|  |  |
| --- | --- |
| D:\W Varios\Logos\Logo 2008\Logo08.jpg | Graficos con 3 ejes o masGraficos con Python en Excel, creando graficos con tres o mas ejes.Jose Ignacio González Gómez Departamento de Economía, Contabilidad y Finanzas - Universidad de La Laguna[www.jggomez.eu](http://www.jggomez.eu) en el TEIDE C B **V. 2.5**Ejercicio basado : [Grafico de 3 ejes o más con Python en Excel](https://www.youtube.com/watch?v=BGfSdq4t-ZQ)  |

Contenido

[Presentacion del caso 1. Dólar Peso – GLD (fondo de inversión en oro) – Acción de Nvidia – Acción de Ecopetrol 2](#_Toc211013026)

[Introducción al problema de dispersión de los datos. 2](#_Toc211013027)

[Propuesta de solución con Python en Excel 2](#_Toc211013028)

[Script Python 3](#_Toc211013029)

[Resultado 5](#_Toc211013030)

[Presentacion del caso 2. Temperatura, Presión, Humedad y Velocidad 5](#_Toc211013031)

[Datos a representar 5](#_Toc211013032)

[Propuesta de solución con Python en Excel 5](#_Toc211013033)

[Cargar las librerías necesarias (script) 5](#_Toc211013034)

[Definimos el dataframe (script) 6](#_Toc211013035)

[Creamos el gráfico (script) 6](#_Toc211013036)

[Resultado 7](#_Toc211013037)

# Presentacion del caso 1. Dólar Peso – GLD (fondo de inversión en oro) – Acción de Nvidia – Acción de Ecopetrol

## Introducción al problema de dispersión de los datos.

Este caso aborda la solución a un problema común en Excel: la **dificultad de crear gráficos con tres o más ejes** de forma nativa y para ello aplicaremos **Python en Excel**.

En este caso propuesto disponemos de los datos de cotización de cuatro valores que están muy dispersas en sus unidades de medida:

1. **Dólar Peso:** Sus valores oscilan entre los 3,000 y 4,000 pesos colombiano
2. **GLD (Fondo de Inversión en Oro):** Sus valores se mueven entre 140 y 243.
3. **Acción de Nvidia:** Experimentó un rendimiento positivo, pasando de 5 en 2020 a 121 en 2024.
4. **Acción de Ecopetrol:** Tuvo un rendimiento negativo, cayendo de 18.5 en 2020 a 8.93 en 2024 (casi a la mitad). Empresa Colombiana de Petróleo S.A

Cuando estas cuatro series, que tienen indicadores muy dispersos, se colocan en un solo gráfico con la **configuración nativa de Excel** (que permite un máximo de dos ejes), los resultados son difíciles de comparar e interpretar.



Ilustración

Por ejemplo, al colocar el Dólar Peso en el eje izquierdo y el resto de las series en el eje derecho, la acción de Ecopetrol (la línea amarilla) se ve literalmente **como una constante**, ocultando el hecho de que su precio se redujo "casi a la mitad". Por esta razón, usar gráficos de dos ejes con series muy dispersas no es una buena idea.

Este problema esta relacionado concretamente con la dispersión de los datos.

## Propuesta de solución con Python en Excel

La solución se presenta mediante el uso de Python dentro de Excel. El resultado es un nuevo gráfico con **cuatro ejes** en la visualización.

* **Mejora de la Visualización:** Al tener cada serie en su propio eje, el comportamiento de la acción de Ecopetrol se visualiza correctamente; se ve "mucho menos lineal" y se observa que la acción ha **caído contundentemente** entre 2020 y 2024.
* **Oportunidad de Comparación:** Este tipo de gráficos ofrece una "gran oportunidad" para comparar series que están muy dispersas sin ningún problema.



Ilustración

## Script Python

#Crear dataframe: ajústado a tus celdas

df = xl("A3:E60", headers=True)

# Excluir la columna 'Fecha'

series\_names = df.columns[1:]

# Obtener dinámicamente los nombres de las series

fig, ax1 = plt.subplots(figsize=(12, 6)) # Ancho 12, alto a 6

ax1.set\_xlabel('Fecha') # Primer gráfico en el primer eje Y

ax1.set\_ylabel(series\_names[0], color='tab:red')

ax1.plot(df['Fecha'], df[series\_names[0]], color='tab:red', label=series\_names[0])

ax1.tick\_params(axis='y', labelcolor='tab:red')

# Número de etiquetas que quieres en el eje X y formato

num\_ticks = 6

# Índices distribuidos uniformemente

tick\_indices = np.linspace(0, len(df['Fecha']) - 1, num\_ticks, dtype=int)

# Etiquetas de fecha correspondientes

tick\_labels = df['Fecha'].iloc[tick\_indices]

# Aplicar al eje X

ax1.set\_xticks(df['Fecha'].iloc[tick\_indices])

# Segundo eje Y

ax2 = ax1.twinx()

ax2.set\_ylabel(series\_names[1], color='tab:blue')

ax2.plot(df['Fecha'], df[series\_names[1]], color='tab:blue', label=series\_names[1])

ax2.tick\_params(axis='y', labelcolor='tab:blue')

