Instituto Tecnológico de Costa Rica

IC6400 | Investigación De Operaciones

Grupo 20

Examen 2

Subgrupo 5:

Valery Carvajal Oreamuno - Carné: 2022314299

Jesús Gabriel Cordero Díaz - Carné: 2020081049

Anthony Josué Rojas Fuentes - Carné: 2018027141

Profesor:

Manuel Alejandro Mendez Flores

Semestre II

2024

Índice de Contenido

INSTITUTO TECNOLÓGICO DE COSTA RICA	**Examen: Parcial II** **Código: IC-6400**
Escuela de Ingeniería en Computación	**Duración: 3 horas**
Curso: Investigación de Operaciones I	**Fecha: 5-11-2024**
Profesor: Ing. Manuel Méndez Flores, MSc.	**Hora: 8:15 a.m.**
Valor de la Prueba 10 %	**Puntos Totales: 60 puntos**
Estudiante: Valery Carvajal Oreamuno Jesús Gabriel Cordero Díaz Anthony Josué Rojas Fuentes	**Identificación:** 2022314299 2020081049 2018027141

INSTRUCCIONES:

1. La prueba es grupal.
2. Dispone de 3 horas para realizar la prueba escrita.
3. Suba su respuesta al Tec Digital
4. La prueba escrita consta de:

Rúbrica de cada ejercicio	Valor	Puntos Obtenidos
Programación del Modelo en R o Python (video con descripción de los resultados)	5	5 = correcto, 3 = incompleto, 0 = no presentó
Respuesta con los cálculos correctos	5	5 = correcto, 3 = incompleto, 0 = no presentó
Análisis de los resultados	5	5 = correcto, 3 = incompleto, 0 = no presentó
Video breve mostrando los resultados (1 minuto de vídeo por ejercicio).	5	5 = 3 videos
Total 3 ejercicios	60	

Problema 1.

Programe y resuelva la solución del problema de transporte (5 Puntos)

MG Auto cuenta con tres plantas en Los Ángeles, Detroit y Nueva Orleáns, y dos importantes centros de distribución en Denver y Miami. Las capacidades trimestrales de las tres plantas son 1000, 1500 y 1200 automóviles, y las demandas de los dos centros de distribución durante el mismo periodo son de 2300 y 1400 automóviles. La distancia en millas entre las plantas y los centros de distribución aparece en la tabla 5.1.

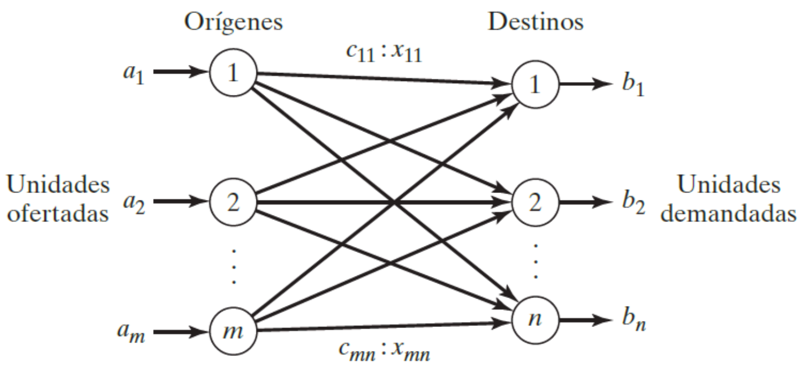


FIGURA 5.1

Representación del modelo de transporte con nodos y arcos

TABLA 5.1 Gráfica de distancia en millas

	Denver	Miami
Los Ángeles	1000	2690
Detroit	1250	1350
Nueva Orleáns	1275	850

La compañía transportista cobra 8 centavos por milla por automóvil. En la tabla 5.2 se dan los costos de transporte por automóvil en las diferentes rutas, redondeados al dólar más cercano.

