**INDEX**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Sr. No.** | **Practical List** | **Page No.** | **Date** | **Sign** |
| 1. | Fitting and plotting of modified exponential curve | 1 – 3 |  |  |
| 2. | Fitting and plotting of Gompertz curve | 4 – 5 |  |  |
| 3. | Fitting and plotting of logistic curve. | 6 – 7 |  |  |
| 4. | Fitting of trend by Moving Average Method. | 8 – 9 |  |  |
| 5. | Measurement of Seasonal indices Ratio-to-Trend method. | 10 – 12 |  |  |
| 6. | Measurement of Seasonal indices Ratio-to-Moving Average method. | 13 – 16 |  |  |
| 7. | Measurement of seasonal indices Link Relative method. | 17 |  |  |
| 8. | Calculation of variance of random component by variate difference  method. | 18 – 19 |  |  |
| 9. | Forecasting by exponential smoothing. | 20 |  |  |
| 10. | Forecasting by short term forecasting methods. | 21 – 22 |  |  |

**Practical 1**

**Aim:** Fitting and plotting of modified exponential curve.

**Theory:**

A modified exponential curve is a mathematical model used to describe data that follows an exponential growth or decay pattern but includes an additional constant term to account for shifts or offsets in the data. This type of model is particularly useful in various fields such as finance, biology, and physics, where exponential relationships are observed but data does not start at zero.

*The General Form*

The general form of the modified exponential function can be expressed as:

where:

* is the dependent variable (e.g., gold price).
* is the independent variable (e.g., time).
* is a scaling factor that adjusts the amplitude of the curve.
* is the growth rate (if ) or decay rate (if ).
* is a constant that shifts the entire curve vertically.

*Applications*

The modified exponential curve is useful in various scenarios:

* Finance: Modeling the growth of investments or prices of commodities, like gold prices, over time.
* Biology: Describing population growth with a carrying capacity or decay in the presence of a constant offset.
* Physics: Modeling radioactive decay processes or other phenomena with a constant background level.

*Fitting the Curve*

To fit a modified exponential curve to a dataset, numerical methods such as nonlinear least squares can be used. The goal is to find the parameters , , and that minimize the difference between the observed data points and the values predicted by the model.

The modified exponential curve is a powerful tool for modeling data that exhibits exponential growth or decay with an offset. By incorporating a constant term, this model provides a more accurate fit for many real-world scenarios, making it invaluable in fields like finance, biology, and physics. Understanding the components and interpretation of the parameters helps in applying this model effectively to analyze and predict trends in various datasets.

**Code:**

import pandas as pd

import numpy as np

from scipy.optimize import curve\_fit

import matplotlib.pyplot as plt

from io import StringIO

# Read the data into a DataFrame

df = pd.read\_csv("Gold.csv", parse\_dates=['Date'])

# Convert the Date column to datetime format and create a numerical representation

df['Days'] = (df['Date'] - df['Date'].min()).dt.days

# Extract the days and values for fitting

x = df['Days'].values

y = df['Value'].values

# Define the modified exponential function

def modified\_exponential(x, a, b, c):

return a \* np.exp(b \* x) + c

# Fit the curve

params, \_ = curve\_fit(modified\_exponential, x, y)

# Extract the parameters

a, b, c = params

print(f"Fitted parameters: a = {a}, b = {b}, c = {c}")

# Generate x values for the fitted curve

x\_fit = np.linspace(x.min(), x.max(), 1000)

y\_fit = modified\_exponential(x\_fit, a, b, c)

# Plot the original data and the fitted curve

plo

plt.plot(np.array(df['Date'].min() + pd.to\_timedelta(x\_fit, unit='D')), y\_fit, color='red', label='Fitted Curve')

plt.xlabel('Date')

