Alumno: Brian Alex Fuentes Acuña.

Padrón: 101785.

Clase 7: Estabilidad

Introducción

En control clásico trabajamos con los modelos de entrada-salida en sistemas lineales, y hasta ahora de ellos conocemos la llamada estabilidad externa o estabilidad entrada-salida.

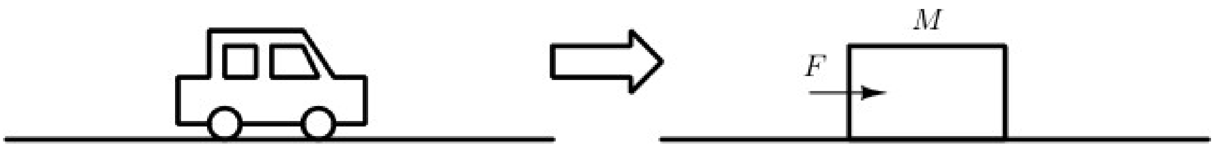
En ese contexto decimos que un sistema es estable si ante cualquier excitación de entrada acotada que apliquemos se generará una respuesta de la salida que deberá ser acotada.

¿Por qué nos importa garantizar la estabilidad en este sentido? Se nos ocurren dos situaciones en las que hay que ser cuidadosos:

En sistemas bajo análisis, es importante conocer su estabilidad porque nos permite establecer una estrategia de control acorde y también un modelo adecuado.

Vayamos a un ejemplo simple y concreto:

Aleksandr Lyapunov, o como le dicen los amigos, Ale, se compró un Porsche y quiso salir a pasear para estrenarlo. Su objetivo era viajar desde su casa y visitar a su viejo amigo J.P. LaSalle, Joe. Ale, como buen físico, decidió hacer un modelo de su problema y con la simplicidad en mente propuso que su auto era equivalente a una masa puntual:



Modelo de auto

El modelo responde a la Ley de Newton $F = Ma$. Le interesaba la posición del auto, $y = x$ variable de interés y como acción de control o entrada la fuerza ejercida del motor sobre la masa $u = f$.

Con estas consideraciones llega al modelo:

$$\ddot{y} = \frac{u}{M}$$

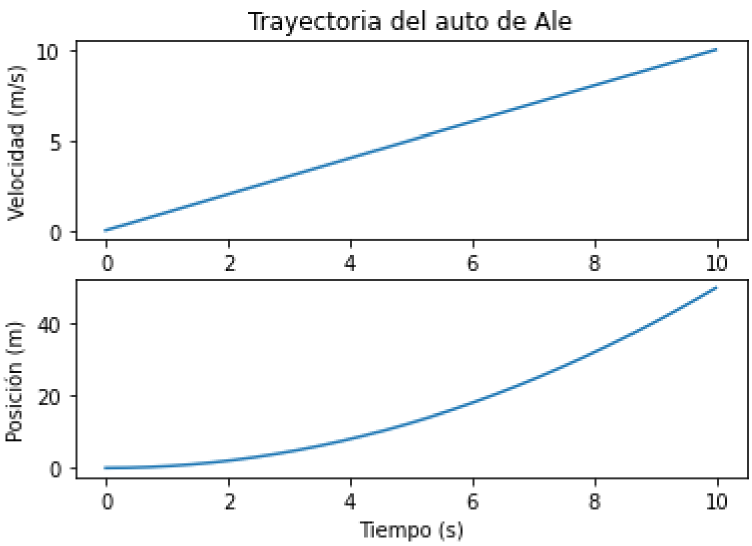
Con transferencia:

$$\frac{Y(s)}{U(s)} = \frac{1/M}{s^2}$$

Y una de las primeras preguntas que se hace es si el sistema es estable. ¿Qué sucede con la respuesta a un escalón de fuerza de amplitud 1 N?

```
!pip install control
import control as ctrl
import matplotlib.pyplot as plt
import numpy as np

M = 1
t_ini = 0
t_fin = 10
t_paso = 0.01
t = np.arange(t_ini, t_fin, t_paso)
u = 1.0*np.ones(len(t))

auto = ctrl.tf([1/M], [1, 0, 0])
[t, y, estado] = ctrl.forced_response(auto, T=t, U=u, return_x=True)
plt.subplot(212)
plt.plot(t, y)
plt.xlabel('Tiempo (s)')
plt.ylabel('Posición (m)')
velocidad = estado[0]
plt.subplot(211)
plt.ylabel('Velocidad (m/s)')
plt.plot(t, velocidad)
plt.title('Trayectoria del auto de Ale')
```

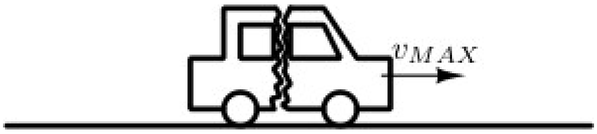
```
Requirement already satisfied: control in /usr/local/lib/python3.7/dist-packages (0.9
Requirement already satisfied: scipy in /usr/local/lib/python3.7/dist-packages (from
Requirement already satisfied: matplotlib in /usr/local/lib/python3.7/dist-packages (
Requirement already satisfied: numpy in /usr/local/lib/python3.7/dist-packages (from
Requirement already satisfied: pyparsing!=2.0.4,!=2.1.2,!=2.1.6,>=2.0.1 in /usr/local
Requirement already satisfied: python-dateutil>=2.1 in /usr/local/lib/python3.7/dist-
Requirement already satisfied: cycler>=0.10 in /usr/local/lib/python3.7/dist-packages
Requirement already satisfied: kiwisolver>=1.0.1 in /usr/local/lib/python3.7/dist-pac
Requirement already satisfied: six in /usr/local/lib/python3.7/dist-packages (from cy
/usr/local/lib/python3.7/dist-packages/control/timeresp.py:294: UserWarning: return_x
  "return_x specified for a transfer function system. Internal "
Text(0.5, 1.0, 'Trayectoria del auto de Ale')
```



Notamos que la velocidad crece linealmente y la posición cuadráticamente. La primera reacción de Ale fue de felicidad y despreocupación, pero lentamente empezó a sospechar.

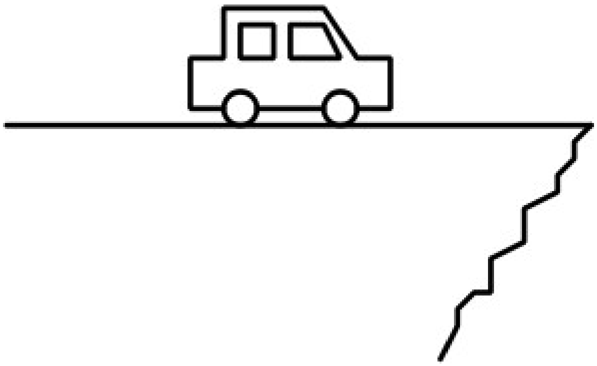
¿Qué puede salir mal? ¿Por qué llamaría inestable a este sistema? Se le ocurrieron 3 situaciones:

1. Que la velocidad se vuelva muy cercana a c = 300000 km/s y el modelo físico no valga más.
2. Que el Porsche no soporte más de 400 km/h y se rompa.



Velocidad máxima del auto

3. Que el auto llegue a un barranco y como no puede convertirse en avión, caiga y explote.



El auto llega a un barranco

Lo que queremos mostrar con este ejemplo es que un sistema inestable a lazo abierto puede traernos varios dolores de cabeza.

