luego tomando los datos del problema tenemos el sistema

$$\dot{p}(t) = -\begin{bmatrix} 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 \end{bmatrix}^T p(t)$$

Equivalentemente

$$\dot{p}(t) = -\begin{bmatrix} 0 & 1 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix} p(t) = -\begin{bmatrix} p_2(t) + p_5(t) \\ p_1(t) \\ p_4(t) + p_5(t) \\ p_3(t) \\ 0 \end{bmatrix}$$

Con condición de borde

$$p(t_f) = -QX(t_f) = -\frac{1}{2}\begin{bmatrix} 1 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix} X(t_f) = -\frac{1}{2}\begin{bmatrix} x(t_f) + y(t_f) \\ \dot{x}(t_f) \\ x(t_f) + y(t_f) \\ \dot{y}(t_f) \\ 0 \end{bmatrix}$$

De esto último, deducimos que $p_5(t) = cte$, además, sabemos que $p_5(t_f) = 0$, por lo tanto, $p_5(t) = 0 \forall t \in [0, t_f]$, de donde obtenemos el sistema equivalente

$$\begin{bmatrix} \dot{p}_1(t) \\ \dot{p}_2(t) \\ \dot{p}_3(t) \\ \dot{p}_4(t) \end{bmatrix} = - \begin{bmatrix} p_2(t) \\ p_1(t) \\ p_4(t) \\ p_3(t) \end{bmatrix} = - \begin{bmatrix} 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix} p(t)$$

Junto a las condición de borde

$$p(t_f) = -\frac{1}{2} \begin{bmatrix} x(t_f) + y(t_f) \\ \dot{x}(t_f) \\ x(t_f) + y(t_f) \\ \dot{y}(t_f) \end{bmatrix}$$

El control óptimo toma la expresión, recordando que $p_5(t) = 0$

$$u(t) = U(t)^{-1} B(t)^T p(t) = I \begin{bmatrix} 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \end{bmatrix} p(t)$$

De donde obtenemos que el control óptimo viene dado por

$$\begin{bmatrix} u(t) \\ v(t) \end{bmatrix} = \begin{bmatrix} p_2(t) \\ p_4(t) \end{bmatrix}$$

4. Ejercicio 3

Se utiliza el resultado anterior, junto a `scipy.integrate.solve.ivp` y nuevamente `scipy.optimize.minimize`, con tal de encontrar un control óptimo para el problema lineal cuadrático. Posteriormente, se realizan gráficos de las trayectorias óptimas junto a los controles óptimos, y el valor objetivo mínimo que alcanza el problema.

Considerando que el control óptimo se escribe

$$\begin{bmatrix} u(t) \\ v(t) \end{bmatrix} = \begin{bmatrix} p_2(t) \\ p_4(t) \end{bmatrix}$$

Definimos el sistema acoplado

$$\dot{\Phi}(t) = \begin{bmatrix} \dot{X} \\ \dot{p} \end{bmatrix} = \begin{bmatrix} \dot{x}(t) \\ x(t) + p_2(t) \\ \dot{y}(t) \\ y(t) + p_4(t) \\ x(t) + y(t) \\ -p_2(t) \\ -p_1(t) \\ -p_4(t) \\ -p_3(t) \end{bmatrix} = A\Phi(t)$$

Donde A viene dado por

$$A = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 \\ 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 \end{bmatrix}$$

Con condición de término

$$\Phi(t_f) = \begin{bmatrix} X(t_f) \\ p(t_f) \end{bmatrix}$$

Notamos que no conocemos a priori $p(0)$ ni $X(t_f)$, luego para resolver la ecuación diferencial mediante *Scipy* con el método *solve-ivp* necesitamos encontrar condiciones iniciales $p(0)$ de manera que la solución con condición inicial

$$\Phi(0) = \begin{bmatrix} X(0) \\ p(0) \end{bmatrix}$$

cumpla la condición de término $p(t_f) = -QX(t_f)$ para la solución en $p(\cdot)$. Para ello definimos un problema de optimización que consiste en recibir una tupla $(p_1(0), p_2(0), p_3(0), p_4(0))$, resolver la ecuación diferencial para $\Phi(\cdot)$ en el intervalo $[0, t_f]$ y minimizar la cantidad

$$||\Phi(t_f) - \begin{bmatrix} X(t_f) \\ -\frac{1}{2}x(t_f) - \frac{1}{2}y(t_f) \\ -\frac{1}{2}\dot{x}(t_f) \\ -\frac{1}{2}x(t_f) - \frac{1}{2}y(t_f) \\ -\frac{1}{2}\dot{y}(t_f) \end{bmatrix} ||_2$$

Equivalentemente, considerando $p(0) \in \mathbb{R}^4$ y $\Phi(t) = \begin{bmatrix} X(t) \\ p(t) \end{bmatrix}$ solución de la ecuación $\dot{\Phi}(t) = A\Phi(t)$ con condiciones iniciales $\Phi(0) = \begin{bmatrix} X(0) \\ p(0) \end{bmatrix}$, definimos el problema de optimización

$$\min_{p(0)\in\mathbb{R}^4} ||p(t_f) - \begin{bmatrix} -\frac{1}{2}x(t_f) - \frac{1}{2}y(t_f) \\ -\frac{1}{2}\dot{x}(t_f) \\ -\frac{1}{2}x(t_f) - \frac{1}{2}y(t_f) \\ -\frac{1}{2}\dot{y}(t_f) \end{bmatrix} ||_2$$

$$\text{sujeto a} \;\; \dot{\Phi}(t) = A\Phi(t), \quad \Phi(0) = \begin{bmatrix} X(0) \\ p(0) \end{bmatrix}$$

De esta manera encontramos el valor de $p(0)$ que hace factible las ecuaciones del principio del máximo de Pontryagin, pues como la EDO es lineal, pasa una única curva integral por $t = 0$ y $t = 5$

Notamos además que podemos escribir

$$p(t_f) = -\frac{1}{2} \begin{bmatrix} 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x(t_f) \\ \dot{x}(t_f) \\ y(t_f) \\ \dot{y}(t_f) \end{bmatrix} = -\frac{1}{2} \begin{bmatrix} x(t_f) + y(t_f) \\ \dot{x}(t_f) \\ x(t_f) + y(t_f) \\ \dot{y}(t_f) \end{bmatrix}$$

Que será útil para efectos del código.