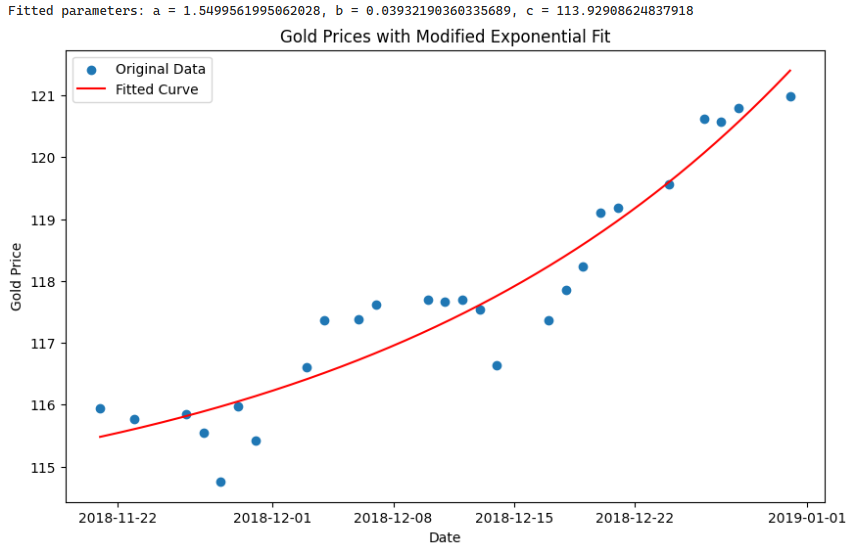
plt.ylabel('Gold Price')

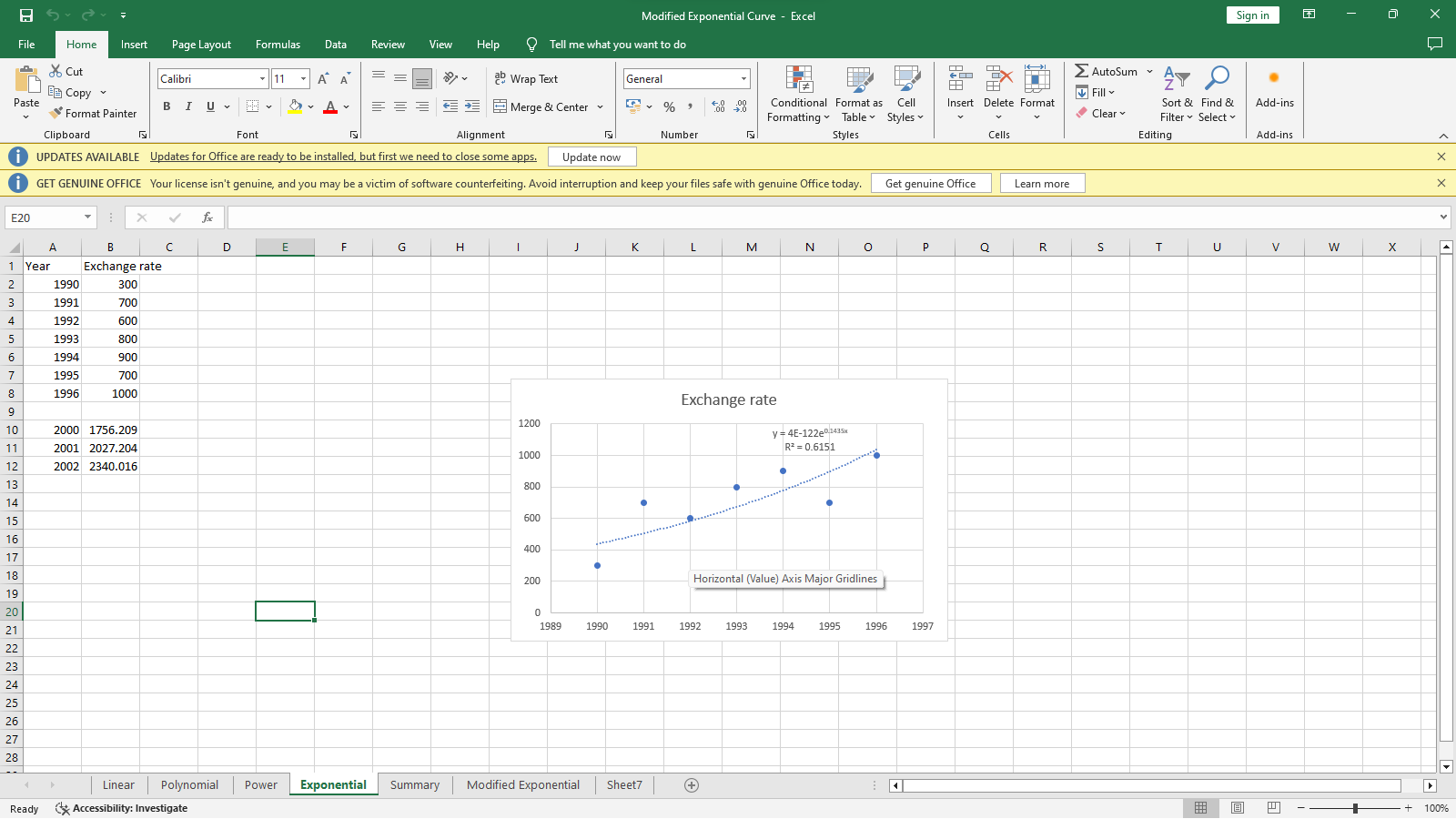
plt.title('Gold Prices with Modified Exponential Fit')