Asimismo, el segundo punto importante de la estabilidad es en el contexto del control aplicado a un sistema. Necesitamos garantizar la estabilidad del sistema junto a su controlador. Es decir, el controlador no debe desestabilizar al sistema sino que parte de su trabajo es mantenerlo dentro de las condiciones de trabajo esperadas o deseadas.

En resumen:

Decimos que un sistema es BIBO estable, si para toda entrada acotada (bounded input) se obtiene una respuesta de salida acotada (bounded output).

Esta condición se da para sistemas lineales e invariantes en el tiempo si y solo si la respuesta al impulso $g(t)$ del sistema es absolutamente integrable:

$$\int_0^{\infty} |g(t)|dt \leq M < \infty$$

O absolutamente sumable en sistemas de tiempo discreto:

$$\sum_{k=0}^{\infty} |g(k)| \leq M < \infty$$

Equivalentemente, se puede llegar a la conclusión que todos los p_i, polos de la función transferencia $H(s) = Y(s)/U(s)$, deber cumplir:

$$\mathrm{Re}(p_i) < 0$$

para sistemas LTI de tiempo continuo. O

$$|p_i| < 1$$

para sistemas LTI de tiempo discreto.

El modelo de auto de Ale, tiene dos polos en el origen $p_1 = p_2 = 0$ y por lo tanto, no es BIBO estable.

Para pensar: ¿Como crees que estas condiciones se extienden al caso MIMO?

¿Es la única definición de estabilidad que existe?

No, no lo es. Con la introducción de los modelos en variables de estado, que contienen una descripción de la dinámica interna de un sistema, surge la necesidad de evaluar la estabilidad no solo desde el punto de vista de la o las entradas de excitación del sistema, sino también de la respuesta no forzada, libre o también conocida como la respuesta a las condiciones iniciales.

Estabilidad interna o de Lyapunov

En la clase teórica vimos que Lyapunov definió su idea de estabilidad alrededor de los puntos de equilibrio de un sistema. ¿Qué son estos puntos de equilibrio?

Para un sistema:

$$\dot{x} = f(x, u)$$

son todos aquellos puntos del espacio de estados que cumplen, que dada $u = 0$,

$$\dot{x} = f(x_e, 0) = 0.$$

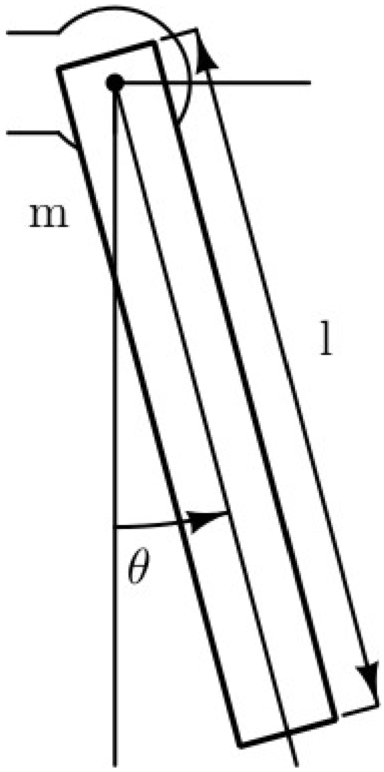
Y bajo esa condición, si $x(0) = x_e$ hace que $x(t) = x_e$ para todo $t \geq 0$.

Luego Lyapunov dijo:

Un punto de equilibrio x_e del sistema $\dot{x} = f(x, 0)$ es estable si para cualquier $\epsilon > 0$ existe un valor $\delta(t_0, \epsilon)$ tal que si $||x(t_0) - x_e|| < \delta$ entonces $||x(t) - x_e|| < \epsilon$ para todo $t > t_0$. Si el δ no depende del instante inicial t_0, el sistema es uniformemente estable.

Ejemplo: Péndulo rígido

Estudiemos el ejemplo más famoso de todos. Es un péndulo sujeto de un punto fijo y que puede girar 360 grados alrededor de ese punto. Vamos a considerar un parámetro γ asociado al rozamiento en el punto de fijación.



Péndulo simple

$$J\ddot{\theta} = -mgsen(\theta)\frac{l}{2} - \gamma\dot{\theta} + u$$

con

$$J = \frac{1}{3}ml^2$$

Las ecuaciones de este sistema son:

$$\dot{x} = \begin{bmatrix} x_2 \\ -\frac{3g}{2l}\sin(x_1) - \frac{3\gamma}{ml^2}x_2 + \frac{3}{ml^2}u \end{bmatrix}$$

Donde:

- $x = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} \theta \\ \dot{\theta} \end{bmatrix}$
- $g \simeq$ 10 m/s^2 es la aceleración de la gravedad
- l es el largo del péndulo
- m es la masa del péndulo
- u es un torque de entrada de perturbación

Ejercicio: Hallar los puntos de equilibrio de este sistema.

Deberías llegar a que los puntos de equilibrio son:

$$x_{ek} = \begin{bmatrix} k\pi \\ 0 \end{bmatrix}$$

con $k \in \mathbb{Z}$.

RTA:

Para encontrar los puntos de equilibrio como se muestra arriba se plantea:

$$\dot{x}_1 = x_2 \Rightarrow x_{2e} = 0$$
$$\dot{x}_2 = -\frac{3g}{2l}\sin(x_1) - \frac{3\gamma}{ml^2}(x_{2e} = 0) + \frac{3}{ml^2}.0$$

entonces se obtiene:

$$-\frac{3g}{2l}\sin(x_{1e}) = 0$$

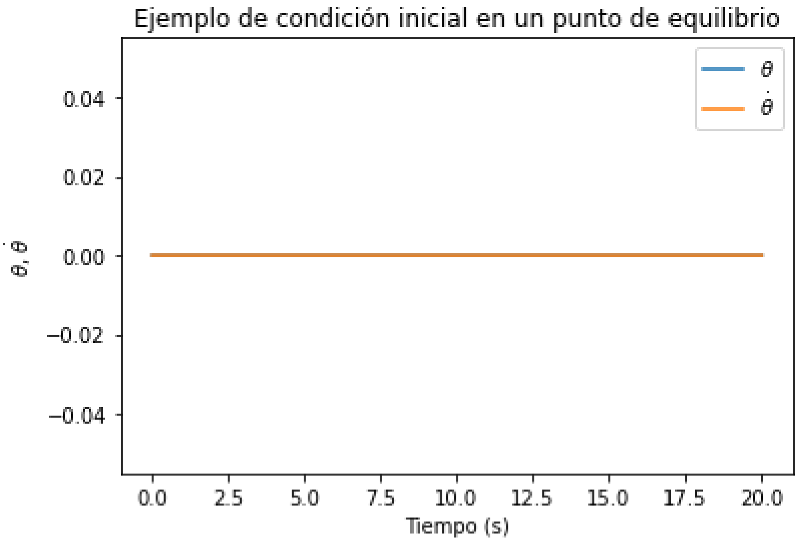
para que se cumpla la igualdad $x_{1e} = k\pi$ con $k \in \mathbb{Z}$.

Para pensar: ¿Qué interpretación le das al $x_{e0} = [0\,0]^T$ y $x_{e1} = [\pi\,0]^T$? ¿Y para los otros valores de k?

