Matemática para Ciencia de los Datos

Trabajo Práctico 1

Profesor: Luis Alexánder Calvo Valverde

Instituto Tecnológico de Costa Rica,

Programa Ciencia de Datos

Fecha de entrega: Lunes 24 de Abril del 2023, a más tardar a las 3:00 pm.

Medio de entrega: Por medio del TEC-Digital.

Entregables: Un archivo jupyter (.IPYNB).

Estudiante(s):

1. Ricardo Chacón Brenes
2. Gabriel Valentine Fonseca

```
%matplotlib inline
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
```

Pregunta 1 (20 puntos, 10 pts c/u)

Demuestre de manera matemática si los siguientes sistemas $L\{\cdot\}$ (con entrada $u(t)$ y salida $g(t)$) son lineales o no lineales (**escriba las fórmulas en celdas de texto**). Una vez hecho su mayor esfuerzo, si no sabe cómo seguir matemáticamente, puede sustituir con valores y mostrar por contra-ejemplo.

a)
$g(t) = \log_{10} u(t)$

$L\{\alpha f_1(x) + \beta f_2(x)\} = \alpha L\{f_1(x)\} + \beta L\{f_2(x)\}$

$\log_{10}(\alpha u_1(t) + \beta u_2(t)) =? \alpha \log_{10} u_1(t) + \beta \log_{10} u_2(t)$

$\log_{10}(\alpha u_1(t) + \beta u_2(t)) =? \log_{10}(u_1(t)^{\alpha} u_2(t)^{\beta})$

No es un sistema lineal

b)
$g(t) = 5 * u(t) + 13$

$L\{\alpha f_1(x) + \beta f_2(x)\} = \alpha L\{f_1(x)\} + \beta L\{f_2(x)\}$

$5(\alpha u_1(t) + \beta u_2(t)) + 13 =? \alpha(5u_1(t) + 13) + \beta(5u_2(t) + 13)$

$5\alpha u_1(t) + 5\beta u_2(t) + 13 =? 5\alpha u_1(t) + 13\alpha + 5\beta u_2(t) + 13\beta$

$5\alpha u_1(t) + 5\beta u_2(t) + 13 =? 5\alpha u_1(t) + 5\beta u_2(t)\, 13(\alpha + \beta)$

No es un sistema lineal

Pregunta 2 (20 puntos, 10 pts c/u)

Para cada uno de los siguientes vectores calcule la norma L_2:

1. De manera matemática.
2. Programe una implementación en python de lo anterior, pero sin utilizar la función **norm** de la biblioteca.
3. Luego compare su resultado con una versión usando **norm**.

a)

$$a = \begin{bmatrix} -9 \\ 7 \end{bmatrix}$$

1. a = $\sqrt{(-9)^2 + (7)^2}$ = $\sqrt{130}$ = 11.410175

```
# 2. Sin norm

a = [-9, 7]
norma2 = 0
for i in a:
  norma2 += i**2

norma2 = norma2**0.5
print(norma2)
```

```
11.40175425099138
```

```
# 3. Con norm

import numpy as np

a = np.array ([-9. , 7])
norma3 = np.linalg.norm(a)
print(norma3)
```

```
11.40175425099138
```

b)

$$b = \begin{bmatrix} 1 \\ 2 \\ 4 \\ 5 \end{bmatrix}$$

1. a = $\sqrt{(1)^2 + (2)^2 + (-4)^2 + (5)^2}$ = $\sqrt{46}$ = 6.78233

```
# 2. Sin norm

a2 = [1, 2, 4, 5]
norma2 = 0
for i in a2:
  norma2 += i**2

norma2 = norma2**0.5
print(norma2)
```

```
6.782329983125268
```

```
# 3. Con norm

import numpy as np

a3 = np.array ([1, 2, 4, 5])
norma3 = np.linalg.norm(a3)
print(norma3)
```

```
6.782329983125268
```

Pregunta 3 (20 puntos, 10 pts c/u)

En Python, calcule el producto punto (o producto escalar entre vectores) $a \cdot b$ para los siguientes pares de vectores, una versión utilizando **dot**, y otra sin utilizar dicha función (programa en python con ciclos):

a)

$$a = \begin{bmatrix} 2 \\ 7 \end{bmatrix}, b = \begin{bmatrix} 10 \\ 2 \end{bmatrix}$$

```
# Sin dot:
a = [2 , 7]
b = [10 , 2]

producto_punto = 0
for i in range(len(a)):
    producto_punto += a[i] * b[i]

print(producto_punto)
```

```
34
```

```
import numpy as np

#Con dot:
a = np.array([2 , 7])
b = np.array([10 , 2])
producto_punto_dot = np.dot(a, b)

print(producto_punto_dot)
```

```
34
```

b)

$$a = \begin{bmatrix} -1 \\ 8 \\ 3 \end{bmatrix}, b = \begin{bmatrix} 2 \\ 6 \\ 5 \end{bmatrix}$$

```
# Sin dot:
a = [-1, 8, 3]
b = [2, 6, 5]

producto_punto = 0
for i in range(len(a)):
    producto_punto += a[i] * b[i]

print(producto_punto)
```

```
61
```

```
#Con dot:

import numpy as np

a = np.array([-1, 8, 3])
b = np.array([2, 6, 5])
producto_punto_dot = np.dot(a, b)

print(producto_punto_dot)
```

```
61
```

