CMA CENTRO DE INVESTIGACIÓN EN METODOS CUANTITATIVOS APLICADOS A LA ECONOMÍA Y LA GESTIÓN
IADCOM - UBA

Universidad de Buenos Aires
Facultad de Ciencias Económicas

http://www.economicas.uba.ar/institutos_y_centros/revista-modelos-matematicos/

NOTAS DE CLASE: ECUACIONES DIFERENCIALES ORDINARIAS Y SU RESOLUCIÓN CON PYTHON

García Fronti, Verónica[1] y Dotta, Milena[2]

[1]Universidad de Buenos Aires. Facultad de Ciencias Económicas. Instituto de Investigaciones en Administración, Contabilidad y Métodos Cuantitativos para la Gestión (IADCOM). Centro de Investigación en Métodos Cuantitativos Aplicados a la Economía y la Gestión (CMA). Ciudad Autónoma de Buenos Aires. Argentina.

[2]Universidad de Buenos Aires. Facultad de Ciencias Económicas. Ciudad Autónoma de Buenos Aires. Argentina.

vgarciafronti@economicas.uba.ar mile.dotta@gmail.com

Resumen

Recibido: 10-10-2022

Aceptado: 12-12-2022

Palabras clave

Ecuaciones diferenciales.

Python.

Estas notas de clase presentan una introducción al lenguaje de programación Python a través de la resolución de ecuaciones diferenciales ordinarias lineales con coeficientes constantes de segundo orden.

Las mismas están destinadas a los estudiantes de la asignatura Análisis Matemático II que ya han estudiado la resolución analítica de las mismas. A partir de estos conocimientos se plantea introducirlos en la programación en Python.

El enfoque que se utilizará es el de aprendizaje mediante aplicaciones económicas como es el caso del modelo de mercado dinámico. Para esto en la primera parte se explicarán los conceptos fundamentales del modelo involucrado y se describirá resumidamente el análisis realizado con los estudiantes sobre esta temática y luego se enseña paso a paso el código necesario para resolver las ecuaciones diferenciales involucradas en el modelo utilizando la librería Sympy de Python y la confección de los gráficos correspondientes mediante la librería Matplotlib de Python.

ISSN (En línea) 2362 3225

NOTAS DE CLASE: ECUACIONES DIFERENCIALES ORDINARIAS Y SU RESOLUCIÓN CON PYTHON

García Fronti, Verónica y Dotta, Milena

Abstract

Keywords

Differential equations.

Python.

These lecture notes present an introduction to the Python programming language through solving linear ordinary differential equations with second-order constant coefficients.

They are intended for students who have already studied their analytical resolution. Based on this knowledge, it is proposed to introduce them into programming in Python.

The approach that will be used is learning through economic applications such as the dynamic market model. For this, in the first part the fundamental concepts of the model involved will be explained and the analysis carried out with the students on this subject will be briefly described and then the code necessary to solve the differential equations involved in the model using the Sympy library is taught step by step. of Python and the preparation of the corresponding graphs using the Matplotlib library.

ISSN (En línea) 2362 3225

INTRODUCCIÓN

El aprendizaje de lenguajes de programación se ha vuelto cada vez más necesario para los estudiantes en Ciencias Económicas. Python es de los lenguajes que se ha vuelto más popular ya que es un lenguaje versátil y fácil de aprender. Es así que Thomas Sargent posee un sitio web, desarrollado con John Stachurski en el que presenta herramientas de Python en la resolución de modelos económicos y financieros (Koruki, 2021).

Muchas veces ingresar al mundo de los lenguajes de programación se vuelve tedioso cuando los ejemplos propuestos están alejados del interés de los estudiantes, es por este motivo que se buscó aprovechar los conocimientos que ya poseen los estudiantes sobre ecuaciones diferenciales y su resolución para que sean el puntapié inicial para despertar interés en esta temática.

La actividad propuesta en estas notas de clase se debe desarrollar luego de que los alumnos hayan aprendido los diferentes métodos analíticos para lo resolución de ecuaciones diferenciales ordinarias (EDO) y los modelos matemáticos básicos en donde es necesario resolverlas.

El artículo se estructura de la siguiente forma, en la primera parte se describen los conceptos básicos involucrados en un modelo de mercado dinámico y la metodología para resolver una EDO con coeficientes constantes, luego se presentan dos ejemplos numéricos que se resuelven tanto analíticamente como mediante el lenguaje de programación Python en el entorno de Google Colaboratory.