Simulemos lo que pasa cuando el péndulo arranca en la condición inicial $x_{e0} = [0\,0]^T$:

```
import control as ctrl
import numpy as np
import scipy as sp
import sympy as symb
import matplotlib.pyplot as plt
from numpy import pi

l = 0.5 # m
m = 0.24 # kg
g = 10 # m/s^2
gamma = 0.01 # Nms

tiempo_inicial = 0
tiempo_final = 20
time_span = (tiempo_inicial, tiempo_final)
paso_tiempo = 0.001
tiempo = np.arange(tiempo_inicial, tiempo_final, paso_tiempo)

def u_function(t):
    u = 0
    return u

def f_pendulo(t, x, l=l, m=m, g=g, gamma=gamma):
    u = u_function(t)
    x1 = x[0] # x1: theta(t)
    x2 = x[1] # x2: dtheta_dt(t)
    dx1_dt = x2
    dx2_dt = -3*g/(2*l)*np.sin(x1) - 3*gamma/(m*l**2)*x2 + 3/(m*l**2)*u
    return [dx1_dt, dx2_dt]

x0 = [0, 0]

respuesta_pendulo = sp.integrate.solve_ivp(f_pendulo, time_span, x0, t_eval=tiempo)

plt.plot(tiempo, respuesta_pendulo.y[0], label=r'$\theta$')
plt.plot(tiempo, respuesta_pendulo.y[1], label=r'$\dot{\theta}$')
plt.title('Ejemplo de condición inicial en un punto de equilibrio')
plt.legend()
```

```
plt.ylabel(r'$\theta$, $\dot{\theta}$')
plt.xlabel('Tiempo (s)')
plt.show()
```



Notamos que el sistema se queda en el punto de equilibrio. O por lo menos pasan 20 segundos y el sistema no evoluciona alejandose de él. Este es el resultado esperado.

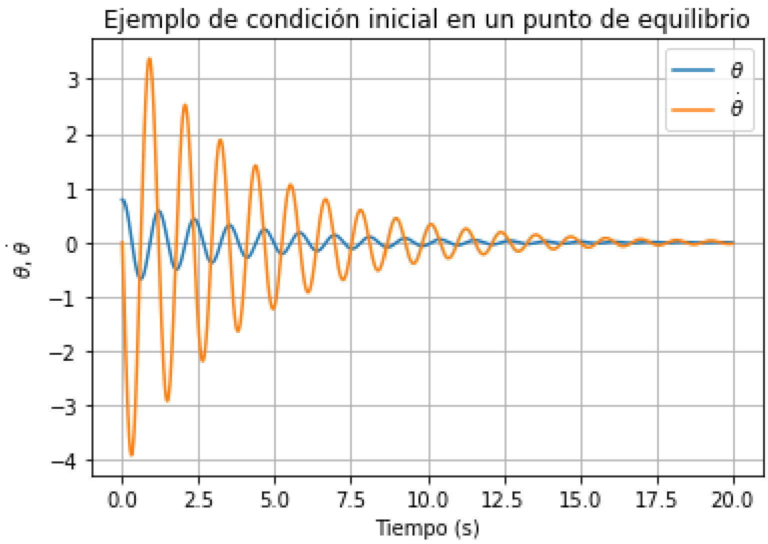
Esta respuesta tiene sentido físico porque el punto $\theta = 0$ rad y $\dot{\theta} = 0$ rad/s del espacio de estados corresponde al péndulo colgando en la posición inferior y sin ninguna velocidad angular. Tiene sentido que se quede quieto.

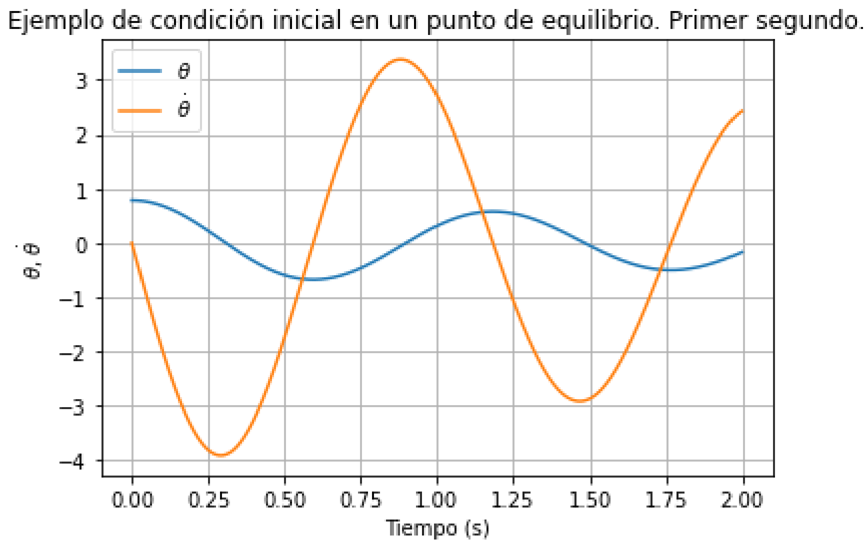
¿Qué pasará ahora si lo corremos un poco de ese punto de equilibrio? Probemos correrlo 45 grados en θ.

Sería que arranque en $x_0 = [\frac{\pi}{4}\ 0]^T$:

```
x0 = [pi/4, 0]

respuesta_pendulo = sp.integrate.solve_ivp(f_pendulo, time_span, x0, t_eval=tiempo)

plt.plot(tiempo, respuesta_pendulo.y[0], label=r'$\theta$')
plt.plot(tiempo, respuesta_pendulo.y[1], label=r'$\dot{\theta}$')
plt.title('Ejemplo de condición inicial en un punto de equilibrio')
plt.legend()
plt.grid()
plt.ylabel(r'$\theta$, $\dot{\theta}$')
plt.xlabel('Tiempo (s)')
plt.show()

idx_primeros_segundos = np.searchsorted(tiempo, 2)
plt.plot(tiempo[0:idx_primeros_segundos], respuesta_pendulo.y[0][0:idx_primeros_segundos],
plt.plot(tiempo[0:idx_primeros_segundos], respuesta_pendulo.y[1][0:idx_primeros_segundos],
plt.title('Ejemplo de condición inicial en un punto de equilibrio. Primer segundo.')
plt.legend()
plt.grid()
plt.ylabel(r'$\theta$, $\dot{\theta}$')
```

```
plt.xlabel('Tiempo (s)')
plt.show()
```





En este caso vemos que al tener un desplazamiento inicial de 45 grados, o $\pi/4$, al iniciar la simulación evoluciona con ángulo decreciente, hacia el punto de equilibrio x_{e0} antes simulado. En azul vemos cómo oscila en ángulo alrededor del 0, decreciendo la amplitud de la oscilación.

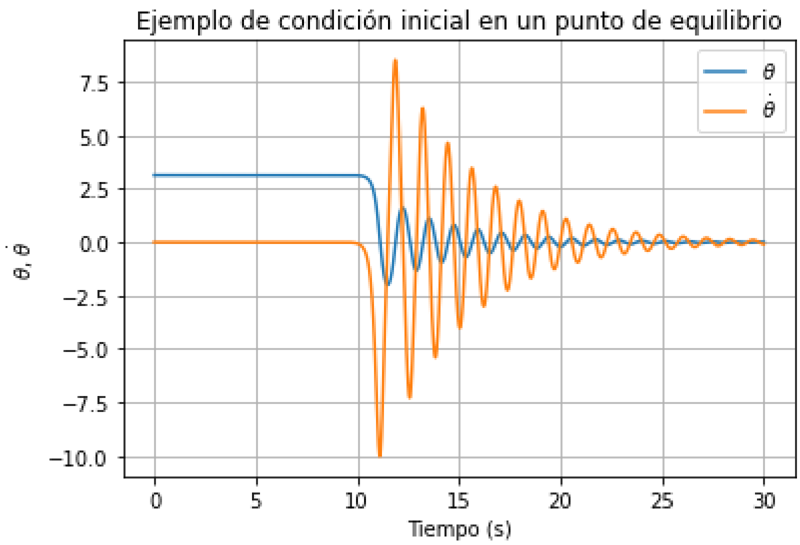
La velocidad angular oscila, porque el péndulo alcanza velocidad máxima en valor absoluto al cruzar el punto de equilibrio, y velocidad angular nula al llegar al ángulo máximo de cada oscilación. Notar en el detalle de los primeros segundo cómo la amplitud máxima de oscilación va cayendo en cada oscilación (trazo azul). Lo mismo sucede para la velocidad angular máxima en cada cruce sucesivo del origen (trazo naranja).