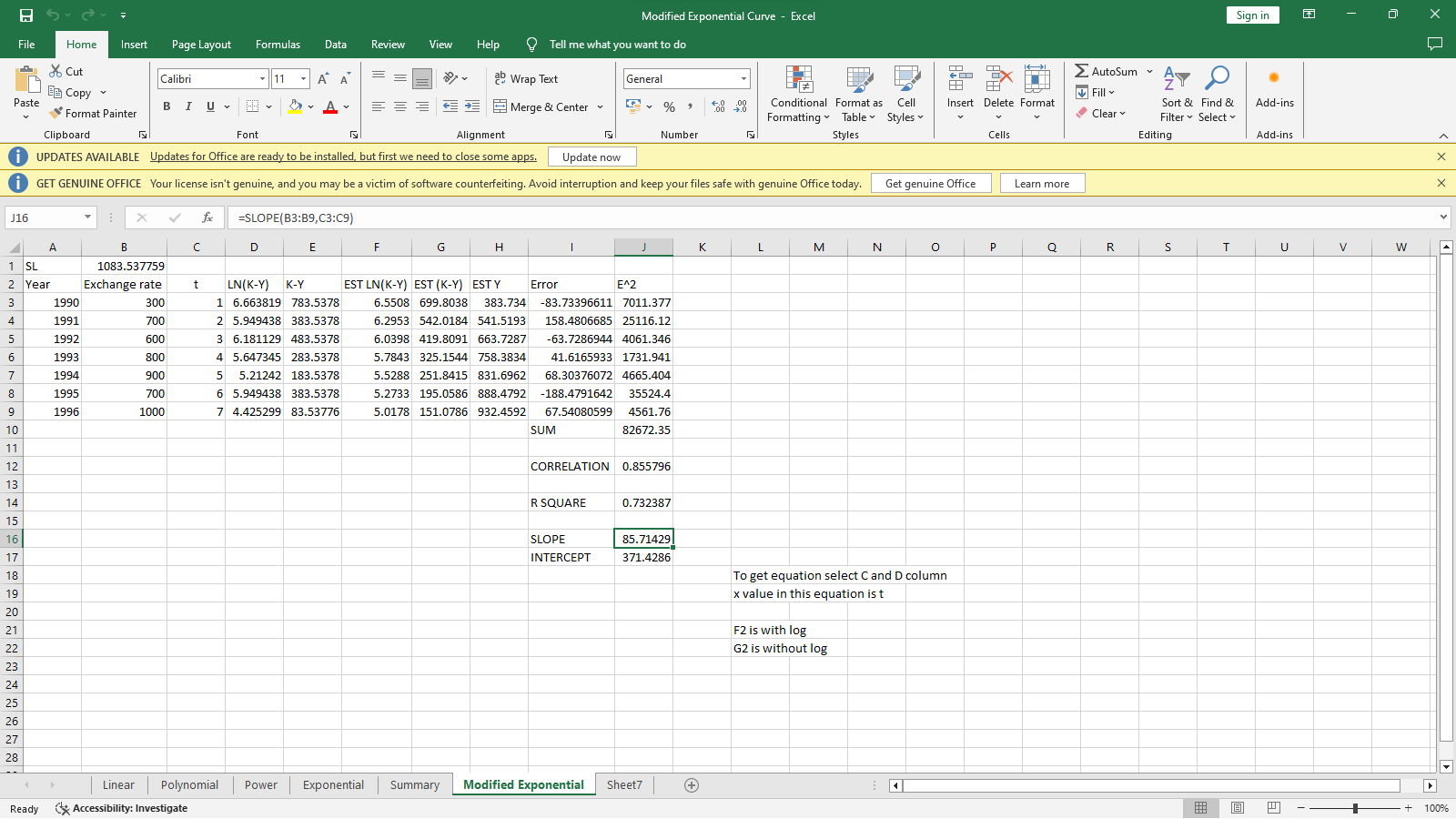
plt.legend()

plt.show()