# Tercer eje Y (desplazado hacia la derecha)

ax3 = ax1.twinx()

ax3.spines['right'].set\_position(('outward', 60)) # Mover el tercer eje hacia afuera

ax3.set\_ylabel(series\_names[2], color='tab:green')

ax3.plot(df['Fecha'], df[series\_names[2]], color='tab:green', label=series\_names[2])

ax3.tick\_params(axis='y', labelcolor='tab:green')

# Cuarto eje Y (desplazado aún más hacia la derecha)

ax4 = ax1.twinx()

ax4.spines['right'].set\_position(('outward', 120)) # Mover el cuarto eje más hacia afuera

ax4.set\_ylabel(series\_names[3], color='tab:pink')

ax4.plot(df['Fecha'], df[series\_names[3]], color='tab:pink', label=series\_names[3])

ax4.tick\_params(axis='y', labelcolor='tab:pink')

# Ajustar el layout para evitar solapamientos

fig.tight\_layout()

# Crear título dinámico basado en las series

plt.title(f'Gráfico de {", ".join(series\_names)} con Ejes Múltiples')

# Crear leyenda y situarla en la parte inferior

# Guardar las líneas para la leyenda

lines = []

labels = []

# Primer eje

line1, = ax1.plot(df['Fecha'], df[series\_names[0]], color='tab:red', label=series\_names[0])

lines.append(line1)

labels.append(series\_names[0])

# Segundo eje

line2, = ax2.plot(df['Fecha'], df[series\_names[1]], color='tab:blue', label=series\_names[1])

lines.append(line2)

labels.append(series\_names[1])

# Tercer eje

line3, = ax3.plot(df['Fecha'], df[series\_names[2]], color='tab:green', label=series\_names[2])

lines.append(line3)

labels.append(series\_names[2])

# Cuarto eje

line4, = ax4.plot(df['Fecha'], df[series\_names[3]], color='tab:pink', label=series\_names[3])

lines.append(line4)

labels.append(series\_names[3])

# Añadir leyenda en la parte inferior

fig.legend(lines, labels, loc='lower center', ncol=4, bbox\_to\_anchor=(0.5, -0.05))

# Mostrar el gráfico

plt.show()

## Resultado



Ilustración

# Presentacion del caso 2. Temperatura, Presión, Humedad y Velocidad

## Datos a representar

En este caso vamos a representar cuatro valores igualmente de magnitudes diferentes, tal y como vemos en la siguiente ilustración, Temperatura, Presión, Humedad y Velocidad.



Ilustración

## Propuesta de solución con Python en Excel

En este caso valor a proponer la solución basada en los tres pasos mostrados en la ilustración anterior.

### Cargar las librerías necesarias (script)

Comenzamos garantizando que se cargan las librerías necesarias

# cargamos las librerías

import pandas as pd

import matplotlib.pyplot as plt

### Definimos el dataframe (script)

#Asignado el Dataframe

df=xl("A2:E12", headers=True)

### Creamos el gráfico (script)

# Crear figura y eje principal

fig, ax1 = plt.subplots(figsize=(10, 6))

# Primer eje: Temperatura

ax1.plot(df['Tiempo'], df['Temperatura'], 'r-', label='Temperatura')

ax1.set\_ylabel('Temperatura (°C)', color='r')

ax1.tick\_params(axis='y', labelcolor='r')

# Segundo eje: Presión

ax2 = ax1.twinx()

ax2.plot(df['Tiempo'], df['Presion'], 'b--', label='Presión')

ax2.set\_ylabel('Presión (kPa)', color='b')

ax2.tick\_params(axis='y', labelcolor='b')

# Tercer eje: Humedad

ax3 = ax1.twinx()

ax3.spines['right'].set\_position(('outward', 60))

ax3.plot(df['Tiempo'], df['Humedad'], 'g-.', label='Humedad')

ax3.set\_ylabel('Humedad (%)', color='g')

ax3.tick\_params(axis='y', labelcolor='g')

# Cuarto eje: Velocidad

ax4 = ax1.twinx()

ax4.spines['right'].set\_position(('outward', 120))

ax4.plot(df['Tiempo'], df['Velocidad'], 'm:', label='Velocidad')

ax4.set\_ylabel('Velocidad (m/s)', color='m')

ax4.tick\_params(axis='y', labelcolor='m')

# Combinar leyendas

lines = ax1.get\_lines() + ax2.get\_lines() + ax3.get\_lines() + ax4.get\_lines()

labels = [line.get\_label() for line in lines]

# Título del gráfico

plt.title('Gráfico de líneas con cuatro escalas diferentes')

# Etiqueta del eje X

ax1.set\_xlabel('Tiempo (segundos)', labelpad=5)

# Ajustar márgenes inferiores para que se vea la etiqueta del eje X y la leyenda

plt.subplots\_adjust(bottom=0.10)

# Colocar la leyenda debajo del eje X

fig.legend(lines, labels, loc='lower center', bbox\_to\_anchor=(0.5, -0.05), ncol=4)

# Mostrar gráfico

plt.show()

## Resultado



Ilustración