Los ejemplos que se analizarán serán de EDO de segundo orden resultante de un modelo de mercado, para esto previamente se plantean algunos conceptos relevantes del modelo y se describe el análisis dinámico que se realiza. Para profundizar sobre estos conceptos se recomienda: Chiang, 2006.

1. CONCEPTOS BÁSICOS DEL MODELO MATEMÁTICO

En los modelos de mercado dinámico se tiene en cuenta que los compradores y vendedores pueden tomar sus decisiones no solo en base al precio corriente del bien sino también en base a la tendencia temporal del precio. Si se considera al tiempo continuo la tendencia temporal del precio se obtiene, principalmente, a través de sus derivadas primera y segunda con respecto al tiempo. De esta forma el modelo de mercado dinámico (en un mercado perfecto) se podría plantear de la siguiente forma:

$$\begin{cases} Q_d = f(p(t), p'(t), p''(t)) \\ Q_s = g(p(t), p'(t), p''(t)) \\ Q_d = Q_s \end{cases}$$

Donde:

$Q_d: Cantidad\ demandada\ del\ bien$
$Q_s: Cantidad\ ofrecida\ del\ bien$
$p: precio\ del\ bien$
$p'(t) = \frac{dp}{dt}$
$p''(t) = \frac{d^2p}{dt^2}$

Si se considera que las funciones de oferta y demanda poseen un comportamiento lineal el modelo presenta la siguiente estructura:

$$\begin{cases} Q_d = a_0\, p''(t) + a_1\, p'(t) + a_2 p(t) + a_3 \\ Q_s = b_0\, p''(t) + b_1\, p'(t) + b_2 p(t) + b_3 \\ Q_d = Q_s \end{cases} \quad (1)$$

Donde:

$Q_d: Cantidad\ demandada\ del\ bien$
$Q_s: Cantidad\ ofrecida\ del\ bien$
$p: precio\ del\ bien$
$p'(t) = \frac{dp}{dt}$
$p''(t) = \frac{d^2p}{dt^2}$

Los números reales: $a_0, a_1, a_2, a_3, b_0, b_1, b_2, b_3$ son los parámetros del modelo.

En este modelo el precio de equilibrio es: $p_e = \frac{b_3 - a_3}{a_2 - b_2}$

Este modelo (1) puede representarse a través de la siguiente ecuación diferencial de segundo orden con coeficientes constantes:

$$(a_0 - b_0)p''(t) + (a_1 - b_1)\, p'(t) + (a_2 - b_2)p(t) = b_3 - a_3 \quad (2)$$

Donde la ecuación diferencial homogénea asociada es:

$$(a_0 - b_0)p''(t) + (a_1 - b_1)\, p'(t) + (a_2 - b_2)p(t) = 0 \quad (3)$$

La *solución general,* p_g, de la ecuación diferencial (2) se puede expresar de la siguiente forma:

$$p_g(t) = p_h(t) + p_c(t)$$

En donde $p_h(t)$ es la solución de la *ecuación homogénea asociada* (3) y $p_c(t)$ es la *solución complementaria* que se obtiene al resolver la ecuación diferencial (2).

Para determinar la solución homogénea, $p_h(t)$, se ensaya la solución: $p_h(t) = e^{rt}$. Cuyas derivadas son: $p_h{'}(t) = re^{rt}$ $p_h{''}(t) = r^2e^{rt}$. Al reemplazar la función propuesta y sus sucesivas derivadas en la ecuación diferencial homogénea (3) resulta:

$$(a_0 - b_0)r^2e^{rt} + (a_1 - b_1)\,re^{rt} + (a_2 - b_2)e^{rt} = 0$$

Al sacar factor común e^{rt}:

$$e^{rt}[(a_2 - b_2)r^2 + (a_1 - b_1)r + (a_0 - b_0)] = 0$$

Como $e^{rt} \neq 0$, se obtiene una ecuación a la que se denomina *ecuación característica*:

$$(a_2 - b_2)r^2 + (a_1 - b_1)r + (a_0 - b_0) = 0 \qquad (4)$$

Esta es una ecuación de segundo grado en r en donde se pueden presentar tres casos:

Caso I Raíces reales diferentes $r_1 \neq r_2$:

Cuya solución general de la ecuación diferencial homogénea (3) es: $p_h(t) = C_1e^{r_1t} + C_2e^{r_2t}$

Caso II: Raíces reales iguales $r_1 = r_2 = r$:

Cuya solución general de la ecuación diferencial homogénea (3) es: $p_h(t) = C_1e^{rt} + C_2te^{rt}$

Caso III: Raíces complejas $r = a \pm bi$:

Cuya solución general de la ecuación diferencial homogénea (3) es:

$$p_h(t) = e^{at}[C_1sen(bt) + C_2\cos(bt)]$$

Para hallar la solución complementaria, $p_c(t)$, se puede utilizar el método de los coeficientes indeterminados. Por lo que para la ecuación diferencial (2) la solución complementaria es:

$$p_c(t) = \frac{b_3 - a_3}{a_0 - b_0} \qquad (5)$$

De esta forma, es posible encontrar la solución general de la ecuación diferencial (2) como:

$$p_g(t) = p_h(t) + \frac{b_3 - a_3}{a_0 - b_0}$$

Estabilidad de la solución

Resulta interesante notar que en términos económicos la solución complementaria encontrada (5) se interpreta como el valor del precio de equilibrio estacionario de la solución del modelo dado. De esta forma, la diferencia entre la solución general y la solución complementaria muestra las desviaciones del equilibrio:

$$p_g(t) - \frac{b_3 - a_3}{a_0 - b_0} = p_h(t)$$

La estabilidad dinámica de la solución requerirá entonces que la solución homogénea se anule cuanto el tiempo, t, tiende a infinito. De esta forma, la solución de la ecuación diferencial (2) es asintóticamente estable si la solución de la ecuación homogénea tiende a cero cuando t tiende a

infinito independientemente del valor que tomen las constantes. Se puede demostrar que una ecuación diferencial a coeficientes constantes será asintóticamente estable cuando la parte real de las raíces de la ecuación característica (4) sean todas negativas. Para profundizar sobre la estabilidad de la solución se recomienda el libro de Bernardello, et. al. (2010).

A partir de estos conceptos se analizarán dos ejemplos numéricos de modelo de mercado con expectativas de precios. En ambos ejemplos se hallará la solución general y particular en forma analítica, se determinará el precio de equilibrio y se establecerá si el modelo es dinámicamente estable o no. Luego se procederá a describir el código en Python necesario para resolver la ecuación diferencial dada de forma de hallar la solución general y particular y por último se aprenderá el código para graficar las soluciones particulares encontradas.

2. EJEMPLO I: MODELO DE MERCADO ESTABLE

El siguiente modelo describe un mercado en donde tanto los compradores como los vendedores basan su comportamiento en el precio presente y su tendencia futura:

$$\begin{cases} Q_d = 8 - p(t) - 2p'(t) + p''(t) \\ Q_s = -2 + p(t) + p'(t) + 2p''(t) \\ Q_d = Q_s \end{cases} \qquad (6)$$

Donde:

Q_d: *Cantidad demandada del bien*
Q_s: *Cantidad ofrecida del bien*
p: *precio del bien*
$p'(t) = \frac{dp}{dt}$
$p''(t) = \frac{d^2p}{dt^2}$

La ecuación diferencial resultante del modelo es una ecuación diferencial a coeficientes constantes:

$$p''(t) + 3p'(t) + 2p(t) = 10 \qquad (7)$$

Se desea encontrar la trayectoria del precio de mercado en el modelo dado (6), ante dos condiciones iniciales. La primera condición inicial es: $p(0) = 1\ y\ p'(0) = 1$ y la segunda condición inicial que se desea analizar es: $p(0) = 8\ y\ p'(0) = 1$.

A continuación, se hallará analíticamente la solución general y las soluciones particulares para el problema planteado.

2.1 Resolución analítica

Para resolver analíticamente la ecuación diferencial (7), se debe hallar la solución homogénea y la solución complementaria. Para profundizar la resolución analítica de las EDOs se recomienda el libro: Di Caro, et. al. (2000).

Solución homogénea:

Se parte de la ecuación característica asociada a la ecuación diferencial homogénea:

$$r^2 + 3r + 2 = 0$$

Cuyas raíces son: -2 y -1, por lo tanto, la solución homogénea es:

$$p_h(t) = C_1 e^{-2t} + C_2 e^{-t}$$

Solución complementaria:

La solución complementaria se resuelve con el método de los coeficientes indeterminados. Se propone como solución complementaria $p_c(t) = A$ y se determina el coeficiente al reemplazar en la ecuación diferencial (7):

$$2A = 10 \implies A = 5$$

Por lo tanto, la solución complementaria es: $p_c(t) = 5$

La solución general es:

$$p_g(t) = C_1 e^{-2t} + C_2 e^{-t} + 5$$

Como ambas raíces de la ecuación característica son negativas se puede concluir que la solución es asintóticamente estable, es decir que cualesquiera sean las condiciones iniciales el precio converge al precio de equilibrio que en este caso es 5.