**Output:**







**Practical 2**

**Aim:** Fitting and plotting of Gompertz curve.

**Theory:**

The Gompertz curve is a type of sigmoid function that is often used to describe growth processes, such as the growth of populations, the spread of diseases, and the progression of certain types of cancers. It is named after Benjamin Gompertz, who introduced it in 1825 as a mathematical model for human mortality. The Gompertz function is characterized by its asymmetrical S-shape, reflecting rapid growth at the beginning that slows down as it approaches an upper limit.

*The General Form:* The general form of the modified exponential function can be expressed as:

where:

* is the dependent variable (e.g., population size, price level).
* is the independent variable (e.g., time).
* is the upper asymptote or the maximum value that 𝑦 can reach.
* and are parameters that control the displacement along the x-axis and the growth rate, respectively.

*Fitting the Gompertz Curve*

To fit a Gompertz curve to empirical data, nonlinear regression techniques are typically used. This involves finding the parameters a, b, and c that minimize the difference between the observed data points and the values predicted by the Gompertz function. The curve\_fit function from the scipy.optimize module in Python is commonly used for this purpose.

The Gompertz curve is a powerful and versatile tool for modeling a wide range of growth processes. Its asymmetrical S-shape and the flexibility of its parameters allow it to accurately represent growth phenomena in biology, demography, economics, and beyond. Understanding the underlying theory and properties of the Gompertz curve helps in effectively applying this model to analyze and predict trends in various datasets.

**Code:**

import pandas as pd

import numpy as np

from scipy.optimize import curve\_fit

import matplotlib.pyplot as plt

# Read the data into a DataFrame

df = pd.read\_csv("Gold.csv", parse\_dates=['Date'])

# Convert the Date column to datetime format and create a numerical representation

df['Days'] = (df['Date'] - df['Date'].min()).dt.days

# Extract the days and values for fitting

x = df['Days'].values

y = df['Value'].values

# Scale the data

x\_scaled = x / max(x)

y\_scaled = y / max(y)

# Define the Gompertz function

def gompertz(x, a, b, c):

return a \* np.exp(-b \* np.exp(-c \* x))

# Fit the curve

params, \_ = curve\_fit(gompertz, x\_scaled, y\_scaled, p0=[1, 1, 1], maxfev=100000)

# Extract the parameters

a, b, c = params

print(f"Fitted parameters: a = {a}, b = {b}, c = {c}")

# Generate x values for the fitted curve

x\_fit = np.linspace(x.min(), x.max(), 1000)

x\_fit\_scaled = x\_fit / max(x)

y\_fit\_scaled = gompertz(x\_fit\_scaled, a, b, c)

y\_fit = y\_fit\_scaled \* max(y)

# Plot the original data and the fitted curve

plt.figure(figsize=(10, 6))

plt.scatter(df['Date'], y, label='Original Data')

plt.plot(np.array(df['Date'].min() + pd.to\_timedelta(x\_fit, unit='D')), y\_fit, color='red', label='Fitted Gompertz Curve')

plt.xlabel('Date')

plt.ylabel('Gold Price')

plt.title('Gold Prices with Gompertz Curve Fit')

plt.legend()

plt.show()

***Output:***

A graph with a red line

Description automatically generated

**Practical 3**

**Aim**: Fitting and plotting of logistic curve.

**Theory:**

The logistic curve, or logistic function, is a sigmoidal mathematical model that describes growth processes that initially accelerate and then decelerate as they approach an upper limit. It is widely used in various fields, including biology, economics, epidemiology, and more, to model phenomena where growth saturates over time.