Para su resolución numérica, se definen funciones para resolver la dinámica acoplada y se resuelve el problema de optimización expresando el respectivo problema en su versión discretizada para ser resuelto con *minimize* para encontrar el $p(0)$ óptimo, el cual sabemos que es el valor de $p(0)$ real para el sistema acoplado, pues como la ecuación diferencial es lineal, en particular, del TEU existe una única curva integral que conecta las soluciones para los tiempos $p(0)$ y $p(t_f)$, como la solución con condición inicial para dicho $p(0)$ es tal que

$$p(t_f) = \begin{bmatrix} -\frac{1}{2}x(t_f) - \frac{1}{2}y(t_f) \\ -\frac{1}{2}\dot{x}(t_f) \\ -\frac{1}{2}x(t_f) - \frac{1}{2}y(t_f) \\ -\frac{1}{2}\dot{y}(t_f) \end{bmatrix}$$

Concluimos que $p(0)$ es el valor real asociado a las soluciones del problema, de donde podemos graficar los resultados

Código 1: Solución del problema de optimización

```
# Definimos la dinámica del sistema acoplado
def PhiDynamics(t,phi):
    return np.dot(A, phi)

# Definimos función toma como variables las condiciones iniciales de p y devuelve la condición inicial para Phi
def Phi0(p_0):
    X_0 = np.array([1, 0, 1, 0, 0])
    Phi_0 = np.concatenate((X_0,p_0))
    return Phi_0

# Definimos problema de optimización

def objective_function(p_0):
    #Condiciones iniciales
    Phi_0 = Phi0(p_0)

    #Resolver la EDO y devolver Phi(t_f)
    Phi_tf = solve_ivp(PhiDynamics, t_span, Phi_0, t_eval=[t_f], method='RK45') #Phi(t_f)
    p_tf = Phi_tf.y[-4:] #p(t_f)
    X_tf = Phi_tf.y[:4] #(x, dx, y, dy)
    QX_tf = np.dot(Q, X_tf) #Q x_tf

    # Minimización
    error = np.linalg.norm(p_tf - QX_tf)
    return error
```

Código 2: Solución máximo de Pontryagin (PMP)

```
# Iteración inicial para p(0)
p0_initial_guess = np.array([1,2,0,4])

# Resolver el problema de optimización
result = minimize(objective_function, p0_initial_guess)
p0_optimal = result.x

# Función para resolver el sistema acoplado
def solve_coupled_system(p_0):
    # Condiciones iniciales
    Phi_0 = Phi0(p_0)

    # Resolver la EDO
    Phi_solution = solve_ivp(PhiDynamics, t_span, Phi_0, t_eval=np.linspace(0, t_f, 100), method='RK45')
    Phi_solution_tf = solve_ivp(PhiDynamics, t_span, Phi_0, t_eval=[t_f], method='RK45')
    # Obtener resultados
    t = Phi_solution.t
    X_solution = Phi_solution.y[:5]
    X_solution_tf = Phi_solution_tf.y[:5]
    # Calcular los controles óptimos (u(t), v(t)) = (p_2(t), p_4(t))
    u_optimal = Phi_solution.y[6]
    v_optimal = Phi_solution.y[8]

    objetive_value = 0.5*(X_solution_tf[0]**2 + X_solution_tf[1]**2 + X_solution_tf[2]**2 + X_solution_tf[3]**2 + 2*X_solution_tf[0]*X_solution_tf[2]) + integrate.simpson(u_optimal**2 + v_optimal**2, t)
    return t, X_solution, u_optimal, v_optimal, objetive_value

# Resolver el sistema acoplado con p_0 óptimo
t, X_solution, u_optimal, v_optimal, objetive_value = solve_coupled_system(p0_optimal)
```

Se grafican los resultados de la trayectoria óptima y sus respectivos controles óptimos y se presenta el valor objetivo alcanzado $J(u(\cdot))$.