Hasta acá no hay ninguna sorpresa. Como propusimos en el modelo, el sistema tiene un rozamiento y sabemos que eso hace que pierda energía y eventualmente el péndulo deje de oscilar y termine en el punto de equilibrio.

Nos queda poner al péndulo en el otro punto de equilibrio: $x_{e2} = [\pi \, 0]^T$:

```
x0 = [pi, 0]

tiempo_final = 30
```

```
time_span = (tiempo_inicial, tiempo_final)
tiempo = np.arange(tiempo_inicial, tiempo_final, paso_tiempo)

respuesta_pendulo = sp.integrate.solve_ivp(f_pendulo, time_span, x0, t_eval=tiempo, method

plt.plot(tiempo, respuesta_pendulo.y[0], label=r'$\theta$')
plt.plot(tiempo, respuesta_pendulo.y[1], label=r'$\dot{\theta}$')
plt.title('Ejemplo de condición inicial en un punto de equilibrio')
plt.legend()
plt.grid()
plt.ylabel(r'$\theta$, $\dot{\theta}$')
plt.xlabel('Tiempo (s)')
plt.show()
```



¡Sorpresa! A pesar de arrancar lo más bien en el punto de equilibrio x_{e1}, vemos que pasado un tiempo el péndulo empieza a oscilar y termina evolucionando hacia x_{e0}.

Para pensar: ¿Por qué te parece que sucede esto?

En síntesis, Lyapunov dice que un sistema es estable alrededor de un punto de equilibrio si puedo ubicar la condición inicial lo suficientemente cerca de él y garantizar que no se aleje en una cierta distancia del mismo. Esa distancia puede ser cualquiera que se nos ocurra.

Si bien no lo demostramos, sino que lo mostramos, y con la intuición de la física del problema podemos decir que:

- $x_{e0} = [0\,0]^T$ es un punto de equilibrio estable y
- $x_{e1} = [\pi\,0]^T$ es un punto de equilibrio inestable

ambos, según el criterio de estabilidad de Lyapunov.

Ejercicio: Para demostrar la estabilidad Lyapunov propone un método llamado directo y que vimos en la clase teórica en donde se busca una función relacionada a la energía del sistema. ¿Podés demostrarlo para este caso?. Queda como desafío para ver si te animás.

RTA:(incompleto) Definimos las coordenas del centro de masa $L = l/2$

$$r_m = (Lsen(\theta), -Lcos(\theta))$$

$$v_m = (Lcos(\theta)\dot{\theta}, Lsen(\theta)\dot{\theta}) \Rightarrow |v_m| = (L\dot{\theta})^2$$

La energia mecanica será la suma de las energias cinetizas y potenciales.

$$V_{mec} = \frac{1}{2}m(L\dot{\theta})^2 + mg(-Lcos(\theta)) + \frac{1}{2}\gamma\dot{\theta}^2$$

es una buena candidata a funcion de Lyapunov y entonces derivo respecto del tiempo.

$$\frac{\partial V_{mec}}{\partial t} = mL^2\dot{\theta}\ddot{\theta} + mgLsen(\theta)\dot{\theta} + \gamma\dot{\theta}\ddot{\theta}$$

si reemplazo la ecuacion de movimiento del pendulo mostrada arriba en $\ddot{\theta} = \frac{-mgLsen(\theta)}{J} - \frac{-\gamma\dot{\theta}}{J}$, (entrada nula)

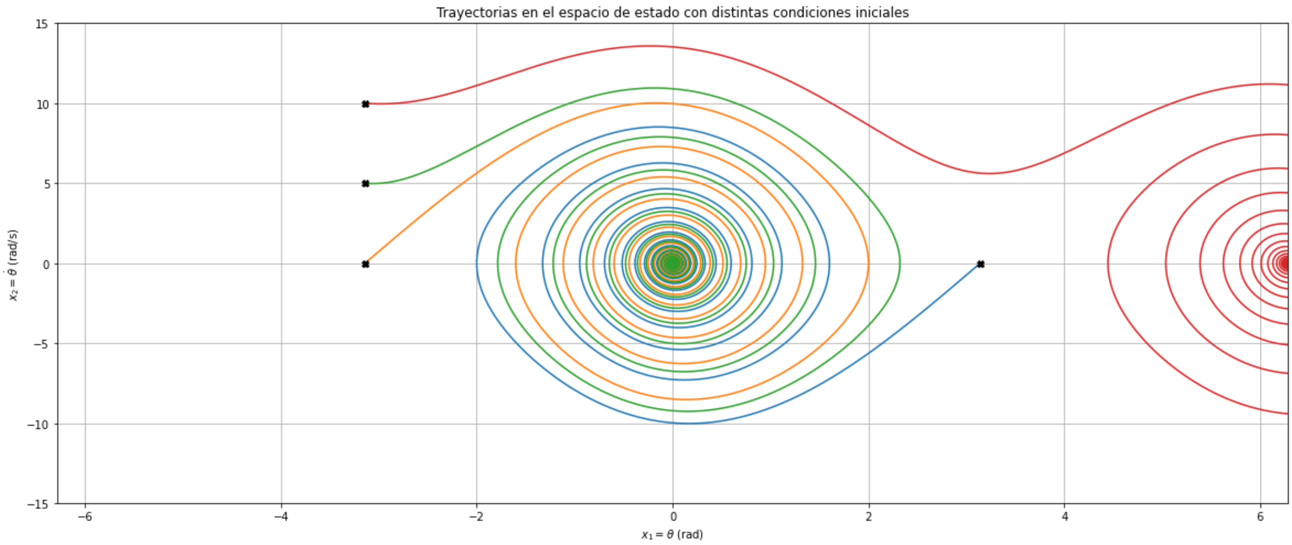
$$\frac{\partial V_{mec}}{\partial t} = mL^2\dot{\theta}(\frac{-mgLsen(\theta)}{J} - \frac{\gamma\dot{\theta}}{J}) + mgLsen(\theta)\dot{\theta} + \gamma\dot{\theta}(\frac{-mgLsen(\theta)}{J} - \frac{\gamma\dot{\theta}}{J})$$

$$J = \frac{1}{3}ml^2$$

Reemplazando debria llegar a que se cancelan los terminos con senos y me quedaria en funcion del rozamiento, ya que de este depende que sea asintoticamente estable, si no considero los terminos de rozamiento deberia tener derivada nula y eso indicara que es estable(oscila eternamente alrededor de un punto de equilibrio)NO SE CANCELAN

Repitamos ahora varias simulaciones, para diversas condiciones iniciales, pero en un diagrama de fase:

```
# Diagrama de fase de python-control. Requiere invertir el argumento de la función t «-» x
def ode_f_pendulo(x, t):
    return f_pendulo(t, x)

X0 = np.array([
     [pi,0],
     [-pi,0],
     [-pi,5],
     [-pi,10],
     ])

plt.figure(figsize=(20,8))
plt.axis([-2*pi, 2*pi, -15, 15])
plt.title('Trayectorias en el espacio de estado con distintas condiciones iniciales')
plt.ylabel(r'$x_2 = \dot{\theta}$ (rad/s)')
plt.xlabel(r'$x_1 = \theta$ (rad)')
ctrl.phase_plot(ode_f_pendulo, X0=X0, T=tiempo)
plt.plot(X0[:,0], X0[:,1], 'Xk')
plt.grid()
plt.show()
```



Ejercicio: Explicá qué es lo que está pasando en cada uno de los cuatro casos. Los puntos negros marcan cada una de las condiciones iniciales probadas.