*Mathematical Formulation:* The logistic curve is defined by the following equation:

where:

* is the dependent variable (e.g., population size, price level).
* is the independent variable (e.g., time).
* is the curve's maximum value, also known as the carrying capacity or saturation level.
* is the growth rate parameter that determines the steepness of the curve.
* is the x-value of the sigmoid's midpoint, representing the point where the curve reaches half of its maximum value .

*Fitting a Logistic Curve*

To fit a logistic curve to empirical data, nonlinear regression techniques are typically employed. The curve\_fit function from the scipy.optimize module in Python is commonly used for this purpose. It iteratively adjusts the parameters L, k, and to minimize the difference between the observed data points and the values predicted by the logistic function.

*Plotting the Logistic Curve*

Once the logistic curve is fitted, it can be plotted along with the original data to visualize how well the model fits the empirical observations. This graphical representation helps in understanding the growth dynamics and making predictions about future trends based on the fitted model.

**Code:**

import pandas as pd

import numpy as np

from scipy.optimize import curve\_fit

import matplotlib.pyplot as plt

# Read the data into a DataFrame

df = pd.read\_csv("Gold.csv", parse\_dates=['Date'])

# Convert the Date column to datetime format and create a numerical representation

df['Days'] = (df['Date'] - df['Date'].min()).dt.days

# Extract the days and values for fitting

x = df['Days'].values

y = df['Value'].values

# Define the logistic function

def logistic(x, L, k, x\_0):

return L / (1 + np.exp(-k \* (x - x\_0)))

# Fit the curve

params, \_ = curve\_fit(logistic, x, y, p0=[max(y), 1, np.median(x)], maxfev=10000)

# Extract the parameters

L, k, x\_0 = params

print(f"Fitted parameters: L = {L}, k = {k}, x\_0 = {x\_0}")

# Generate x values for the fitted curve

x\_fit = np.linspace(x.min(), x.max(), 1000)

y\_fit = logistic(x\_fit, L, k, x\_0)

# Plot the original data and the fitted curve

plt.figure(figsize=(10, 6))

plt.scatter(df['Date'], y, label='Original Data')

plt.plot(np.array(df['Date'].min() + pd.to\_timedelta(x\_fit, unit='D')), y\_fit, color='red', label='Fitted Logistic Curve')

plt.xlabel('Date')

plt.ylabel('Gold Price')

plt.title('Gold Prices with Logistic Curve Fit')

plt.legend()

plt.show()

**Output:**

**A graph with a red line

Description automatically generated**

**Practical 4**

**Aim**: Fitting of trend by Moving Average Method

**Theory:**

The Moving Average Method is a statistical technique used to analyze time series data by calculating averages of subsets of data points over a specified period. This method is particularly useful for smoothing out short-term fluctuations to identify underlying trends in the data. Here’s a detailed explanation of the theory behind fitting a trend using the Moving Average Method:

*Purpose of Moving Average Method*

The primary goal of the Moving Average Method is to reduce the noise or random fluctuations in time series data, thereby making it easier to identify the underlying trend or pattern. By averaging out short-term variations, the method highlights longer-term trends that may be obscured by noise.

*Mathematical Formulation*

Given a time series where t represents time, the moving average at time with a window size n is calculated as:

​

where:

* are the observed values at time 𝑖
* is the window size, representing the number of periods over which to average.

***Advantages:***

1. *Smoothing Out Fluctuations:* One of the primary advantages of moving averages is their ability to smooth out short-term fluctuations and noise in the data. This is particularly useful in financial markets where prices can be volatile on a daily basis. By averaging out these fluctuations, moving averages provide a clearer picture of the underlying trend over time.
2. *Highlighting Trends:* Moving averages help to identify and visualize trends within the data. By averaging data points over a specified window, they emphasize long-term patterns and cycles that may not be immediately apparent from raw data. This is crucial for understanding the overall direction and behavior of the time series.
3. *Simplifying Data Interpretation:* They simplify complex time series data into a smoother curve, making it easier to interpret and analyze trends visually. This simplification enhances the ability to communicate insights to stakeholders and decision-makers who may not be familiar with the technical details of the data.
4. *Forecasting and Prediction:* Moving averages can be used to forecast future values based on historical trends. Once the underlying trend is identified through moving averages, extrapolating future values becomes more reliable, making them valuable tools for predictive modeling and forecasting.