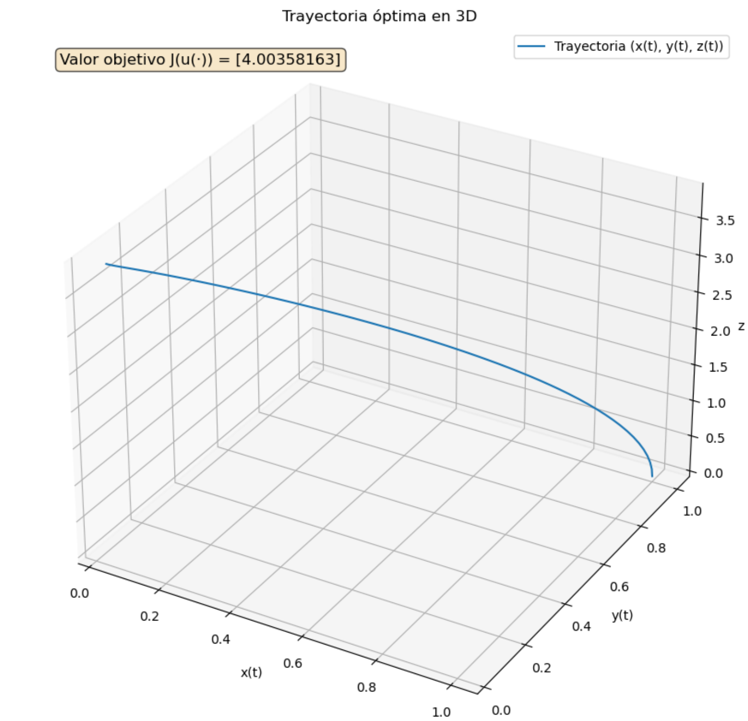

Figura 1: Trayectoria Óptima obtenido del principio del máximo de Pontryagin

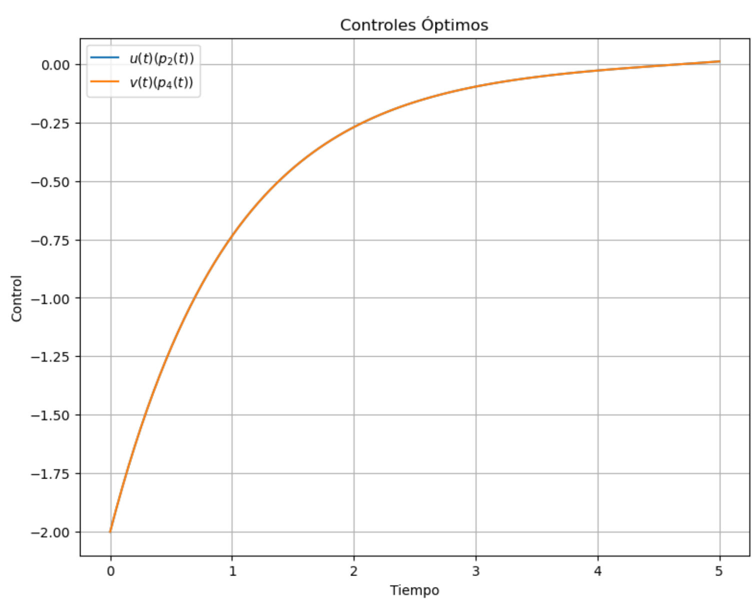

Figura 2: Control óptimo del principio del máximo de Pontryagin

5. Ejercicio 4

Establecemos la ecuación de Riccati del problema para resolverla numéricamente.

Sabemos que $u(\cdot)$ se puede escribir en forma de feedback lineal, pues se cumple la condición de existencia y unicidad trivialmente al ser $U = I$.

El único control óptimo de nuestro problema (PLC) se puede expresar como el feedback lineal

$$u(t) = U^{-1}(t)B(t)^\top E(t)X(t) \text{ c.t.p. } t \in [0, t_f]$$

donde $E(\cdot)$ es la solución sobre $[0, t_f]$ de la ecuación matricial de Riccati:

$$\begin{cases} \dot{E}(t) = W(t) - A(t)^\top E(t) - E(t)^\top A(t) - E(t)^\top B(t)U^{-1}(t)B(t)^\top E(t) \\ E(t_f) = -Q \end{cases}$$

Más aún, $E(\cdot)$ es a valores simétricos y la función valor del problema cumple

$$\text{val(PLC)} = -x_0^\top E(0)x_0$$

Identificando términos de nuestro problema tenemos

$$Q = \frac{1}{2}\begin{bmatrix} 1 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix} \ ; \ U = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \ ; \ W = 0_5$$

Junto a

$$A = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 \end{bmatrix} \ ; \ B = \begin{bmatrix} 0 & 0 \\ 1 & 0 \\ 0 & 0 \\ 0 & 1 \\ 0 & 0 \end{bmatrix}$$

Y entonces la ecuación de Riccati asociada al control óptimo

$$u(t) = B(t)^\top E(t)X(t) \text{ c.t.p. } t \in [0, t_f]$$

Se formula como

$$\begin{cases} \dot{E}(t) = -A^\top E(t) - E(t)^\top A - E(t)^\top BB^\top E(t) \\ E(t_f) = -Q \end{cases}$$

Para ello resolvemos el sistema inverso que sigue
$\hat{E}(t) = E(t_f - t) \implies \dot{\hat{E}}(t) = -\dot{E}(t_f - t) = -\hat{E}(t)$,
con el fin de resolver usando `solve_ivp` con condición inicial $\hat{E}(0) = -Q$.
Se tiene la dinámica

$$\begin{cases} \dot{\hat{E}}(t) = A^\top \hat{E}(t) + \hat{E}(t)^\top A + \hat{E}(t)^\top BB^\top \hat{E}(t) \\ \hat{E}(0) = -Q \end{cases}$$

Código 3: Solución del problema por feedback lineal (FL)

```python

# Función de dinámica de Riccati inversa
def RiccatiDynamics(t, E):
    # Reformar la matriz de 5x5
    E_matrix = np.array(E).reshape(5, 5)
    # Dinámica
    dynamics = np.dot(A.T, E_matrix) + np.dot(E_matrix.T, A) + np.dot(E_matrix, np.
     ↪ dot(np.dot(B,B.T),E_matrix))
    return dynamics.ravel() # Aplanar la matriz resultante

# Función hat_E(t) solución de la ecuación de Riccati inversa con dense_output, se usará
     ↪ para definir el control feedback.
hat_E = solve_ivp(RiccatiDynamics, t_span, Q_ravel, method='RK45', t_eval = np.
     ↪ linspace(0,t_f,1001), dense_output = True).sol

# Funcion E(t) solución de la ecuación de Riccati inversa como función lambda
def E_matrix(t):
    E_matrix = hat_E(t_f-t).reshape(5,5)
    return E_matrix

# Definimos la dinámica del sistema controlado por feedback lineal
def DynamicsRFeedback(t, X):
    return np.dot(A,X) + np.dot(B,np.dot(B.T, np.dot(E_matrix(t), X)))

# Entregamos la solución del sistema
X_solution_feedback = solve_ivp(DynamicsRFeedback, t_span, X_0, method='RK45',
     ↪ t_eval = np.linspace(0,t_f,1001))
objetive_value_feedback = -np.dot(X_0, np.dot(E_matrix(0),X_0)) #-X_0^T E(0) X_0
```

A partir de lo anterior, se entregan las siguientes soluciones para la ecuación, y además, se imprime el valor de $J(u(\cdot)) = -X_0^T E(0) X_0$ entregando un valor muy similar a la solución del problema vía (PMP), siendo coherente así con la teoría y las propiedades de la función matricial $E(t)$