RTA:

Se observa que para el punto $[\pi, 0]$ que es un punto de equilibrio inestable, es estable respecto del origen $[0, 0]$ luego en los demas casos son condiciones iniciales que evolucionan asintoticamente hacia el punto de equilibrio estable $[0, 0]$. son asintoticamente estables respecto de x_{e0}

El ultimo caso $[-\pi, 10]$ es tal que es suficiente velocidad como para que el pendulo dé un giro completo y termine en el siguiente punto estable. puesto que hay infinitos puntos de estabilidad.

No es estable respecto de x_{e0} pero si respecto de $[2\pi, 0]$

Para pensar: Vimos que el origen $x_{e0} = [0\,0]^T$ es un punto de equilibrio estable, ¿lo es localmente o globalmente y por qué? ¿Es asintóticamente estable?

RTA:

Es localmente estable. puesto que se analiza respecto de un punto de equilibrio sin tener en cuenta los otros. entonces como vimos en el caso $[-\pi, 10]$ no evoluciona hacia el origen x_{e0}, pero hay un rango de condiciones iniciales que si lo hacen y lo hacen de forma asintotica.

Estabilidad interna en sistemas LTI

¿Cómo se aplica el criterio de estabilidad interna o de Lyapunov a sistemas LTI? Repasá la clase de estabilidad y anotá cuáles son las condiciones que la garantizan (https://youtu.be/tw-Jkd4qjWw).

Vamos a explorarlo como parte de la tarea, con los siguientes ejercicios

Tarea:

1) a) *Linealizar alrededor de los puntos de equilibrio hallados:*

$$x_{e0} = [0\,0]^T$$

y

$$x_{e1} = [\pi\,0]^T$$

Estudiar para cada uno de ellos la estabilidad interna del modelo aproximado lineal. ¿Son Lyapunov estables? ¿Son asintóticamente estables? ¿Son exponencialmente estables?

RTA:

Para linealizar alrededor de un punto hay que aproximar con Taylor, truncar etc.

Se puede llegar a la conclusión de que para linealizar un sistema en espacio de estados las matrices linealizadas serán los jacobianos evaluados en el punto de equilibrio. es decir:

Dado:

$$A = \begin{bmatrix} \frac{\partial f_1(x,u)}{\partial x_1} & \frac{\partial f_1(x,u)}{\partial x_2} \\ \frac{\partial f_2(x,u)}{\partial x_1} & \frac{\partial f_2(x,u)}{\partial x_2} \end{bmatrix}_{\substack{x=x_e \\ u=u_e}} = \begin{bmatrix} 0 & 1 \\ \frac{-3g}{2l}cos(x_{1e}) & \frac{-3\gamma}{ml^2} \end{bmatrix}$$

$$B = \begin{bmatrix} \frac{\partial f_1(x,u)}{\partial u} \\ \frac{\partial f_2(x,u)}{\partial u} \end{bmatrix}_{\substack{x=x_e \\ u=u_e}} = \begin{bmatrix} 0 \\ \frac{3}{ml^2} \end{bmatrix}$$

Luego para cada posición de equilibrio seleccionada, la matriz de estados A cambiara. mientras que la matriz de entrada B será la misma.

$$A_{x_{e0}} = \begin{bmatrix} 0 & 1 \\ \frac{-3g}{2l} & \frac{-3\gamma}{ml^2} \end{bmatrix} A_{x_{e1}} = \begin{bmatrix} 0 & 1 \\ +\frac{3g}{2l} & \frac{-3\gamma}{ml^2} \end{bmatrix}$$

Ahora mismo son sistemas LTI por lo que vale la estabilidad de Lyapunov para sistemas LTI respecto de algún punto de equilibrio.

Caso Continuo:

1. Es estable segun Lyapunov si los autovalores de A no están en el semiplano derecho. Los autovalores que esten sobre el eje imaginario tienen multiplicidad algebraica=geometrica

2. Es exponencialmente estable segun Lyapunov si los autovalores de A estan en el semiplano izquierdo.

```
#Matrices de estado linealizadas.
Ae0 = symb.Matrix([ [0, 1], [(-3 * g)/(2 * l), (-3 * gamma)/(m * l**2)] ])
Ae1 = symb.Matrix([ [0, 1], [(+3 * g)/(2 * l), (-3 * gamma)/(m * l**2)] ])

print('Autovalres con Xe0=[0; 0] :'); symb.pprint(Ae0.eigenvals())#Autoval complejos conju
print('Autovalres con Xe1=[pi;0] :'); symb.pprint(Ae1.eigenvals())#Autoval reales, distint
```

```
Autovalres con Xe0=[0; 0] :
{-0.25 - 5.47151715705982⋅i: 1, -0.25 + 5.47151715705982⋅i: 1}
Autovalres con Xe1=[pi;0] :
{-5.73292804986533: 1, 5.23292804986533: 1}
```

los autovalores para $x_{e0} = [0, 0]$: $\lambda_{1,2} = -0.25 \pm 5.47151715705982j$ entonces segun Lyapunov es exponencialmente estable(tambien asintoticamente).

Para $x_{e1} = [\pi, 0]$: $\lambda_1 = -5.73292804986533$ y $\lambda_2 = 5.23292804986533$ vemos que no es estable segun Lyapunov dado que solo un autovalor está en el semiplano izquierdo. esto tambien se lo conoce como punto de ensillaadura inestable.

b) *Repetir para $\gamma = 0$. ¿Son Lyapunov estables? ¿Asintóticamente estables? ¿Exponencialmente estables?*

Simulá respuestas temporales para mostrar lo que sucede en a) y en b) para ambos puntos de equilibrio. Podés mostrar los resultados con gráficos en función del tiempo o diagramas de fase.

Relacioná lo que observas con el sistema no lineal original.