**Code:**

import pandas as pd

import numpy as np

import matplotlib.pyplot as plt

# Sample data (replace with your actual dataset)

data = {

'Date': pd.date\_range(start='2023-01-01', periods=100),

'Value': np.random.rand(100) \* 100 # Random values for demonstration

}

df = pd.DataFrame(data)

def moving\_average(data, window\_size):

ma = data['Value'].rolling(window=window\_size, min\_periods=1).mean()

return pd.DataFrame({'Date': data['Date'], 'Moving Average': ma})

window\_size = 7 # Example: 7-day moving average

# Calculate moving average

df\_ma = moving\_average(df, window\_size)

# Plotting

plt.figure(figsize=(12, 6))

plt.plot(df, label='Original Data', marker='o')

plt.plot(df\_ma, color='red', label=f'{window\_size}-day Moving Average')

plt.xlabel('Date')

plt.ylabel('Value')

plt.title('Trend Fitting using Moving Average Method')

plt.legend()

plt.grid(True)

plt.show()

**Output:**

**A graph showing a number of different types of method

Description automatically generated with medium confidence**

**Practical 5**

**Aim:** Measurement of Seasonal indices Ratio-to-Trend method.

**A white rectangular object with black text

Description automatically generated**

**A close-up of a number

Description automatically generated**

**A table with numbers and numbers

Description automatically generated**

**A screenshot of a computer

Description automatically generated**

A screenshot of a computer

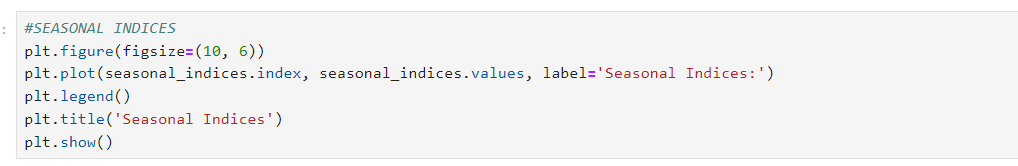
Description automatically generated

A screenshot of a computer

Description automatically generated

A graph with blue lines and orange lines

Description automatically generated



A graph showing the number of blue lines

Description automatically generated with medium confidence

A white background with black text

Description automatically generated

A graph with a line

Description automatically generated

**Practical 6**

**Aim:**

**A computer code with text

Description automatically generated**

**A close-up of a computer code

Description automatically generated**