El modelo de PL del problema es

$$\text{Minimizar } z = 80x_{11} + 215x_{12} + 100x_{21} + 108x_{22} + 102x_{31} + 68x_{32}$$

sujeto a

$$\begin{array}{llllllll}
x_{11} & + x_{12} & & & & & = 1000 & \text{(Los Ángeles)} \\
 & & x_{21} & + x_{22} & & & = 1500 & \text{(Detroit)} \\
 & & & & + x_{31} & + x_{32} & = 1200 & \text{(Nueva Orleáns)} \\
x_{11} & & + x_{21} & & + x_{31} & & = 2300 & \text{(Denver)} \\
 & x_{12} & & + x_{22} & & + x_{32} & = 1400 & \text{(Miami)}
\end{array}$$

$$x_{ij} \geq 0, i = 1, 2, 3, \ j = 1, 2$$

TABLA 5.2 Costo de transporte por automóvil

	Denver (1)	Miami (2)
Los Ángeles (1)	\$80	\$215
Detroit (2)	\$100	\$108
Nueva Orleáns (3)	\$102	\$68

A. Código en Python3 y

```
import matplotlib.pyplot as plt

# Capacidades de las plantas
capacidad = [1000, 1500, 1200]

# Demandas de los centros de distribución
demanda = [2300, 1400]

# Costos de transporte entre plantas y centros de distribución
costos = [
    [80, 215],    # Los Angeles -> Denver, Miami
    [100, 108],   # Detroit -> Denver, Miami
    [102, 68]     # Nueva Orleans -> Denver, Miami
]

# Matriz para almacenar la cantidad de autos transportados en
cada ruta
x = [[0, 0], [0, 0], [0, 0]]

# Copiamos capacidad y demanda para no modificar las originales
cap = capacidad[:]
dem = demanda[:]

# Metodo de Esquina Noroeste
for i in range(len(cap)):
    for j in range(len(dem)):
        cantidad = min(cap[i], dem[j])
        x[i][j] = cantidad
        cap[i] -= cantidad
        dem[j] -= cantidad
```

```
def costo_total(x, costos):
    total = 0
    for i in range(len(x)):
        for j in range(len(x[i])):
            total += x[i][j] * costos[i][j]
    return total


def optimizar_modi(x, costos):
    # Inicializamos multiplicadores de filas y columnas
    u = [None] * len(x)
    v = [None] * len(x[0])
    u[0] = 0  # Fijamos el primer multiplicador de fila a 0

    while True:
        # Actualizar u y v con base en las rutas básicas (donde
x > 0)
        for i in range(len(x)):
            for j in range(len(x[i])):
                if x[i][j] > 0:  # Ruta básica
                    if u[i] is not None and v[j] is None:
                        v[j] = costos[i][j] - u[i]
                    elif u[i] is None and v[j] is not None:
                        u[i] = costos[i][j] - v[j]

        # Calcular los costos reducidos para las celdas no
básicas
        costos_reducidos = [[0] * len(x[0]) for _ in
range(len(x))]
        for i in range(len(x)):
            for j in range(len(x[i])):
                if x[i][j] == 0:  # Celda no básica
                    costos_reducidos[i][j] = costos[i][j] -
(u[i] + v[j])

        # Si todos los costos reducidos son >= 0, la solución es
óptima
        min_costo_reducido = min(min(fila) for fila in
costos_reducidos)
        if min_costo_reducido >= 0:
```

```
            break

        # Encuentra la celda con el costo reducido más negativo
        min_pos = None
        for i in range(len(costos_reducidos)):
            for j in range(len(costos_reducidos[i])):
                if costos_reducidos[i][j] < min_costo_reducido:
                    min_costo_reducido = costos_reducidos[i][j]
                    min_pos = (i, j)

# Se requiere encontrar un camino cerrado en la tabla de
asignación `x`

    return x

# Solución inicial usando el método de esquina noroeste
solucion_inicial = [fila[:] for fila in x]

# Optimizar usando el método MODI
solucion_optima = optimizar_modi(solucion_inicial, costos)

# Calcular el costo total
costo_minimo = costo_total(solucion_optima, costos)
print("Costo mínimo total:")
print("Zmin =", costo_minimo)
print("Distribución óptima de transporte (autos en cada
ruta):")
for i in range (len(solucion_optima)):
    for j in range (len(solucion_optima[i])):
        print("x" + str(i+1) + str(j+1) + " = " +
str(solucion_optima[i][j]))

# Datos de posiciones para cada nodo
plantas = {
    "Los Angeles": (0, 3),
    "Detroit": (0, 1.5),
    "Nueva Orleans": (0, 0)
}

centros = {
    "Denver": (3, 2.5),
```

```
    "Miami": (3, 0.5)
}

# Configuración de los nombres de las rutas
rutas = [
    ("Los Angeles", "Denver", solucion_optima[0][0]),
    ("Los Angeles", "Miami", solucion_optima[0][1]),
    ("Detroit", "Denver", solucion_optima[1][0]),
    ("Detroit", "Miami", solucion_optima[1][1]),
    ("Nueva Orleans", "Denver", solucion_optima[2][0]),
    ("Nueva Orleans", "Miami", solucion_optima[2][1]),
]

# Crear el gráfico
plt.figure(figsize=(14, 8))
plt.title("Rutas de Transporte Óptimas")

# Graficar nodos de plantas y centros de distribución
for nombre, (x, y) in plantas.items():
   plt.plot(x, y, 'bo')  # Nodo de planta
   plt.text(x - 0.3, y, nombre, ha="right", va="center",
fontsize=12)

for nombre, (x, y) in centros.items():
   plt.plot(x, y, 'ro')  # Nodo de centro de distribución
   plt.text(x + 0.3, y, nombre, ha="left", va="center",
fontsize=12)

# Graficar rutas con flechas y etiquetas de cantidad
for origen, destino, cantidad in rutas:
   x_origen, y_origen = plantas[origen] \
       if origen in plantas \
       else centros[origen]
   x_destino, y_destino = plantas[destino] \
       if destino in plantas \
       else centros[destino]
   plt.arrow(x_origen, y_origen, x_destino - x_origen,
y_destino - y_origen,
             head_width=0.1, length_includes_head=True,
color="gray")
   plt.text((x_origen + x_destino) / 2, (y_origen + y_destino)
```

```
/ 2, f"{cantidad} autos",
            ha="center", va="center", fontsize=10,
color="darkgreen")

# Mostrar el gráfico
plt.xlabel("Coordenadas X")
plt.ylabel("Coordenadas Y")
plt.grid(True)
plt.show()
```