RTA:

si $\gamma = 0$ entonces las matrices de estados son:

$$A_{x_{e0}} = \begin{bmatrix} 0 & 1 \\ \frac{-3g}{2l} & 0 \end{bmatrix} A_{x_{e1}} = \begin{bmatrix} 0 & 1 \\ +\frac{3g}{2l} & 0 \end{bmatrix}$$

```
#Matrices de estado linealizadas: gamma= 0.
Ae0 = symb.Matrix([ [0, 1], [(-3 * g)/(2 * l), 0] ])
Ae1 = symb.Matrix([ [0, 1], [(+3 * g)/(2 * l), 0] ])

print('Autovalres con Xe0=[0; 0] :'); symb.pprint(Ae0.eigenvals())#Autoval complejos conju
print('Autovalres con Xe1=[pi ;0] :'); symb.pprint(Ae1.eigenvals())#Autoval reales, iguale

print('Autovectores con Xe0=[0; 0] :\n'); symb.pprint(Ae0.eigenvects()) #ma=mg.
print('Autovectores con Xe1=[pi;0] :\n'); symb.pprint(Ae1.eigenvects()) #ma=mg.
```

```
Autovalres con Xe0=[0; 0] :
{-5.47722557505166⋅i: 1, 5.47722557505166⋅i: 1}
Autovalres con Xe1=[pi ;0] :
{-5.47722557505166: 1, 5.47722557505166: 1}
Autovectores con Xe0=[0; 0] :
⎡⎛                          ⎡⎡-0.179605302026775 ⎤⎤⎞  ⎛                         ⎡⎡
⎢⎜-5.47722557505166⋅i, 1, ⎢⎢                    ⎥⎥⎟, ⎜5.47722557505166⋅i, 1, ⎢⎢
```

0.983738753675929·i

0.191991874580346·i

-1.05158280565358

Autovectores con Xe1=[pi;0] :

-5.47722557505166, 1, 0.179605302026775 -0.983738753675929, 5.47722557505166, 1, 0.191 1.051

991874580346

58280565358

Observar que los autovalores son los mismos!, en modulo.

Para x_{e0} : $\lambda_{1,2} = \pm 5.47722557505166j$

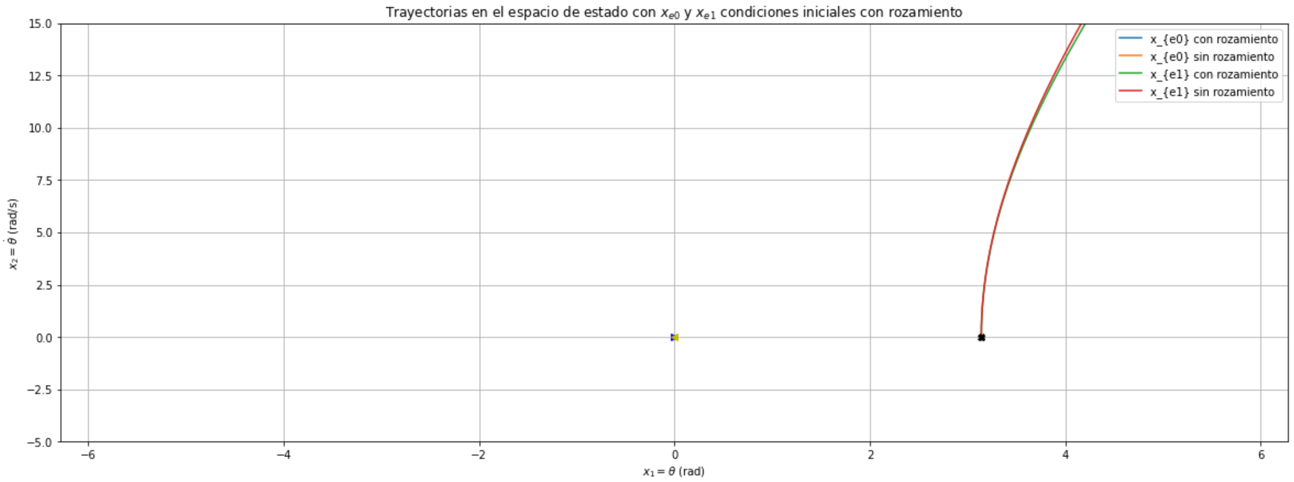
Segun Lyapunov para el primer caso, los autovalores que esten sobre el eje imaginario requiere multiplicidad algebraica=geometrica, esto se cumple. Asi que converge segun Lyapunov. es lo que se conoce como foco/centro estable.

Para x_{e1} : $\lambda_{1,2} = 5.47722557505166$

Ambos autovalores son los mismos pero estan en el semiplano derecho. son inestables. se lo conoce como punto estrella inestable.

```
#SS_c = ctrl.StateSpace(A, B ,C, D)
################################################################################
def f_penduloA0Ros(x, t, l=l, m=m, g=g, gamma=gamma):
    u = u_function(t)
    x1 = x[0] # x1: theta(t)
    x2 = x[1] # x2: dtheta_dt(t)
    dx1_dt = x2
    dx2_dt = -3*g/(2*l) * x1 - 3*gamma/(m*l**2) * x2 + 3/(m*l**2)*u
    return [dx1_dt, dx2_dt]
def f_penduloA1Ros(x, t, l=l, m=m, g=g, gamma=gamma):
    u = u_function(t)
    x1 = x[0] # x1: theta(t)
    x2 = x[1] # x2: dtheta_dt(t)
    dx1_dt = x2
    dx2_dt = +3*g/(2*l) * x1 - 3*gamma/(m*l**2) * x2 + 3/(m*l**2)*u
    return [dx1_dt, dx2_dt]
################################################################################
def f_penduloA0(x, t, l=l, m=m, g=g, gamma=0):
    u = u_function(t)
    x1 = x[0] # x1: theta(t)
    x2 = x[1] # x2: dtheta_dt(t)
    dx1_dt = x2
    dx2_dt = -3*g/(2*l) * x1 - 3*gamma/(m*l**2) * x2 + 3/(m*l**2)*u
    return [dx1_dt, dx2_dt]
def f_penduloA1(x, t, l=l, m=m, g=g, gamma=0):
    u = u_function(t)
    x1 = x[0] # x1: theta(t)
```

```
    x2 = x[1] # x2: dtheta_dt(t)
    dx1_dt = x2
    dx2_dt = +3*g/(2*l) * x1 - 3*gamma/(m*l**2) * x2 + 3/(m*l**2)*u
    return [dx1_dt, dx2_dt]
################################################################################
#X0 = np.array([ [pi,0], [0,0] ])
xe0= np.array([[0, 0]])
xe1= np.array([[pi, 0]])


#Diagrama de fase sistema con rozamiento y sin rozamiento.
plt.figure(figsize=(20,7))
plt.axis([-2*pi, 2*pi, -5, 15])
plt.title('Trayectorias en el espacio de estado con $x_{e0}$ y $x_{e1}$ condiciones inicia
plt.ylabel(r'$x_2 = \dot{\theta}$ (rad/s)')
plt.xlabel(r'$x_1 = \theta$ (rad)')
ctrl.phase_plot(f_penduloA0Ros, X0=xe0, T=tiempo)
ctrl.phase_plot(f_penduloA0, X0=xe0, T=tiempo)
ctrl.phase_plot(f_penduloA1Ros, X0=xe1, T=tiempo)
ctrl.phase_plot(f_penduloA1, X0=xe1, T=tiempo)
plt.plot(xe0[0,0], xe0[0,1], '>b');plt.plot(xe0[0,0], xe0[0,1], '<y'); plt.plot(xe1[0,0],
plt.legend(('x_{e0} con rozamiento', 'x_{e0} sin rozamiento', 'x_{e1} con rozamiento','x_{
plt.grid()
plt.show()
```