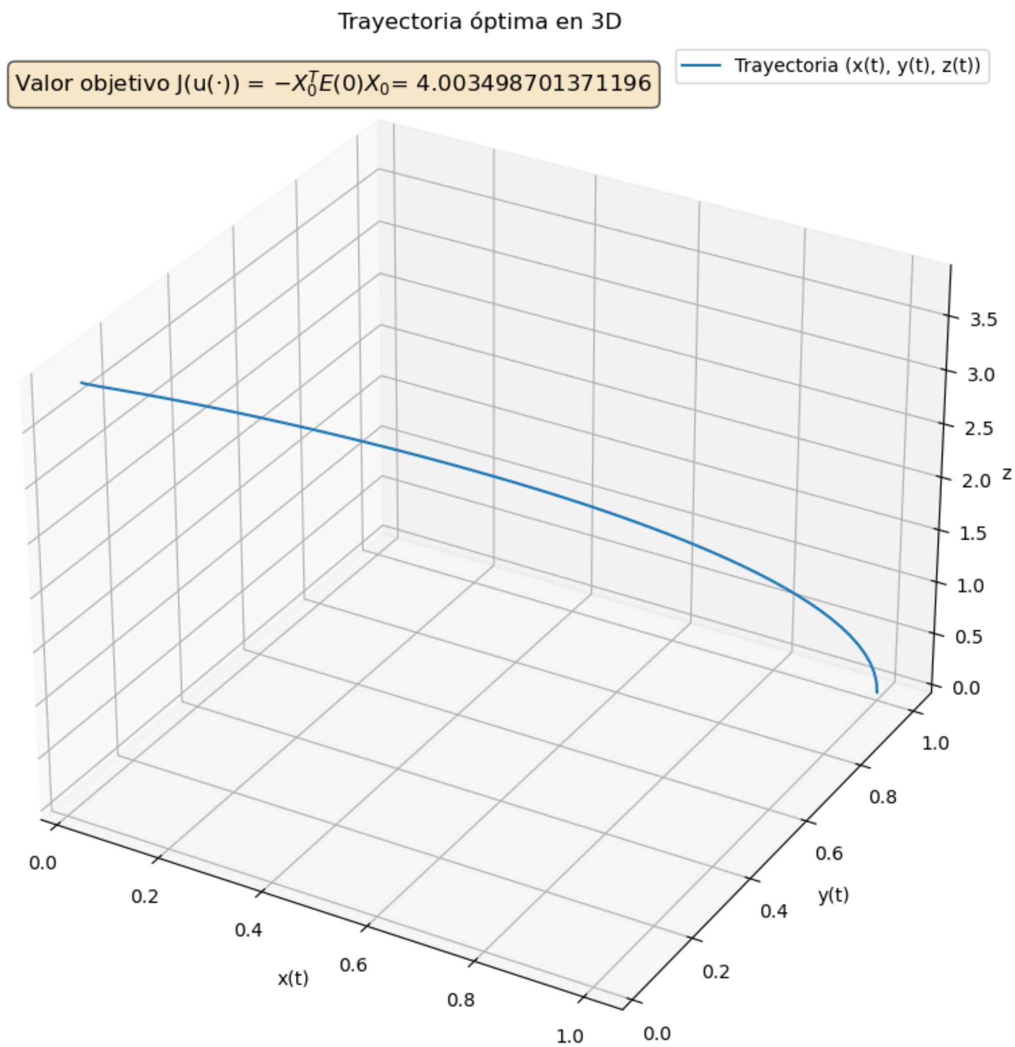

Figura 3: Trayectoria óptima feedback mediante ecuación de Riccati

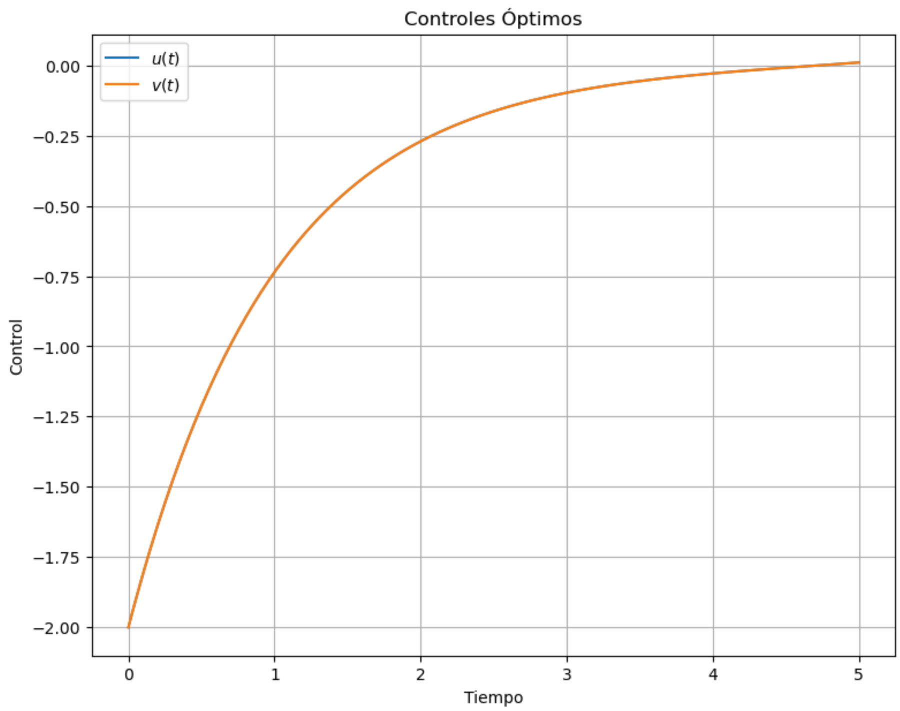

Figura 4: Control óptimo feedback mediante ecuación de Riccati

Comparativas: Notamos a simple vista que ambas soluciones son casi identicas, para la solución encontrada mediante el principio del máximo de Pontryagin, se alcanza un valor objetivo de $J(u(\cdot)) = 4.003579$ que es ligeramente mayor al valor objetivo alcanzado mediante un control de tipo feedback obtenido de la ecuación de Riccati que alcanza un valor objetivo $J(u(\cdot)) = -X_0^T E(0) X_0 = 4.0034987$.

Con el fin de estimar la diferencia de las soluciones, computamos sus diferencias en la norma de $L^2([0, t_f], \mathbb{R}^3)$ para el caso de las trayectorias $(x(t), y(t), z(t))$, y en la norma de $L^2([0, t_f], \mathbb{R}^2)$ para los controles $(u(t), y(t))$ que son irrestrictos

Denotamos las soluciones asociadas al principio del máximo de Pontryagin por (PMP), y a las soluciones asociadas a la ecuación de Riccati de tipo feedback lineal por (FL), se calculan las cantidades

$$||X_{PMP} - X_{FL}||_{L^2} = \sqrt{\int_0^{t_f} (x_{PMP}(t) - x_{FL}(t))^2 + (y_{PMP}(t) - y_{FL}(t))^2 + (z_{PMP}(t) - z_{FL}(t))^2 dt}$$

$$||u_{PMP} - u_{FL}||_{L^2} = \sqrt{\int_0^{t_f} (u_{PMP}(t) - u_{FL}(t))^2 + (v_{PMP}(t) - v_{FL}(t))^2 dt}$$