Si las condiciones iniciales son: $p(0) = 1$, es decir un precio inicial menor al precio de equilibrio y $p'(0) = 1$ La solución particular de la EDO es:

$$p_p(t) = 5 - 7e^{-t} + 3e^{-2t}$$

Si ahora se supone un precio inicial mayor al precio de equilibrio: $p(0) = 8\ y\ p'(0) = 1$, la solución particular de la EDO es:

$$p_p(t) = 5 + 7e^{-t} - 4e^{-2t}$$

De esta forma, se ha resuelto siguiendo el mismo procedimiento visto en la clase el modelo de mercado dado, a continuación, se describirá como resolver la EDO mediante la programación en Python y se graficará, en el mismo lenguaje, la solución para las dos condiciones iniciales planteadas de forma de visualizar gráficamente la estabilidad de la solución.

2.2 Resolución en Python

Para facilitar la escritura del código en Python se trabajará en el entorno de Google Colaboratory[1] (Google Colab de aquí en adelante) que permite programar y ejecutar Python en el navegador sin que sea necesaria su instalación. Además, al usar Google Colab Google asigna aproximadamente 12GB de memoria RAM para ejecutar los procesos, lo cual asegura un estándar de capacidad de procesamiento independientemente del hardware con el que se esté trabajando.

Una vez abierto el cuaderno de Google Colab se debe instalar e importar las bibliotecas que se utilizarán. Se instalarán tres librerías, la librería Sympy que permite realizar una manipulación simbólica de fórmulas y ecuaciones de forma de poder ingresar las EDO y resolverlas; la librería Matplotlib destinada para la creación de gráficos, en este caso se utilizará el módulo Pyplot y por último la librería Numpy que está especializada en cálculo numérico y en el análisis de datos.

El código de instalación de las librerías es:

```
!pip install numpy
!pip install matplotlib
!pip install sympy
```

Cabe aclarar que el signo de exclamación antes de cada sentencia es solamente necesario si el código se ejecuta desde Google Colab. Si se optara por realizar las instalaciones desde la consola no sería necesario.

Una vez instaladas las librerías es necesario importarlas, es decir, traer los paquetes de funciones que vamos a usar para nuestro trabajo. En este caso las sentencias son:

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

Al usar la sentencia "as" lo que se hace es abreviar el nombre de la librería para que sea más sencillo llamarla cada vez que se la utilice. Esto es una práctica usual cuando se trabaja con las librerías más populares.

Para resolver las EDOs se utiliza la librería Sympy de Python en donde el primer paso es declarar las variables simbólicas que se utilizarán, en este caso la variable independiente tiempo (t) y la función de precio (p). El código para declarar las variables es:

```
t = sympy.Symbol('t')
p = sympy.Function('p')
```

[1]Para más información sobre Google Colaboratory: https://colab.research.google.com/notebooks/welcome.ipynb?hl=es#scrollTo=5fCEDCU_qrC0

El próximo paso es declarar la ecuación diferencial del modelo. Para esto es necesario conocer el código para ingresar derivadas y ecuaciones.

Para derivar una función en Sympy se usa la función diff(): `sympy.diff(func, var,n)`

Donde los tres argumentos utilizados son:

- func: la función que se va a derivar
- var: la variable respecto a la cual se deriva
- n: orden de la derivada (por defecto, con la primera derivada no es necesario ingresar este argumento)

Por ejemplo, si se quisiera realizar la primera derivada de p con respecto a t se debería plantear:

sympy.diff(p(t), t, 1)

Se puede observar que para llamar a la función diff, como esta pertenece al paquete sympy se tuvo que declarar también el paquete junto con un punto para precisar de dónde proviene la función diff.