**A computer code with text

Description automatically generated**

**A screenshot of a computer

Description automatically generated**

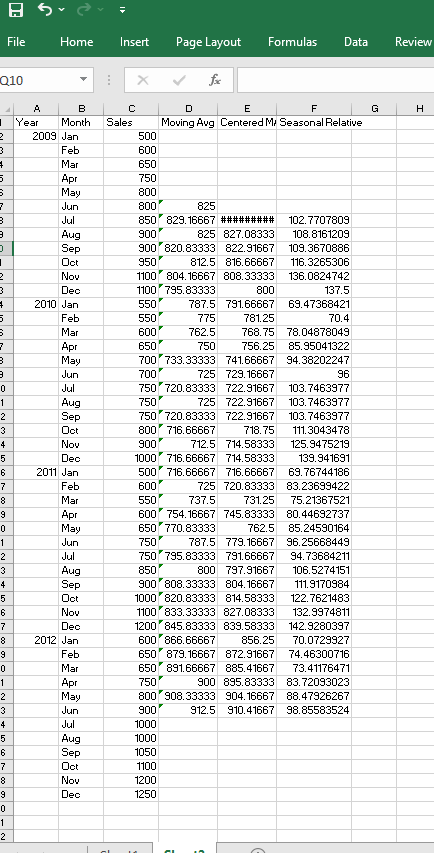
**Output:**

**A graph with a line

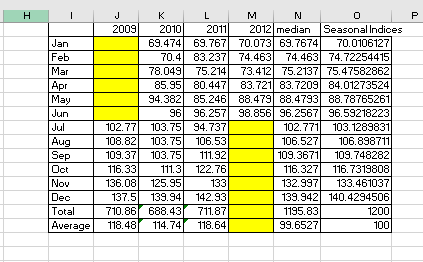
Description automatically generated**

**A graph with blue lines and orange lines

Description automatically generated**

****

**­­**

****

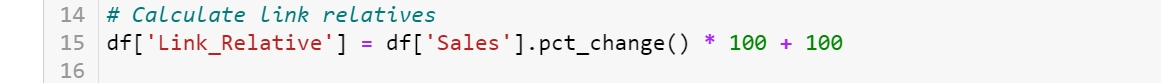
**Practical 7**

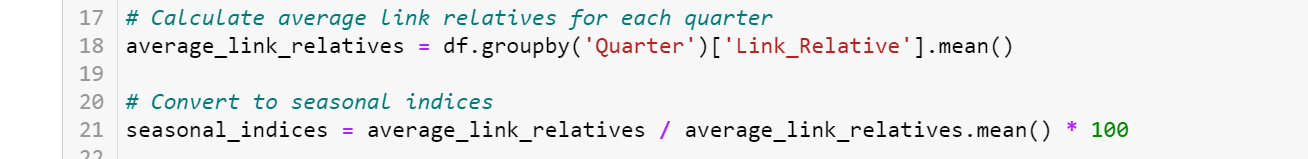
**Aim:** Measurement of seasonal indices Link Relative method.

**Code:**

**A number of numbers and symbols

Description automatically generated with medium confidence**

****

****

**A close-up of a computer screen

Description automatically generated**

**Output:**

**A graph with a line pointing to the center

Description automatically generated with medium confidence**

**Practical 8**

**Aim:** Calculation of variance of random component by variate difference method.

**Code:**

**A screenshot of a computer code

Description automatically generated**

**A graph with blue lines

Description automatically generated**

**A computer code with text

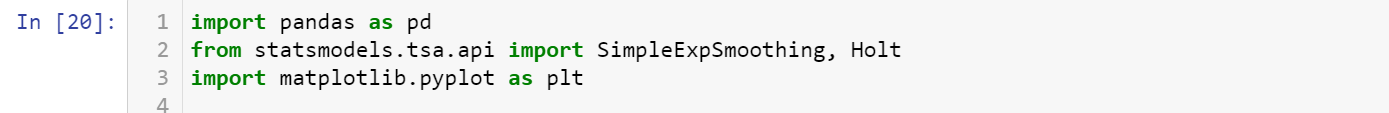
Description automatically generated**

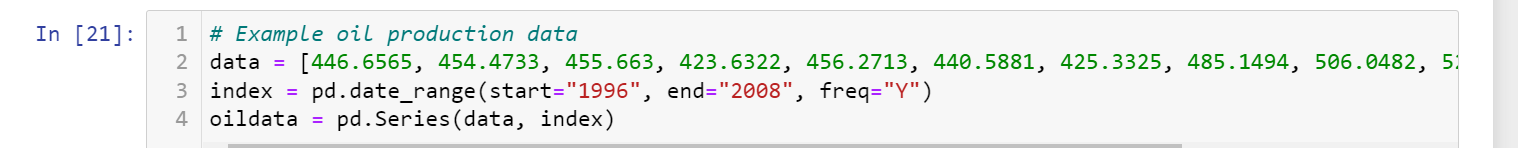
**A graph showing different differences

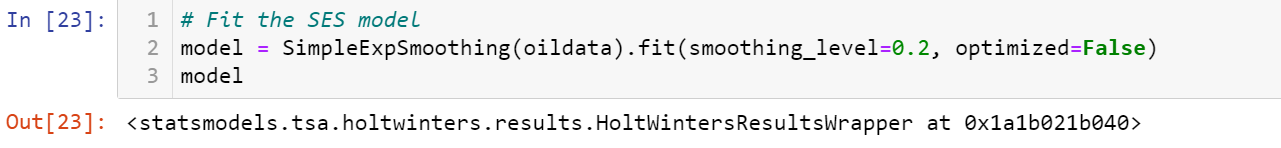
Description automatically generated**

**Practical 9**

**Aim:** Forecasting by exponential smoothing.

****

****

****

**A screenshot of a computer code

Description automatically generated**

**A screenshot of a computer program

Description automatically generated**

**A graph with numbers and lines

Description automatically generated**

**Practical 10**

**Aim:** Forecasting by short term forecasting methods.

**A screenshot of a computer program

Description automatically generated**

**A graph with lines and numbers

Description automatically generated**

**A computer screen shot of a computer program

Description automatically generated with medium confidence**