Para ello se escribe el siguiente código

Código 4: Error L2 de las trayectorias y controles

```python
# Calculamos error L2 de las trayectorias
error_trayectorias_L2 = np.sqrt(integrate.simpson((x - x_feedback)**2 + (y - y_feedback)**2 + (z - z_feedback)**2, t))
print(r'El error L2 de las trayectorias obtenidas ||X_{PMP}(t) - X_{FL}||_L2', error_trayectorias_L2)

# Calculamos error L2 de los controles
error_control_L2 = np.sqrt(integrate.simpson((u_optimal - u_feedback)**2 + (v_optimal - v_feedback)**2, t))
print(r'El error L2 de los controles obtenidos ||u_{PMP}(t) - u_{FL}||_L2', error_control_L2)
```

De donde se obtiene que

$$||X_{PMP} - X_{FL}||_{L^2} = 0.000161217$$

$$||u_{PMP} - u_{FL}||_{L^2} = 0.000115025$$

A partir de esto confirmamos que las soluciones obtenidas mediante el principio del máximo de Pontryagin y las soluciones de tipo feedback obtenidas mediante la ecuación de Riccati resultan ser casi idénticas en el sentido de L^2.

6. Ejercicio 5

Resolvemos el problema lineal cuadrático haciendo uso de BOCOP bajo las mismas condiciones iniciales, y el mismo valor de $t_f = 5$, para su resolución, debemos formular el problema como un problema de tipo Mayer, es decir, debemos hacer un cambio en la dinámica para expresar la cantidad de tipo Lagrangiano

$$\int_0^{t_f} u(t)^2 + v(t)^2 dt$$

Como una condición que dependa de las condiciones de borde $\Phi(t_f)$, consideramos entonces

$$\Phi(t) = \int_0^{t} u(t)^2 + v(t)^2 dt$$

Entonces Φ cumple la dinámica, $\dot{\Phi}(t) = u(t)^2 + v(t)^2$, que por construcción cumple la condición inicial $\Phi(0) = 0$

Considerando esta nueva función, el problema se puede escribir

$$\min_{(u(\cdot),v(\cdot))} \frac{1}{2}(\dot{x}^2 + \dot{y}^2 + x^2 + 2xt + y^2)(t_f) + \Phi(t_f)$$

Sujeto a la dinámica

$$\frac{d}{dt}\begin{bmatrix} x(t) \\ \dot{x}(t) \\ y(t) \\ \dot{y}(t) \\ z(t) \\ \Phi(t) \end{bmatrix} = \begin{bmatrix} \dot{x}(t) \\ x(t) + u(t) \\ \dot{y}(t) \\ y(t) + v(t) \\ x(t) + y(t) \\ u(t)^2 + v(t)^2 \end{bmatrix} \quad ; \quad \begin{bmatrix} x(0) \\ \dot{x}(0) \\ y(0) \\ \dot{y}(0) \\ z(0) \\ \Phi(0) \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \\ 1 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

Se procede a escribir el problema en BOCOP inicializando la cantidad de variables de estado como 6, las variables de control como 2, tiempo final como 5 y condiciones de borde para las variables de estado, junto a las etiquetas de las variables dentro del GUI de BOCOP, seguido a esto, se escriben los archivos *criterion.tpp, dynamics.tpp, boundarycond.tpp*.

Una vez se obtienen los resultados, se exportan desde BOCOP hacia Python, y se procede a graficar trayectorias y realizar comparativas con respecto de los resultados anteriores.

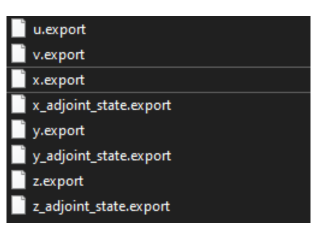

Figura 5: Exporte de datos de BOCOP

Para definir el criterio de minimización se escribe

Código 5: criterion.tpp

```
criterion = (0.5 * ((final_state[1] * final_state[1]) + (final_state[3] * final_state[3]) + (
  ↪ final_state[0] * final_state[0]) + (2 * final_state[0] * final_state[2]) + (final_state[2]
  ↪ * final_state[2]))) + final_state[5];
```

Para definir la dinámica del sistema se escribe

Código 6: dynamics.tpp

```
Tdouble x = state[0];
Tdouble dx = state[1];
Tdouble y = state[2];
Tdouble dy = state[3];
Tdouble z = state[4];
Tdouble phi = state[5];
Tdouble control_u = control[0];
Tdouble control_v = control[1];

state_dynamics[0] = dx;
state_dynamics[1] = x + control_u;
state_dynamics[2] = dy;
state_dynamics[3] = y + control_v;
state_dynamics[4] = x + y;
state_dynamics[5] = (control_u * control_u) + (control_v * control_v);
```

Y por último, para definir las condiciones iniciales del sistema se escribe

Código 7: dynamics.tpp

```
boundary_conditions[0] = initial_state[0];
boundary_conditions[1] = initial_state[1];
boundary_conditions[2] = initial_state[2];
boundary_conditions[3] = initial_state[3];
boundary_conditions[4] = initial_state[4];
boundary_conditions[5] = initial_state[5];
```

Luego de escribir el problema, se resuelve mediante distintos métodos de discretización, todos entregan resultados idénticos para la práctica