El problema planteado es un modelo de transporte, en el que la compañía “MG Auto” debe transportar automóviles desde sus tres plantas ubicadas en Los Ángeles, Detroit y Nueva Orleans hacia dos centros de distribución en Denver y Miami. El objetivo es minimizar los costos de transporte, teniendo en cuenta las capacidades de las plantas y las demandas en los centros de distribución, junto con los costos asociados a cada ruta.

Para resolver el problema, se utilizó programación lineal con un modelo de optimización para minimizar el costo total, para el cual los pasos son:

1. Definir la función objetivo: Minimizar el costo de transporte total entre las plantas y los centros.
2. Establecer restricciones:
    a. Capacidades de las plantas: Cada planta tiene un número máximo de automóviles que puede enviar.
    b. Demandas de los centros: Cada centro de distribución requiere una cantidad específica de automóviles que debe ser satisfecha.
3. Solución Inicial: Aplicar el método de esquina noroeste para obtener una distribución inicial.
4. Optimización: Utilizar el método MODI (Modified Distribution Method) para mejorar la solución inicial y encontrar el mínimo costo de transporte.

B. Resultados

Tras la optimización, la distribución óptima de transporte y el costo mínimo total son los siguientes:

Costo mínimo total: Zmin = $313200

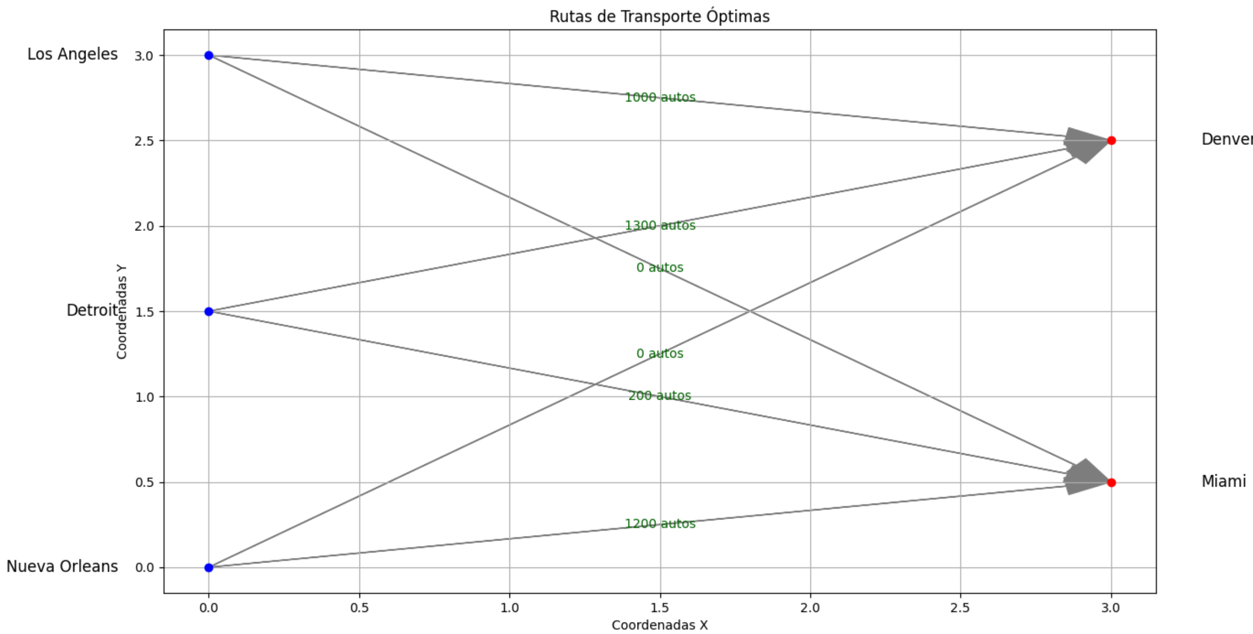
Distribución de transporte:

- Los Ángeles a Denver (x11): 1000 autos
- Los Ángeles a Miami (x12): 0 autos
- Detroit a Denver (x21): 1300 autos
- Detroit a Miami (x22): 200 autos
- Nueva Orleans a Denver (x31): 0 autos
- Nueva Orleans a Miami (x32): 1200 autos

C. Análisis de resultados

- Desde Los Ángeles: Se envían 1000 autos a Denver y ninguno a Miami. Esto sugiere que el costo de transporte desde Los Ángeles a Denver es lo suficientemente bajo para maximizar la capacidad de esta planta hacia ese destino.
- Desde Detroit: Esta planta envía 1300 autos a Denver y 200 autos a Miami. La distribución indica que Detroit cumple con una parte significativa de la demanda en ambos centros, siendo una ubicación estratégicamente económica para ambas rutas.
- Desde Nueva Orleans: Se envían 1000 autos a Miami y ninguno a Denver. Esto sugiere que el costo de transporte desde Nueva Orleans a Miami es lo suficientemente bajo para maximizar la capacidad de esta planta hacia ese destino.

El gráfico generado muestra las rutas y las cantidades de autos transportados entre las plantas y los centros de distribución. Las líneas representan las rutas utilizadas, mientras que las cantidades asignadas a cada ruta están basadas en la solución óptima obtenida.



D. Conclusiones

- La solución obtenida permite satisfacer toda la demanda de ambos centros de distribución, respetando las capacidades de cada planta.
- Las plantas están asignando sus producciones a los destinos de manera que minimizan el costo total de transporte. Los centros de distribución reciben exactamente la cantidad de autos requerida, con:
  - Denver recibiendo 2300 autos en total (de Los Ángeles y Detroit).
  - Miami recibiendo 1400 autos en total (de Detroit y Nueva Orleans).
- La elección de rutas en esta distribución óptima asegura que las plantas no transporten más de su capacidad y que el costo de transporte sea el mínimo posible de acuerdo a los costos por milla establecidos.

Problema 2.

Programe y resuelva la solución del problema de programación no lineal (5 Puntos)

1.- La función de beneficios de una empresa viene dada por la función:

$$B(x,y,z) = x\,y + 2\,z^2$$

donde x, y, z son las cantidades a producir de cada uno de los tres artículos que fabrica y vende.La empresa produce estos tres productos en un única sección en la que hay disponibles 120 horas semanales, empleando en la producción de una unidad del primer articulo 5 horas, en una del segundo 20 horas y en una del tercero 4 horas.