Para definir la ecuación diferencial se utiliza el elemento Eq (Equality), en donde la estructura es la siguiente:

```
Eq(lado izquierdo de la ecuación, lado derecho de la ecuación)
```

Por lo tanto, para ingresar la EDO dada: $p''(t) + 3p'(t) + 2p(t) = 10$ en código Python:

```
sympy.Eq(sympy.diff(p(t),t,2)+3*sympy.diff(p(t),t)+2*p(t),10)
```

Por practicidad, se asigna el objeto encontrado a una variable, por ejemplo, en este caso a la variable EDO_1:

```
EDO_1=sympy.Eq(sympy.diff(p(t),t,2)+3*sympy.diff(p(t),t)+2*p(t),10)
```

Para visualizar, la EDO_1 en consola es necesario invocar a la variable que se creó: `EDO_1`

De esta forma, lo que se visualiza en consola es la EDO dada de la siguiente forma:

$$2p(t) + 3\frac{d}{dt}p(t) + \frac{d^2}{dt^2}p(t) = 10$$

Para resolver esta ecuación diferencial se utiliza la función dsolve(), cuya sintaxis más básica es:

```
dsolve(eq, f)
```

En donde: eq: es una expresión de la clase ecuación (Equality)

f: función incógnita con respecto a la cual se resuelve la EDO dada, en general es un objeto de la clase Function.

En el ejemplo planteado:

```
sympy.dsolve(EDO_1, p(t))
```

Si se asigna la solución encontrada a una variable, por ejemplo: "sol_gral", de forma que cada vez que se quiera utilizar la solución encontrada se invoque a la variable por su nombre:

```
sol_gral = sympy.dsolve(EDO_1, p(t))
```

Si se desea visualizar la solución de la EDO dada se debe invocar a la variable a la que se asignó la solución. De esta forma se visualizará en la consola la solución general de la ecuación diferencial dada:

$$p(t) = C_1 e^{-2t} + C_2 e^{-t} + 5$$

Asimismo, es posible hallar la solución particular de la EDO dada, para ello se vuelve a utilizar la función dsolve en donde los dos primeros argumentos son los presentados previamente y el tercer argumento es un diccionario de Python en donde se deben listar las condiciones iniciales que se tomarán en cuenta para hallar la solución particular. La estructura general es la siguiente:

```
sympy.dsolve(EDO_1, p(t), ics=CI)
```

Donde los primeros dos argumentos son los presentados previamente y el tercer argumento es un diccionario de Python en donde se listan las condiciones iniciales. En el caso de la EDO de segundo orden dada el diccionario tiene dos elementos, para las condiciones iniciales p(0)=1 y p´(0)=1 el diccionario posee la siguiente estructura que lo asignamos a la variable CI_1:

```
CI_1 = {p(0):1, sympy.diff(p(t),t).subs(t,0):1}
```

Luego que se ingresó el diccionario con los dos elementos, se busca la solución particular:

```
sol_part=sympy.dsolve(EDO_1, p(t), ics=CI_1)
```

Que se visualiza en consola de la siguiente forma:

$$p(t) = 5 - 7e^{-t} + 3e^{-2t}$$

De la misma forma para la otra condición inicial dada: p(0)=8 y p´(0)=1

```
CI_2 = {p(0):8, sympy.diff(p(t),t).subs(t,0):1}
sol_part=sympy.dsolve(EDO_1, p(t), ics=CI_2)
sol_part
```

Cuya visualización en consola:

$$p(t) = 5 + 7e^{-t} - 4e^{-2t}$$

Hasta acá utilizando solo la librería Sympy de Python se ha podido hallar la solución general y la solución particular de una EDO de segundo orden.

El análisis de la estabilidad de la solución se puede realizar, como se hizo previamente, analizando el signo de las raíces de la ecuación característica, como se concluyó en el apartado anterior si ambas raíces son negativas se puede asegurar que la solución es dinámicamente estable. Didácticamente resulta relevante graficar las trayectorias del precio para las distintas condiciones iniciales y visualizar lo que se ha concluido analíticamente.

Como se busca visualizar las dos soluciones particulares encontradas, lo primero que se debe hacer es definirlas, en este caso, se definen dos funciones: p1(t) y p2(t). Donde p1 es el nombre de la función y t es el argumento de esta y de la misma forma p2 es el nombre de la segunda función y t su argumento:

```
def p1(t):
        return 5 - 7*np.exp(-t) + 3*np.exp(-2*t)
def p2(t):
        return 5 + 7*np.exp(-t) - 4*np.exp(-2*t)
```

Para realizar la gráfica en Python, se utilizará Pyplot de la librería Matplotlib. Se elige un gráfico de línea en donde el dominio lo indicaremos con la función linspace de la librería Numpy que genera una clase de objeto denominado array (colecciones de datos de un mismo tipo) donde la sintaxis de la función es:

```
np.linspace(valor inicial,valor final,número de valores)
```