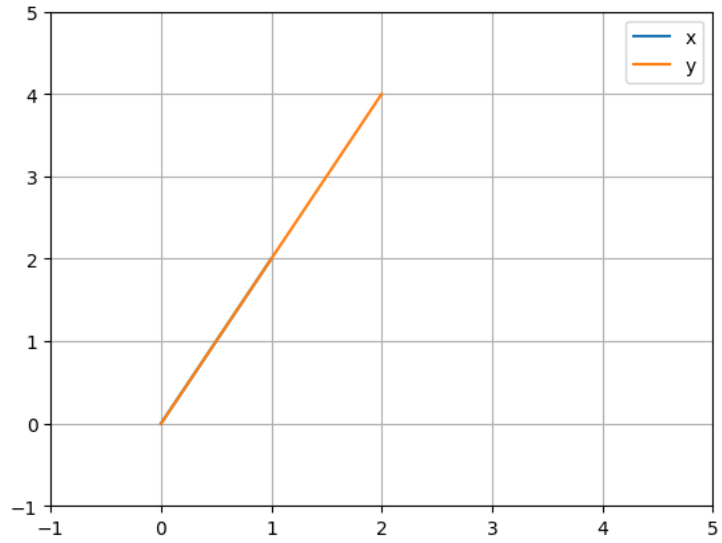
Pregunta 4 (20 puntos, 10 pts c/u)

a) Proponga dos vectores: x e y que sean colineales (con dos elementos cada uno). Programe en python para mostrar que son colineales y luego grafíquelos en un mismo gráfico en python. En el cuaderno visto en clase viene un ejemplo de uso de import matplotlib.pyplot as plt

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

# Definimos los vectores
x = np.array([1, 2])
y = np.array([2, 4])

# Verificamos que son colineales
if np.all(x * 2 == y):
    print("Los vectores son colineales")

# Graficamos los vectores
plt.plot([0, x[0]], [0, x[1]], label="x")
plt.plot([0, y[0]], [0, y[1]], label="y")
```

```
plt.legend()
plt.xlim(-1, 5)
plt.ylim(-1, 5)
plt.grid()
plt.show()
```

```
Los vectores son colineales
```

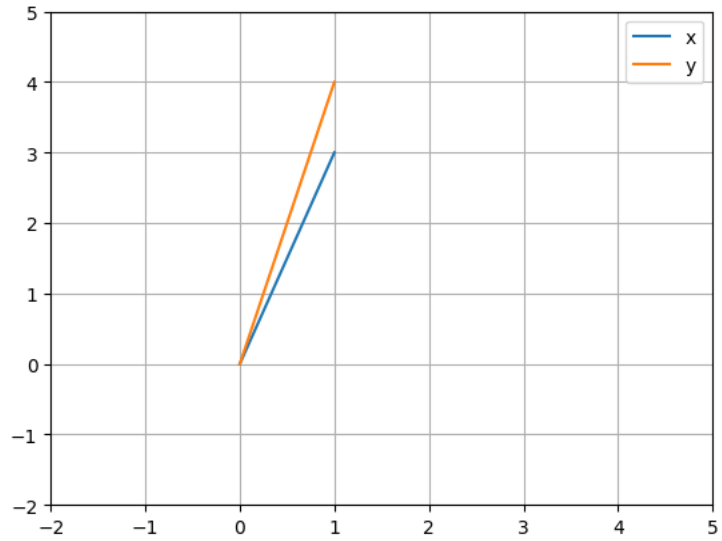


b)

Ahora modifique uno de los vectores para que no sean colineales y luego grafique para mostrar cómo se ven dos vectores que no son colineales.

```
# Definimos los vectores
x = np.array([1, 3])  # Vector modificado
y = np.array([1, 4])

# Verificamos que no son colineales
if np.all(x * 2 != y):
    print("Los vectores no son colineales")

# Graficamos los vectores
plt.plot([0, x[0]], [0, x[1]], label="x")
plt.plot([0, y[0]], [0, y[1]], label="y")
plt.legend()
plt.xlim(-2, 5)
plt.ylim(-2, 5)
plt.grid()
plt.show()
```

```
Los vectores no son colineales
```



Pregunta 5 (20 puntos, 10 pts c/u)

a) Cargue el archivo llamado "Datos_01.csv".

¿Existen atributos colineanes? ¿Cuáles? Programe en python para mostrar su respuesta.