Se sabe además que por razones de demanda la empresa no puede producir menos de 5 unidades del primer articulo, ni más de 10 del segundo.

1. Determinar la producción a realizar.
2. Cuál debería ser la retribución de una hora extraordinaria?

A. **Código Python3**

```
from scipy.optimize import minimize
import numpy as np

# Definir la función objetivo con la función corregida
def beneficio(vars):
    x, y, z = vars
     return -(x * y + 2 * z**2)  # Minimizar la función negativa para
maximizar

# Definir las restricciones
constraints = [
     {'type': 'ineq', 'fun': lambda vars: 120 - (5 * vars[0] + 20 *
vars[1] + 4 * vars[2])},  # 5x + 20y + 4z <= 120
    {'type': 'ineq', 'fun': lambda vars: vars[0] - 5},   # x >= 5
    {'type': 'ineq', 'fun': lambda vars: 10 - vars[1]},  # y <= 10
    {'type': 'ineq', 'fun': lambda vars: vars[0]},       # x >= 0
    {'type': 'ineq', 'fun': lambda vars: vars[1]},       # y >= 0
    {'type': 'ineq', 'fun': lambda vars: vars[2]}        # z >= 0
```

```
]

# Valores iniciales para x, y, z
initial_guess = [5, 10, 0]

# Resolver el problema de optimización
result = minimize(beneficio, initial_guess, constraints=constraints)

# Obtener los valores óptimos y el beneficio máximo
optimal_values = result.x
max_benefit = -result.fun

print("Valores óptimos para la producción:")
print(f"x (Producto 1): {optimal_values[0]}")
print(f"y (Producto 2): {optimal_values[1]}")
print(f"z (Producto 3): {optimal_values[2]}")
print(f"Beneficio máximo: {max_benefit}")


# Resolver el problema original
result_original = minimize(beneficio, initial_guess,
constraints=constraints)
max_benefit_original = -result_original.fun

# Modificar la restricción de tiempo de 120 a 121 horas
constraints_extra_hour = [
     {'type': 'ineq', 'fun': lambda vars: 121 - (5 * vars[0] + 20 *
vars[1] + 4 * vars[2])},  # 5x + 20y + 4z <= 121
    {'type': 'ineq', 'fun': lambda vars: vars[0] - 5},   # x >= 5
    {'type': 'ineq', 'fun': lambda vars: 10 - vars[1]},  # y <= 10
    {'type': 'ineq', 'fun': lambda vars: vars[0]},       # x >= 0
    {'type': 'ineq', 'fun': lambda vars: vars[1]},       # y >= 0
    {'type': 'ineq', 'fun': lambda vars: vars[2]}        # z >= 0
]
```

```
# Resolver el problema con una hora extra
result_extra_hour = minimize(beneficio, initial_guess,
constraints=constraints_extra_hour)
max_benefit_extra_hour = -result_extra_hour.fun

# Calcular el valor de la retribución por una hora extra
extra_hour_value = max_benefit_extra_hour - max_benefit_original

print(f"Retribución de una hora extraordinaria: {extra_hour_value}")
```

B. Resultados

```
Valores óptimos para la producción:
x (Producto 1): 11.999999999999924
y (Producto 2): 2.9999999999996945
z(Producto 3): 2.6714741530043505e-16
Beneficio máximo: 35.999999999996106
Retribución de una hora extraordinaria: 0.6025000000033884
```

C. Análisis de resultados

a. Valores Óptimos para la Producción

Los valores óptimos encontrados para la producción son:
Producto 1 (x): aproximadamente 12 unidades.
Producto 2 (y): aproximadamente 3 unidades.
Producto 3 (z): prácticamente 0 unidades.

La solución indica que, para maximizar los beneficios, la empresa debería concentrarse en producir únicamente los Productos 1 y 2, mientras que el Producto 3 no aporta significativamente al beneficio óptimo en las condiciones dadas. Esto puede ser debido a la estructura de la función de beneficios y las restricciones de tiempo que hacen que la producción del Producto 3 no sea eficiente dentro del límite de 120 horas semanales.

b. Beneficio Máximo

El beneficio máximo alcanzado es aproximadamente 36 unidades monetarias.