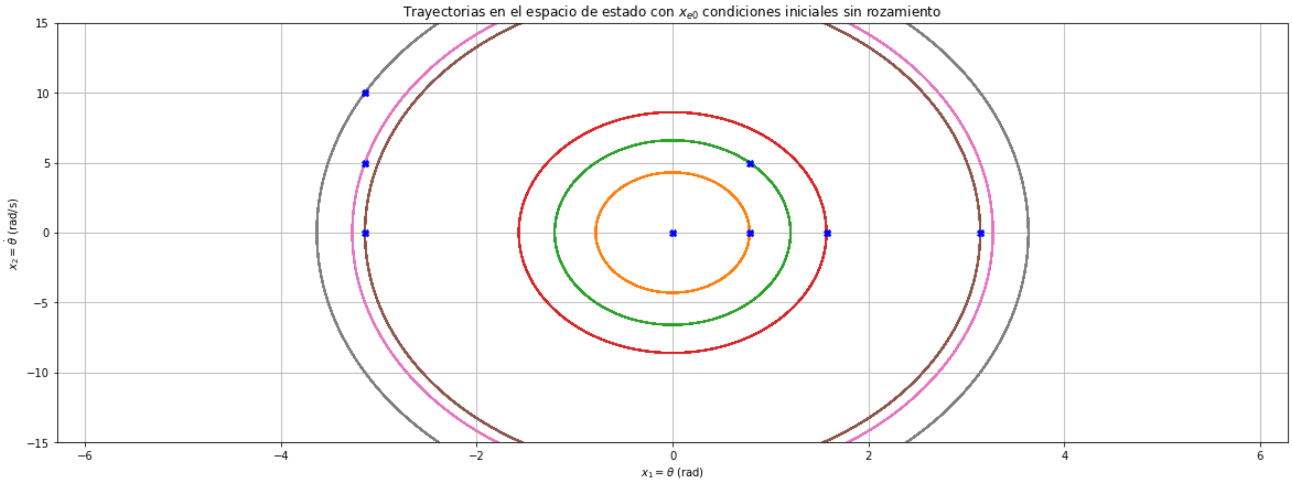


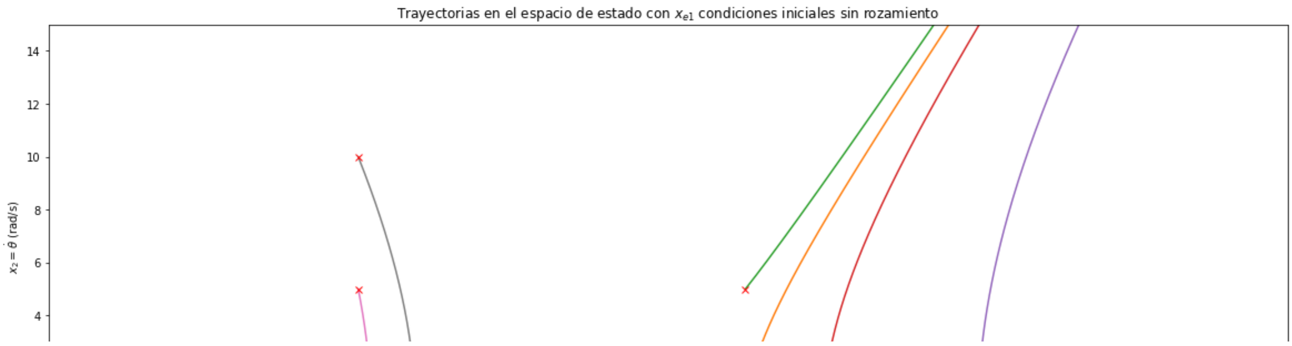
Observar que ambos diagramas de fase son parecidos, con y sin rozamiento en el eje. esto es por que en el punto $[0, 0]$ convergen al mismo punto. y en el punto $[\pi, 0]$ ambos son inestables y divergen.

No obstante para remarcar las diferencias se grafican los diagramas de fase para cada caso con distintas condiciones iniciales.

Esto para hacer notar que para hablar de estabilidad de Lyapunov se habla siempre respecto de un punto de equilibrio.

```
#Para visualizar mejor los diagramas de fase:
X0 = np.array([ [0,0],[pi/4,0],[pi/4,5],[pi/2,0],[pi,0],[-pi,0],[-pi,5],[-pi,10], ])
#linealizado en xe0=[0,0]
plt.figure(figsize=(20,7))
plt.axis([-2*pi, 2*pi, -15, 15])
plt.title('Trayectorias en el espacio de estado con $x_{e0}$ condiciones iniciales sin roz
plt.ylabel(r'$x_2 = \dot{\theta}$ (rad/s)')
plt.xlabel(r'$x_1 = \theta$ (rad)')
ctrl.phase_plot(f_penduloA0, X0=X0, T=tiempo)
plt.plot(X0[:,0], X0[:,1], 'Xb')
plt.grid()
#Linealizado en xe1=[pi,0]
plt.figure(figsize=(20,7))
plt.axis([-2*pi, 2*pi, -1, 15])
plt.title('Trayectorias en el espacio de estado con $x_{e1}$ condiciones iniciales sin roz
plt.ylabel(r'$x_2 = \dot{\theta}$ (rad/s)')
plt.xlabel(r'$x_1 = \theta$ (rad)')
ctrl.phase_plot(f_penduloA1, X0=X0, T=tiempo)
plt.plot(X0[:,0], X0[:,1], 'xr')
plt.show()
```





2)

Para el ejemplo de Ale y su auto de la introducción vimos que el sistema no era estable BIBO. Definí el sistema con las siguientes variables de estado:

$$x = \begin{bmatrix} p(t) \\ v(t) \end{bmatrix}$$
$$y(t) = p(t)$$
$$u(t) = f$$

Con $p(t)$ la posición del vehículo y $v(t)$ su velocidad ¿cuál es el punto de equilibrio? ¿es Lyapunov estable? ¿por qué?

Redefiní el sistema pero ahora de la siguiente manera:

$$x = v(t)$$
$$y(t) = v(t)$$
$$u(t) = f$$

¿Es BIBO estable? ¿Es Lyapunov estable? ¿Es asintóticamente estable? ¿Es exponencialmente estable?

Mostrá lo que pasa con unas simulaciones en el tiempo. Podés mostrar los resultados con gráficos en función del tiempo o diagramas de fase

RTA:

Para el primer caso:

$$\dot{x} = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix} x + \begin{bmatrix} 0 \\ \frac{1}{m} \end{bmatrix} u$$

$$y = \begin{bmatrix} 1 & 0 \end{bmatrix} x$$

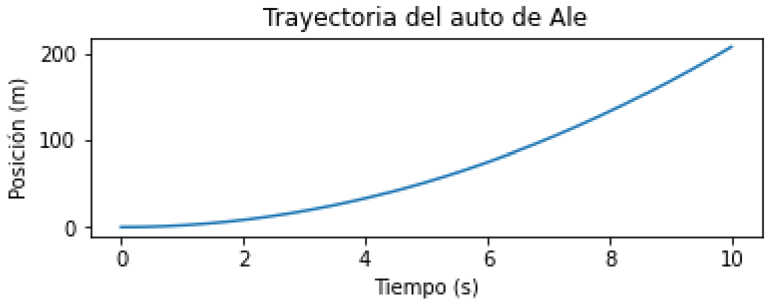
Buscando los puntos de equilibrio de forma analoga al ejercicio anterior: $x_e = [p, 0]^T$ está en equilibrio con cualquier posición y velocidad nula, es decir que hay infinitos puntos de equilibrio.