En este caso, por ejemplo, a la variable t se le asigna un objeto array con 100 valores que se inician en el 0 y finalizan en el 8:

```
t=np.linspace(0,8, 100)
```

De esta forma, se grafican las funciones p1(t) y p2(t) utilizando la función Plot con las siguientes sentencias:

```
plt.plot(t,p1(t), color="b")
plt.plot(t,p2(t), color="g")
```

Si se desea, se puede incorporar el título al gráfico y una leyenda a los ejes coordenados:

```
plt.title("Trayectoria del precio")
plt.xlabel("Tiempo (t)")
plt.ylabel("Precio p(t)")
```

Para el análisis de la estabilidad es importante indicar el precio de equilibrio en el gráfico. Para eso podemos incorporar una recta horizontal e indicar un texto:

```
plt.axhline(5, color='r', linestyle = 'dashed')
plt.text(1, 5, 'precio equilibrio', fontsize= 8, va='center
', ha='center', color='r', backgroundcolor='w')
```

Como las dos trayectorias del precio van a aparecer en el mismo gráfico es posible incorporar una leyenda con las condiciones iniciales para poder diferenciarlas:

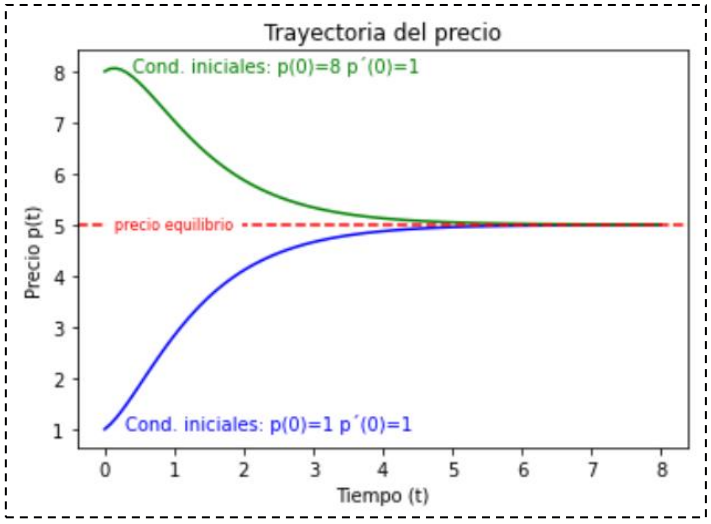
```
plt.text(0.3,1,'Cond. iniciales: p(0)=1 p´(0)=1', color="b")
plt.text(0.4,8,'Cond. iniciales: p(0)=8 p´(0)=1', color="g")
```

Para visualizar el gráfico en la consola, la sentencia es:

```
plt.show()
```

El gráfico que se visualizará es:

Gráfico 1: Trayectoria del precio realizada con Python



Fuente: Elaboración Propia

En el gráfico se observa en la curva de color verde la trayectoria del precio cuando las condiciones iniciales son p(0)=8 y p´(0)=1y en azul la trayectoria cuando las condiciones iniciales son p(0)=1 y p´(0)=1. Ambas curvas convergen al precio de equilibrio que está indicado con la recta horizontal roja punteada (en este caso el precio de equilibrio es 5). Se visualiza así dos casos en donde, aunque cambian las condiciones iniciales como la solución es dinámicamente estable el precio converge al precio de equilibrio.

A continuación, se analizará un modelo de mercado inestable de forma de visualizar gráficamente la solución.

3. EJEMPLO II: MODELO DE MERCADO INESTABLE

El siguiente modelo describe un mercado en donde tanto los compradores como los vendedores basan su comportamiento en el precio presente y su tendencia futura:

$$\begin{cases} Q_d = 8 - 3p(t) - p'(t) + p''(t) \\ Q_s = -4 + 3p(t) - 8p'(t) + 2p''(t) \\ Q_d = Q_s \end{cases} \quad (8)$$

Donde:

Q_d: $Cantidad\ demandada\ del\ bien$
Q_s: $Cantidad\ ofrecida\ del\ bien$
p: $precio\ del\ bien$
Q_d: $Cantidad\ demandada\ del\ bien$
Q_s: $Cantidad\ ofrecida\ del\ bien$
p: $precio\ del\ bien$
$p'(t) = \frac{dp}{dt}$
$p''(t) = \frac{d^2p}{dt^2}$

La ecuación diferencial resultante del modelo (8) es una ecuación diferencial a coeficientes constantes:

$$p''(t) - 7p'(t) + 6p(t) = 12 \quad (9)$$

Se desea encontrar la trayectoria del precio en modelo dado cuando las condiciones iniciales son $p(0) = 3\ y\ p'(0) = 2.$ A continuación, se hallará analíticamente la solución general y las soluciones particulares para el problema planteado.