Este beneficio es el valor más alto que la empresa puede obtener con la producción óptima encontrada y las restricciones de tiempo y demanda establecidas. Dado que se alcanzaron las restricciones sin violarlas, este valor representa la mejor combinación posible de los productos dentro de las condiciones especificadas.

c. Retribución de una Hora Extraordinaria

La retribución calculada para una hora adicional de trabajo es de aproximadamente 0.6025 unidades monetarias.

Cada hora adicional de trabajo disponible aumentaría el beneficio en 0.6025, ayudando a la empresa a evaluar el costo-beneficio de contratar horas extra.

Problema 3.

Programe y resuelva la solución del problema de pronósticos (5 Puntos)

Hospital Escazú proporciona servicios de laboratorio para pacientes de todos sus distritos, con un grupo de 10 médicos que atienden a familias con un nuevo programa de mantenimiento de la salud. Los administradores están interesados en pronosticar el número de análisis de sangre que se piden por semana. Una publicidad reciente acerca de los efectos dañinos del colesterol en el corazón ha ocasionado un incremento nacional en las peticiones de análisis de sangre.

Usar un promedio móvil simple de 3 meses, promedio ponderado de 0.6, 0.3 y 0.1 para los últimos 3 meses, y un modelo de suavización exponencial (α) de 0.4

El primer pronóstico para el modelo de suavización tiene un valor de 26.

A. Calcular: Error Medio Cuadrático (MSE), Error Absoluto Medio (MAE), Error Porcentual Medio (MAPE)
B. Elabore un gráfico con las proyecciones de cada método y las llegadas reales.
C. ¿Cuál es el mejor método, explique sus resultados?

Semana	Llegadas	Semana	Llegadas
1	28	9	61
2	27	10	39
3	44	11	55
4	37	12	54
5	35	13	52
6	53	14	60
7	38	15	60
8	57	16	75

A. Código en Python3

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

# Datos según la tabla
data = {
  "Semana": list(range(1, 17)),
  "Llegadas": [28, 27, 44, 37, 35, 53, 38, 57, 61, 39, 55, 54, 52, 60, 60, 75]
}

# Convertir a Data Frame
df = pd.DataFrame(data)

# Métodos de Pronóstico
# 1. Simple Moving Average (SMA) - 3 semanas
df["SMA"] = df["Llegadas"].rolling(window=3).mean()

# 2. Weighted Moving Average (WMA) - promedios 0.6, 0.3, 0.1 para las últimas 3
semanas
weights = np.array([0.6, 0.3, 0.1])
df["WMA"] = df["Llegadas"].rolling(window=3).apply(lambda x: np.dot(weights,
x) if len(x) == 3 else np.nan)

# 3. Exponential Smoothing (α = 0.4) - pronóstico inicial seteado en 26
alpha = 0.4
df["Exp_Smooth"] = 26.0  # Pronóstico inicial como float
for i in range(1, len(df)):
  df.loc[i, "Exp_Smooth"] = alpha * df.loc[i - 1, "Llegadas"] + (1 - alpha) * df.loc[i -
1, "Exp_Smooth"]
```

```
# Cálculo de Errores
# Calculating MSE, MAE, and MAPE para cada método
df["Error_SMA"] = df["Llegadas"] - df["SMA"]
df["Error_WMA"] = df["Llegadas"] - df["WMA"]
df["Error_Exp_Smooth"] = df["Llegadas"] - df["Exp_Smooth"]

# MSE
mse_sma = (df["Error_SMA"] ** 2).mean()
mse_wma = (df["Error_WMA"] ** 2).mean()
mse_exp_smooth = (df["Error_Exp_Smooth"] ** 2).mean()

# MAE
mae_sma = df["Error_SMA"].abs().mean()
mae_wma = df["Error_WMA"].abs().mean()
mae_exp_smooth = df["Error_Exp_Smooth"].abs().mean()

# MAPE (excluir ceros para evadir la división entre cero)
mape_sma = (df["Error_SMA"].abs() / df["Llegadas"]).mean() * 100
mape_wma = (df["Error_WMA"].abs() / df["Llegadas"]).mean() * 100
mape_exp_smooth = (df["Error_Exp_Smooth"].abs() / df["Llegadas"]).mean() *
100