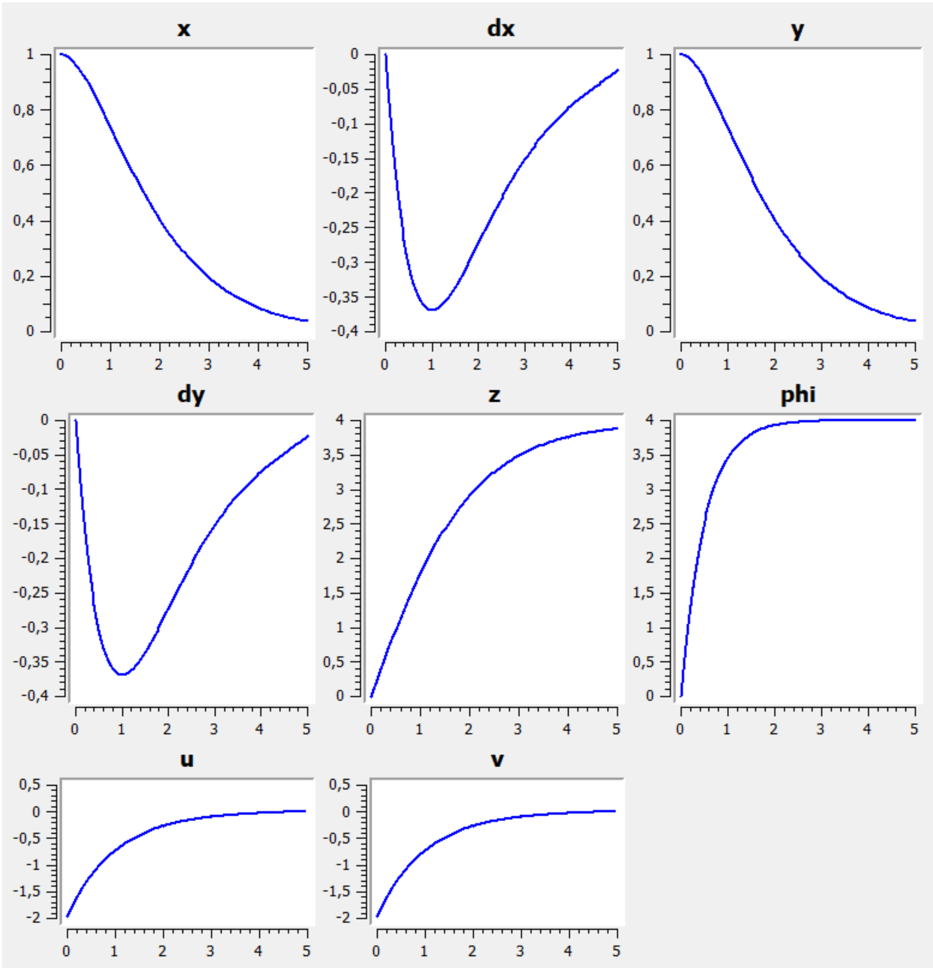

Figura 6: Gráficos obtenidos resolviendo directamente en BOCOP

Dicha solución alcanza un valor objetivo $J(u(\cdot)) = 4.00348$ junto con errores de restricción del estado adjunto despreciables

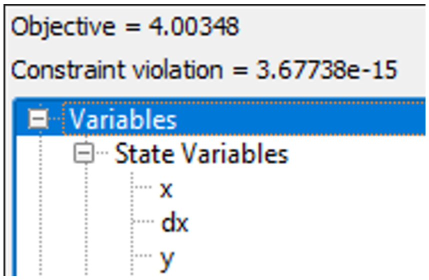

Figura 7: Valor objetivo obtenido a través de BOCOP

Comparativas: Se exportan los resultados obtenidos por BOCOP, tenemos primero para el caso de las trayectorias óptimas:

Código 8: Exporte de las trayectorias de BOCOP

```
# Importe de la trayectoria óptima desde BOCOP
# Solución en x

x_bocop = [
    1, 0.998765, 0.995222, 0.989603, 0.982118, 0.972967, 0.962331, 0.950378, 0.937263, 0.923131,
    0.908112, 0.892329, 0.875892, 0.858905, 0.841461, 0.823646, 0.805539, 0.787211, 0.768728, 0.75015,
    0.73153, 0.712918, 0.694357, 0.675888, 0.657546, 0.639363, 0.621368, 0.603585, 0.586038, 0.568747,
    0.551727, 0.534995, 0.518562, 0.50244, 0.486637, 0.471161, 0.456017, 0.441211, 0.426745, 0.412621,
    0.398840, 0.385403, 0.372308, 0.359554, 0.347140, 0.335061, 0.323315, 0.311898, 0.300806, 0.290034,
    0.279577, 0.269430, 0.259587, 0.250043, 0.240792, 0.231827, 0.223144, 0.214735, 0.206595, 0.198716,
    0.191094, 0.183722, 0.176593, 0.169701, 0.163041, 0.156606, 0.150391, 0.144389, 0.138594, 0.133002,
    0.127607, 0.122404, 0.117386, 0.112550, 0.107890, 0.103402, 0.0990814, 0.0949233, 0.0909239, 0.0870792,
    0.0833855, 0.0798391, 0.0764368, 0.0731753, 0.0700518, 0.0670636, 0.0642082, 0.0614834, 0.0588872,
    0.0564180, 0.0540742, 0.0518548, 0.0497587, 0.0477855, 0.0459348, 0.0442066, 0.0426014, 0.0411196, 0.0397625,
    0.0385314
]
```

```
# Solución en y

y_bocop = [
    1, 0.998765, 0.995222, 0.989603, 0.982118, 0.972967, 0.962331, 0.950378, 0.937263, 0.923131,
    0.908112, 0.892329, 0.875892, 0.858905, 0.841461, 0.823646, 0.805539, 0.787211, 0.768728, 0.75015,
    0.73153, 0.712918, 0.694357, 0.675888, 0.657546, 0.639363, 0.621368, 0.603585, 0.586038, 0.568747,
    0.551727, 0.534995, 0.518562, 0.50244, 0.486637, 0.471161, 0.456017, 0.441211, 0.426745, 0.412621,
    0.398840, 0.385403, 0.372308, 0.359554, 0.347140, 0.335061, 0.323315, 0.311898, 0.300806, 0.290034,
    0.279577, 0.269430, 0.259587, 0.250043, 0.240792, 0.231827, 0.223144, 0.214735, 0.206595, 0.198716,
    0.191094, 0.183722, 0.176593, 0.169701, 0.163041, 0.156606, 0.150391, 0.144389, 0.138594, 0.133002,
    0.127607, 0.122404, 0.117386, 0.112550, 0.107890, 0.103402, 0.0990814, 0.0949233, 0.0909239, 0.0870792,
    0.0833855, 0.0798391, 0.0764368, 0.0731753, 0.0700518, 0.0670636, 0.0642082, 0.0614834, 0.0588872,
    0.0564180, 0.0540742, 0.0518548, 0.0497587, 0.0477855, 0.0459348, 0.0442066, 0.0426014, 0.0411196, 0.0397625,
    0.0385314
]

# Solución en z

z_bocop = [
    0, 0.100968, 0.201693, 0.301953, 0.40155, 0.500305, 0.598059, 0.694671, 0.790015, 0.883983,
    0.976477, 1.06741, 1.15672, 1.24434, 1.33022, 1.41432, 1.49661, 1.57705, 1.65563, 1.73235,
    1.80718, 1.88013, 1.9512, 2.02041, 2.08775, 2.15325, 2.21692, 2.27879, 2.33887, 2.39719,
    2.45377, 2.50866, 2.56186, 2.61343, 2.66338, 2.71175, 2.75857, 2.80388, 2.84772, 2.89011,
    2.93109, 2.97069, 3.00896, 3.04592, 3.08161, 3.11606, 3.14931, 3.18139, 3.21233, 3.24216,
    3.27093, 3.29866, 3.32537, 3.35111, 3.37589, 3.39976, 3.42274, 3.44485, 3.46613, 3.4866,
    3.50628, 3.52521, 3.5434, 3.56089, 3.5777, 3.59384, 3.60934, 3.62423, 3.63852, 3.65223,
    3.66539, 3.67802, 3.69013, 3.70174, 3.71287, 3.72354, 3.73377, 3.74356, 3.75295, 3.76194,
    3.77054, 3.77879, 3.78668, 3.79423, 3.80147, 3.80839, 3.81502, 3.82137, 3.82744, 3.83327,
    3.83885, 3.84419, 3.84933, 3.85425, 3.85898, 3.86353, 3.86792, 3.87214, 3.87623, 3.88018
]
```