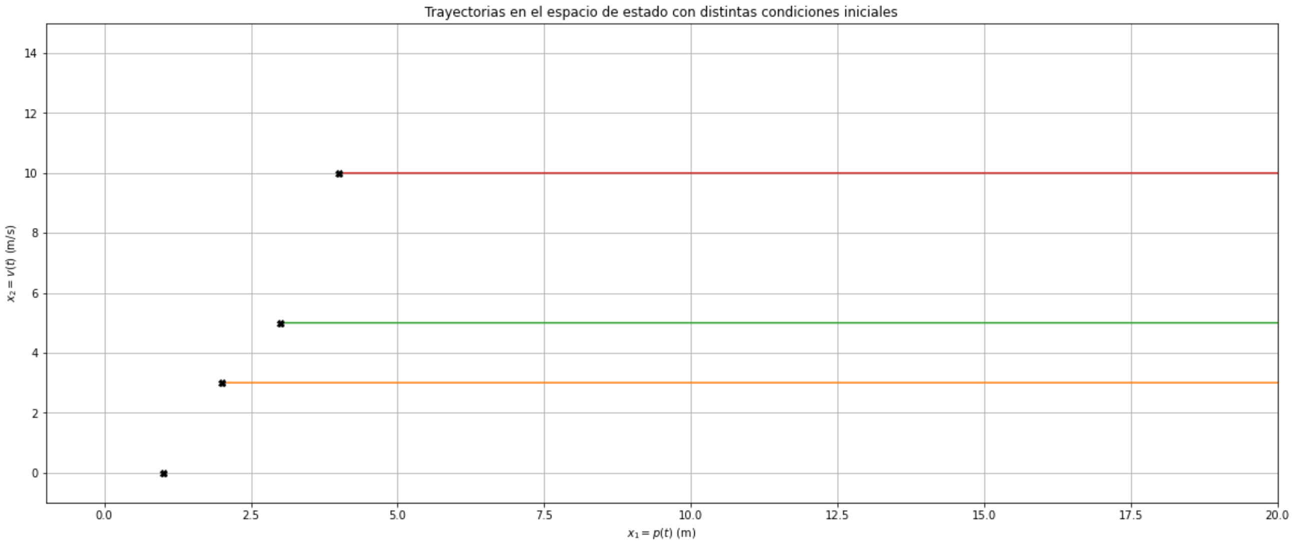
Notar que la matriz de estados es claramente una forma de jordan, salta a la vista que tiene dos autovalores nulos, multiplicidad algbraica 2 y un solo autovector asociado. Por Lyapunov ma $\neq$ mg este sistema no es estable.

```
#Traigo devuelta las condiciones:
t_ini = 0; t_fin = 10; t_paso = 0.01 ;t = np.arange(t_ini, t_fin, t_paso)
u = 1.0*np.ones(len(t))

#para usar en el diagrama de fase:
def auto_pos(x, t, m=M):
    u = u_function(t)
    x1 = x[0] # x1: p(t)
    x2 = x[1] # x2: v(t)
    dx1_dt = x2; dx2_dt = u/m
    return [dx1_dt, dx2_dt]  ##Deveria obtener rectas inestables DIAGONALES

#Creo el ss.
Alin= symb.Matrix([[0, 1],[0, 0]]); Blin=symb.Matrix([[0],[1/m]]); Clin=symb.Matrix([[1, 0
auto= ctrl.StateSpace(Alin, Blin ,Clin, Dlin)
print('Autovectores:\n',Alin.eigenvects())
#[eigenval, multiplicity, eigenspace]==[0, 2, [1,0]] ma/=mg.

#La respuesta al escalon es
t, z,= ctrl.step_response(auto, t)

plt.subplot(212)
plt.plot(t, z)
plt.xlabel('Tiempo (s)')
plt.ylabel('Posición (m)')
"""
velocidad = estado[0]
plt.subplot(211)
plt.ylabel('Velocidad (m/s)')
plt.plot(t, velocidad)
"""
plt.title('Trayectoria del auto de Ale')
plt.show()

##################################################################################
#Respecto al origen.
#Diagrama de fase sistema con rozamiento y sin rozamiento.
X0 = np.array([
     [1,0],
     [2,3],
     [3,5],
     [4,10],
```

```
    ]) #[p(t),v(t)]

plt.figure(figsize=(20,8))
plt.axis([-1, 20, -1, 15])
plt.title('Trayectorias en el espacio de estado con distintas condiciones iniciales')
plt.ylabel(r'$x_2 = v(t)$ (m/s)')
plt.xlabel(r'$x_1 = p(t)$ (m)')
ctrl.phase_plot(auto_pos, X0=X0, T=tiempo)
plt.plot(X0[:,0], X0[:,1], 'Xk')
plt.grid()
plt.show()
```

```
Autovectores:
 [(0, 2, [Matrix([
[1],
[0]])])]
```





Segundo caso:

$$\dot{x} = 0x + \frac{1}{m}u$$
$$y = 1x$$

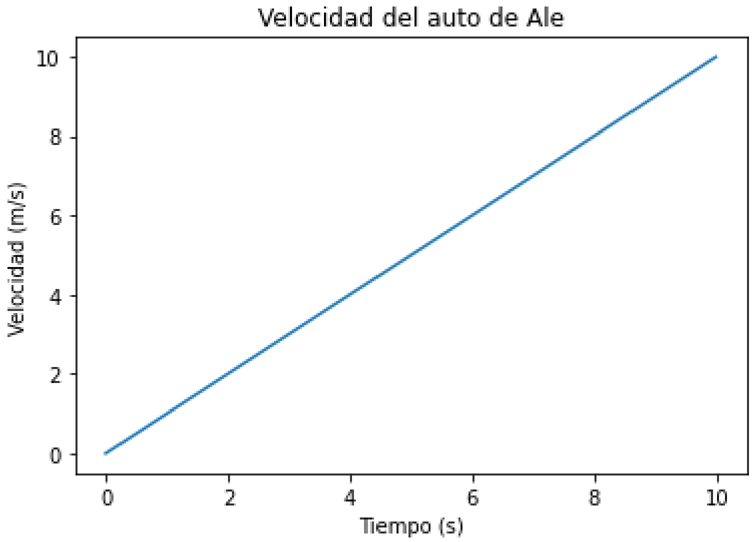
Buscando los puntos de equilibrio para este caso: $x_e = [v]$ está en equilibrio para cualquier velocidad.

Cualquier punto es punto de equilibrio. entonces si se analiza la estabilidad según Lyapunov se vé que mg=ma entonces es Lyapunov estable respecto del punto de equilibrio x_e que se elija, No es asintoticamente estable.

No es BIBO estable, se puede ver en la respuesta al escalon, entrada acotada no dio una salida acotada.

```
#para usar en el diagrama de fase, aca no me sirve.
def auto_vel(x, t, m=M):
    u = u_function(t)
    dx_dt = u/m
    return [dx_dt]

#Creo el ss.
Alinss= symb.Matrix([0]); Blinss=symb.Matrix([1/M]); Clinss=symb.Matrix([1]); Dlinss= symb
auto_velss= ctrl.StateSpace(Alinss, Blinss ,Clinss, Dlinss)
print('Autovectores:\n',Alinss.eigenvects()) #ma=mg
#La respuesta al escalon es
t, velss= ctrl.step_response(auto_velss, t)
plt.plot(t, velss)
plt.xlabel('Tiempo (s)')
plt.ylabel('Velocidad (m/s)')
## mg=ma son Lyapunov estables! pero no son BIBO estables.

plt.title('Velocidad del auto de Ale')
plt.show()
```

```
Autovectores:
 [(0, 1, [Matrix([[1]])])]
```



0 s se ejecutó 23:05