# Gráfico de los resultados
plt.figure(figsize=(12, 6))
plt.plot(df["Semana"], df["Llegadas"], marker='o', label="Actual Llegadas")
plt.plot(df["Semana"], df["SMA"], marker='o', linestyle='--', label="3-Week SMA")
plt.plot(df["Semana"], df["WMA"], marker='o', linestyle='--', label="Weighted
3-Week MA")
```

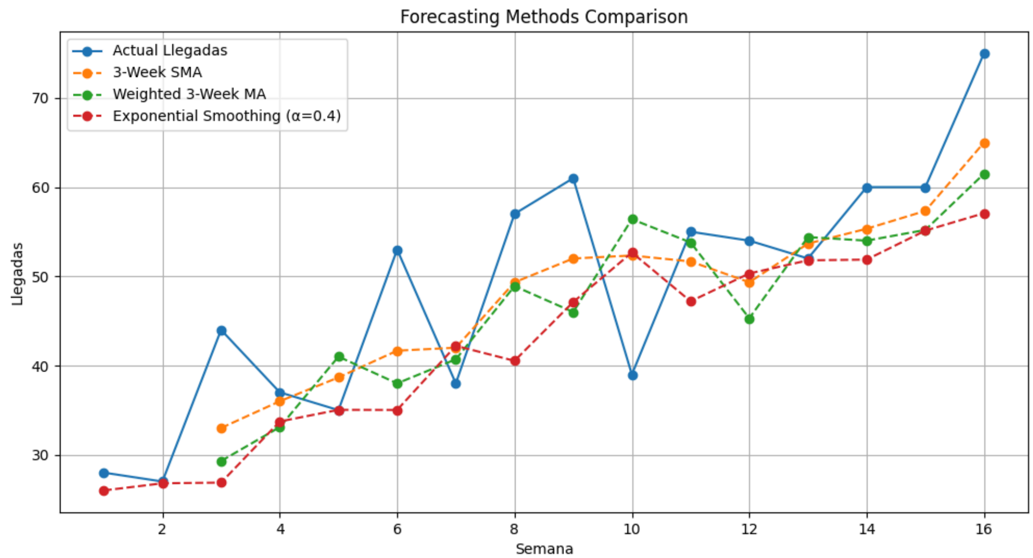
```
plt.plot(df["Semana"], df["Exp_Smooth"], marker='o', linestyle='--',
label="Exponential Smoothing (α=0.4)")

plt.xlabel("Semana")
plt.ylabel("Llegadas")
plt.title("Forecasting Methods Comparison")
plt.legend()
plt.grid(True)
plt.show()

# Mostrar errores
errors = pd.DataFrame({
  "Method": ["SMA", "WMA", "Exp_Smooth"],
  "MSE": [mse_sma, mse_wma, mse_exp_smooth],
  "MAE": [mae_sma, mae_wma, mae_exp_smooth],
  "MAPE": [mape_sma, mape_wma, mape_exp_smooth]
})

# Mostrar el DataFrame de errores en consola
print(errors)
```

B. Gráfico obtenido



C. Resultados

	Method	MSE	MAE	MAPE
0	SMA	54.428571	6.285714	12.567468
1	WMA	101.224286	8.528571	17.058859
2	Exp_Smooth	111.574439	8.215817	15.900766

D. Análisis de resultados

El mejor método de pronóstico es el **Promedio Móvil Simple (SMA)**, ya que presenta los valores más bajos de error en todas las métricas (MSE, MAE y MAPE), lo que indica una mayor precisión en comparación con el Promedio Móvil Ponderado y la Suavización Exponencial. Esto sugiere que el SMA es más adecuado para este conjunto de datos, posiblemente porque los datos no muestran una tendencia o estacionalidad fuerte que beneficie de ponderaciones adicionales o suavización, haciendo que el promedio simple capte mejor las variaciones en las llegadas semanales.

Anexos

Link al Google Drive con los videos de los 3 problemas:

https://drive.google.com/drive/folders/1ZIWqKAMm6R0jF6d1XdANZMquoEBJDx06?usp=sharing