A partir de estos datos, procedemos a graficar la trayectoria descrita en Python y computamos los errores que tiene con respecto a las soluciones obtenidas mediante Pontryagin y Feedback lineal en norma L^2

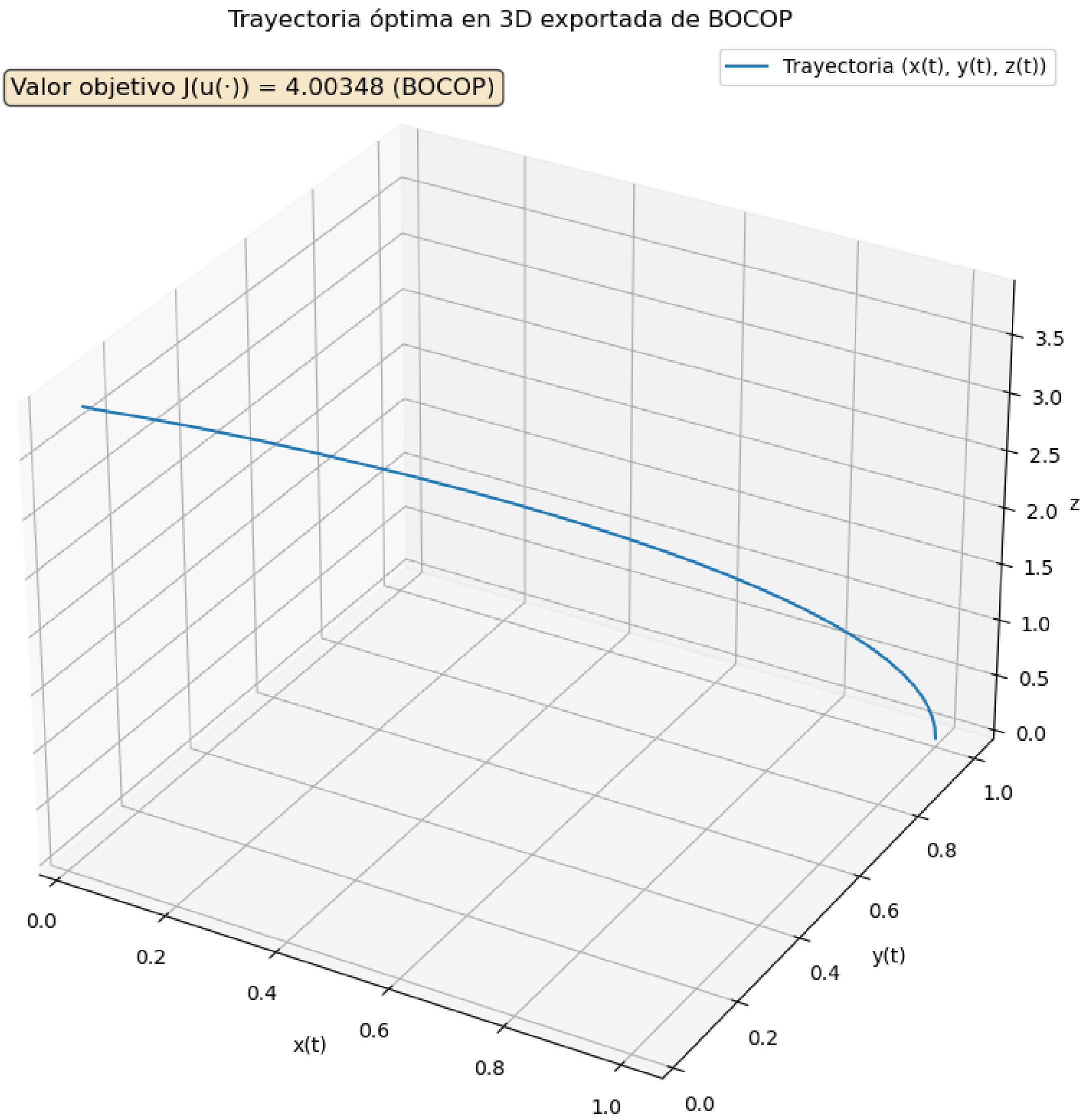

Figura 8: Trayectoria simulada en Python con datos exportados de BOCOP

Computamos los errores escribiendo el código

Código 9: Error entre resultados de BOCOP con Python para trayectorias

```
# Calculamos error L2 de las trayectorias de BOCOP con la solución obtenida del máximo de Pontryagin
error_trayectorias_L2_bocop_pmp = np.sqrt(integrate.simpson((x - x_bocop)**2 + (y - y_bocop)**2 + (z - z_bocop)**2, t))
# Calculamos error L2 de las trayectorias de BOCOP con la solución obtenida del control feedback
error_trayectorias_L2_bocop_fl = np.sqrt(integrate.simpson((x_feedback - x_bocop)**2 + (y_feedback - y_bocop)**2 + (z_feedback - z_bocop)**2, t))
```