3.1 Resolución analítica

Para resolver analíticamente la ecuación diferencial (9), se debe hallar la solución homogénea y la solución complementaria.

Solución homogénea:

Se parte de la ecuación característica asociada a la ecuación diferencial homogénea:

$$r^2 - 7r + 6 = 0$$

Cuyas raíces son: 1 y 6, por lo tanto, la solución homogénea es: $p_h(t) = C_1e^t + C_2e^{6t}$

Solución complementaria:

La solución complementaria se resuelve con el método de los coeficientes indeterminados. Se propone como solución complementaria $p_c(t) = A$ y se determina el coeficiente al reemplazar, la solución propuesta y sus sucesivas derivadas, en la ecuación diferencial (9):

$$6A = 12 \implies A = 2$$

Por lo tanto, la solución complementaria es: $p_c(t) = 2$

La solución general: $p_g(t) = C_1e^t + C_2e^{6t} + 2$

Como las raíces de la ecuación característica son positivas se puede concluir que la solución es inestable. Se debe recordar que la condición para asegurar la estabilidad de la solución es que todas las raíces de la ecuación característica deben ser negativas.

Si la condición inicial es: $p(0) = 3$ y $p'(0) = 2$ La solución particular es:

$$p_p(t) = \frac{4}{5}e^t + \frac{1}{5}e^{6t} + 2$$

A continuación, igual que como se hizo con el ejemplo 1 pero en forma más concisa, se describirá como resolver esta EDO mediante la programación en Python y se graficará la solución para condición inicial planteada de forma de visualizar gráficamente la solución inestable.

3.2 Resolución en Python

Al igual que el ejemplo previo se deben tener instaladas e importadas las librerías que se utilizarán:

```
!pip install numpy
!pip install matplotlib
!pip install sympy
```

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

Se deben declarar las variables a utilizar:

```
t = sympy.Symbol('t')
p = sympy.Function('p')
```

El próximo paso es declarar la ecuación diferencial del modelo. Por lo tanto, para ingresar la EDO del ejemplo 2: $p''(t) - 7p'(t) + 6p(t) = 12$ y asignarlo a la variable EDO_2

```
EDO_2=sympy.Eq(sympy.diff(p(t),t,2)-7*sympy.diff(p(t),t)+6*p(t),12)
```

Para visualizar, la EDO_2 en consola es necesario invocar a la variable que se creó: `EDO_2`. De esta forma, lo que se visualiza en consola es la EDO dada de la siguiente forma:

$$6p(t) - 7\frac{d}{dt}p(t) + \frac{d^2}{dt^2}p(t) = 12$$

Para resolver esta ecuación diferencial se utiliza la función dsolve() y se asigna la solución a la variable "sol_gral"

```
sol_gral = sympy.dsolve(EDO_2, p(t))
```

Si se desea visualizar la solución general se debe invocar a la variable a la que se asignó la solución. De esta forma se visualizará en la consola la solución general de la ecuación diferencial dada:

$$p(t) = C_1 e^{t} + C_2 e^{6t} + 2$$

Para hallar la solución particular, se debe ingresar un diccionario con las dos condiciones iniciales, en este caso: $p(0) = 3\ y\ p'(0) = 2$

```
CI_3 = {p(0):3, sympy.diff(p(t),t).subs(t,0):2}
```

Luego que se ingresa el diccionario con los dos elementos, se busca la solución particular:

```
sol_part=sympy.dsolve(EDO_2, p(t), ics=CI_3)
```

Que se visualiza en consola de la siguiente forma:

$$p(t) = \frac{e^{6t}}{5} + \frac{4e^{t}}{5} + 2$$

Para complementar el análisis de estabilidad se procede a graficar la trayectoria del precio encontrada. Para esto primero se define la función p3(t):

```
def p3(t):
        return (1/5)*np.exp(6*t) +(4/5)*np.exp(t)+2
```

Luego se grafica la función p3(t):

```
t=np.linspace(0,0.5,100)
plt.plot(t,p3(t))
```

Si se desea, se puede incorporar el título al gráfico y una leyenda a los ejes coordenados:

```
plt.title("Trayectoria del precio")
plt.xlabel("Tiempo (t)")
plt.ylabel("Precio p(t)")
```