```
from google.colab import files
from google.colab import drive
drive.mount('/content/drive')
```

```
Drive already mounted at /content/drive; to attempt to forcibly remount, call drive.mount("/content/drive", force_remount=True).
```

```
import numpy as np
import pandas as pd

separador = "-"*40
archivo = "/content/drive/MyDrive/Colab Notebooks/Módulo 1/Tareas/Tarea 1/Datos_01.csv"
# carga el archivo en pandas
dataFrame = pd.read_csv(archivo, header = 0, delimiter=';')

# guarda el nombre de las columnas en una lista
colNames = dataFrame.columns

# muestra los primeros elementos del dataFrame
print(separador)
print("Datos en dataFrame:")
print(dataFrame.head() )

# Convertir de pandas a numpy
datos = pd.DataFrame(dataFrame).to_numpy()

# En cada vector columna hay un atributo
# El atributo1 está en datos[:,0]
# El atributo2 está en datos[:,1]
# y así sucesivamente

# Programar para determinar vectores colineales

def son_colineales(v1, v2):
    return np.array_equal(v1/v1[0], v2/v2[0])

for i in range(datos.shape[1]):
    for j in range(i+1, datos.shape[1]):
        if son_colineales(datos[:,i], datos[:,j]):
            print(f"Los atributos {i+1} y {j+1} son colineales.")
```

```
----------------------------------------
Datos en dataFrame:
   atributo1  atributo2  atributo3  atributo4  atributo5
0        515         15   0.408462     1287.5          8
1        357         22   0.642985      892.5          9
2        633         20   0.582240     1582.5          7
3        295         17   0.531009      737.5          9
4        946         14   0.340640     2365.0          3
Los atributos 1 y 4 son colineales.
```

b)

En el archivo "reales.csv" se encuentran los valores reales de un conjunto de datos, y en el archivo "predicciones.csv" lo que predijo un algoritmo.

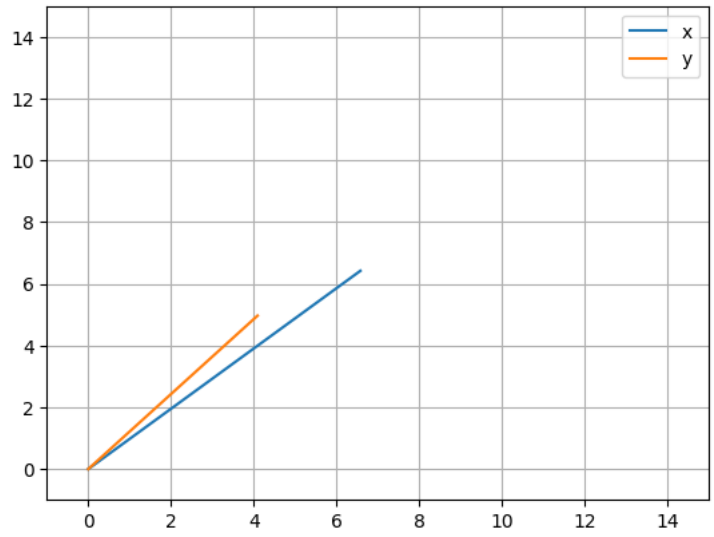
Cargue ambos archivos y muestre la norma 2 y la norma 5, de la diferencia entre el real y el predicho.

Finalmente, grafique el predicho y el real en un mismo gráfico para comparar.

```
import numpy as np
import pandas as pd

# carga archivos

archivo = "/content/drive/MyDrive/Colab Notebooks/Módulo 1/Tareas/Tarea 1/reales.csv"
dataFrameReales = pd.read_csv(archivo, header = 0)
```

```
archivo = "/content/drive/MyDrive/Colab Notebooks/Módulo 1/Tareas/Tarea 1/predicciones.csv"
dataFramePredichos = pd.read_csv(archivo, header = 0)


# Convertir de pandas a numpy

reales = dataFrameReales.to_numpy().squeeze()
predicciones = dataFramePredichos.to_numpy().squeeze()

# Calcular las norma 2

norm2 = np.linalg.norm(reales - predicciones,2)
print(norm2)

# Calcular las norma 5

norm5 = np.linalg.norm(reales - predicciones,5)
print(norm5)

# Graficar
import matplotlib.pyplot as plt

plt.plot([0, reales[0]], [0, reales[1]], label="x")
plt.plot([0, predicciones[0]], [0, predicciones[1]], label="y")
plt.legend()
plt.xlim(-1, 15)
plt.ylim(-1, 15)
plt.grid()
plt.show()
```

```
72.9246404509477
13.38063754748088
```



✓ 0s completed at 11:00 PM