De donde se obtiene que

$$||X_{BOCOP} - X_{PMP}||_{L^2} = 0.00018232$$

$$||X_{BOCOP} - X_{FL}||_{L^2} = 0.000148207$$

A partir de esto, confirmamos la similitud entre las soluciones dadas por BOCOP y por aquellas dadas en Python usando el principio de máximo de Pontryagin y las ecuaciones de Riccati, observando que el error toma valores del orden de 10^{-4} en la norma L^2

Realizamos el mismo procedimiento para los controles óptimos $(u(t), v(t))$, exportando los datos de BOCOP se tiene

Código 10: Exporte de los controles óptimos de BOCOP

```

# Importe de controles óptimos desde BOCOP
u_bocop = [
    -1.95169, -1.85555, -1.76415, -1.67724, -1.59462, -1.51606, -1.44137, -1.37036, -1.30284, -1.23865,
    -1.17762, -1.11959, -1.06442, -1.01196, -0.962088, -0.914669, -0.869583, -0.826716, -0.785958, -0.747206,
    -0.71036, -0.675326, -0.642015, -0.610342, -0.580226, -0.551591, -0.524363, -0.498472, -0.473854, -0.450444,
    -0.428184, -0.407016, -0.386886, -0.367744, -0.349539, -0.332227, -0.315762, -0.300103, -0.285209, -0.271043,
    -0.257568, -0.244751, -0.232558, -0.220959, -0.209923, -0.199422, -0.189431, -0.179923, -0.170874, -0.16226,
    -0.154061, -0.146255, -0.138822, -0.131743, -0.125, -0.118576, -0.112455, -0.106621, -0.101058, -0.0957539,
    -0.0906936, -0.0858647, -0.081255, -0.0768525, -0.0726461, -0.068625, -0.064779, -0.0610983, -0.0575734,
    -0.0541955, -0.0509558, -0.0478461, -0.0448585, -0.0419854, -0.0392193, -0.0365533, -0.0339806, -0.0314946,
    -0.0290889, -0.0267574, -0.0244942, -0.0222935, -0.0201497, -0.0180573, -0.0160109, -0.0140054, -0.0120357,
    -0.0100966, -0.0081833, -0.00629088, -0.0044145, -0.00254939, -0.000690783, 0.00116606, 0.00302588, 0.00489342,
    0.00677345, 0.00867075, 0.0105902, 0.0105902
]

v_bocop = [
    -1.95169, -1.85555, -1.76415, -1.67724, -1.59462, -1.51606, -1.44137, -1.37036, -1.30284, -1.23865,
    -1.17762, -1.11959, -1.06442, -1.01196, -0.962088, -0.914669, -0.869583, -0.826716, -0.785958, -0.747206,
    -0.71036, -0.675326, -0.642015, -0.610342, -0.580226, -0.551591, -0.524363, -0.498472, -0.473854, -0.450444,
    -0.428184, -0.407016, -0.386886, -0.367744, -0.349539, -0.332227, -0.315762, -0.300103, -0.285209, -0.271043,
    -0.257568, -0.244751, -0.232558, -0.220959, -0.209923, -0.199422, -0.189431, -0.179923, -0.170874, -0.16226,
    -0.154061, -0.146255, -0.138822, -0.131743, -0.125, -0.118576, -0.112455, -0.106621, -0.101058, -0.0957539,
    -0.0906936, -0.0858647, -0.081255, -0.0768525, -0.0726461, -0.068625, -0.064779, -0.0610983, -0.0575734,
    -0.0541955, -0.0509558, -0.0478461, -0.0448585, -0.0419854, -0.0392193, -0.0365533, -0.0339806, -0.0314946,
```

```
    -0.0290889, -0.0267574, -0.0244942, -0.0222935, -0.0201497, -0.0180573, -0.0160109, -0.0140054, -0.0120357,
    -0.0100966, -0.0081833, -0.00629088, -0.0044145, -0.00254939, -0.000690783, 0.00116606, 0.00302588, 0.00489342,
    0.00677345, 0.00867075, 0.0105902, 0.0105902
]
```

A partir de estos datos, graficamos los controles y computamos los errores en el sentido de L^2 con las soluciones previamente obtenidas, donde el código sigue la misma estructura que los anteriores, se obtiene

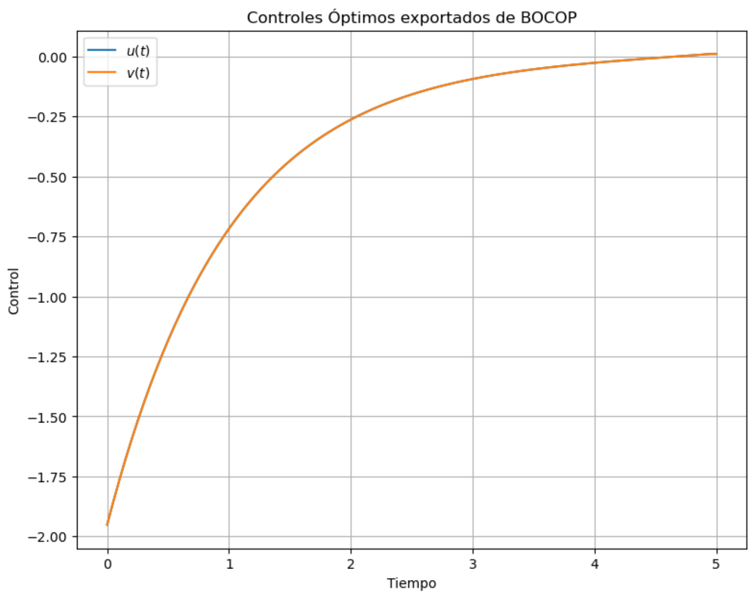

Figura 9: Controles óptimos en Python con datos exportados de BOCOP

Junto a los errores en norma L^2 con respecto de los resultados anteriores

$$||u_{BOCOP} - u_{PMP}||_{L^2} = 0.049735971$$

$$||u_{BOCOP} - u_{FL}||_{L^2} = 0.0497403162$$

Notamos que los valores objetivo obtenidos en cada uno de los problemas anteriores tienen diferencias del orden de 10^{-4}, y diferencias del orden de 10^{-2} en la norma L^2 para las trayectorias óptimas y controles óptimos respectivos, lo cual nos permite concluir no sólo sobre la acertividad/utilidad de los tres métodos, sino que además, son coherentes entre sí.