Para el análisis de la estabilidad es importante indicar el precio de equilibrio en el gráfico. Para eso podemos incorporar una recta horizontal e indicar un texto:

```
plt.axhline(2, color='r', linestyle = 'dashed')
plt.text(0.08, 2, 'precio equilibrio=2', fontsize= 8, va='c
enter', ha='center', color='r', backgroundcolor='w')
```

También se puede incluir otro texto con las condiciones iniciales utilizadas para calcular la trayectoria del precio:

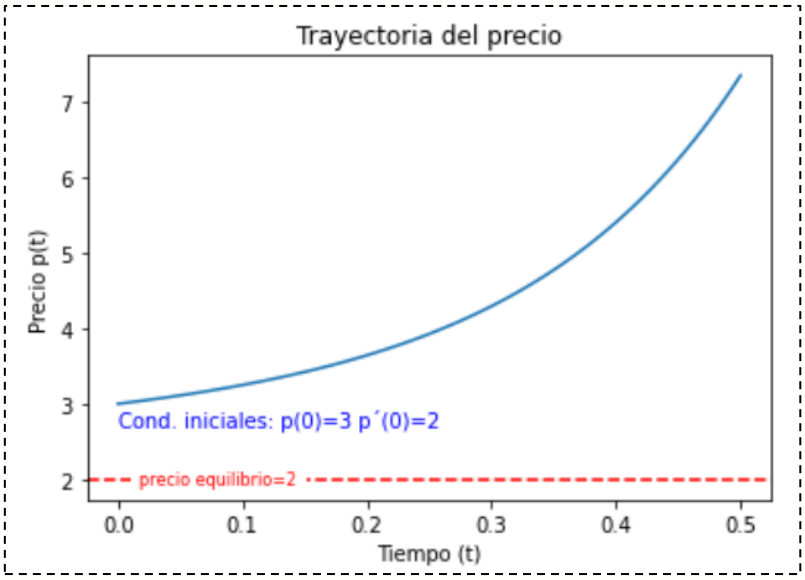
```
plt.text(0,2.7,'Cond. iniciales: p(0)=3 p´(0)=2', color="b"
)
```

Para visualizar el gráfico en consola:

```
plt.show ()
```

En el Gráfico 2 se visualiza la salida en consola:

Gráfico 2: Trayectoria del precio realizada con Python



Fuente: Elaboración Propia

En el gráfico, la curva azul representa la trayectoria del precio cuando las condiciones iniciales son p(0)=3 y p´(0)=2, se observa que el precio no converge al precio de equilibrio que en este ejemplo es 2.

CONCLUSIÓN

En este artículo se presenta una introducción a la programación en el lenguaje Python de forma de aprender diferentes herramientas del mismo a través de la resolución de conceptos matemáticos ya vistos por los estudiantes de Ciencias Económicas. Para esto se desarrollaron dos ejemplos numéricos del modelo de mercado dinámico en donde la resolución de ambas ecuaciones diferenciales resultantes se realizó con la librería Sympy de Python y los gráficos correspondientes con Matplotlib.

Esta actividad permite introducir al estudiante en la programación de lenguaje Python y a su vez suplementar los conceptos de ecuaciones diferenciales y estabilidad de la solución vistos previamente en clase, de forma tal que ambos conocimientos se potencien en la actividad propuesta ya que las herramientas básicas de Python se empiezan a utilizar con conceptos aprendidos por los estudiantes y los conceptos aprendidos se aplican en nuevas situaciones.

REFERENCIAS

BERNARDELLO, A.; BIANCO, M.J.; CASPARRI, M.T.; GARCÍA FRONTI, J.; MARZANA, S. (2010) Matemática para Economistas con Excel y Matlab. Editorial Omicron System, Buenos Aires.

CHIANG, A. y WAINWRIGHT, K. (2006). Métodos fundamentales de economía matemática (4a. ed.). Editorial: McGraw-Hill, México.

DI CARO, H.; GALLEGO, L (2000) Análisis Matemático II con aplicaciones a las Ciencias Económicas. Editorial Macchi. Buenos Aires.

KUROKI, M. (2021). *Using Python and Google Colab to teach undergraduate microeconomic theory*. *International Review of Economics Education*, *38*, 100225.

Librerías utilizadas:

Sympy: https://www.sympy.org/es/

Numpy: https://numpy.org/doc/stable/user/index.html#user

Matplotlib: https://matplotlib.org/