ISA 444: Business Forecasting

05: Time Series Summaries

Fadel M. Megahed, PhD

Endres Associate Professor
Farmer School of Business
Miami University

@FadelMegahed
fmegahed
fmegahed@miamioh.edu
Automated Scheduler for Office Hours

Fall 2023

Quick Refresher from Last Class

✅ - Examine the goals of utilizing line charts in time-series analysis (i.e., detect trends, seasonality, and cycles).

✅ Develop a deeper understanding of the grammar of graphics, which we used to create time series plots in R and Python.

✅ Use numerical summaries to describe a time series.

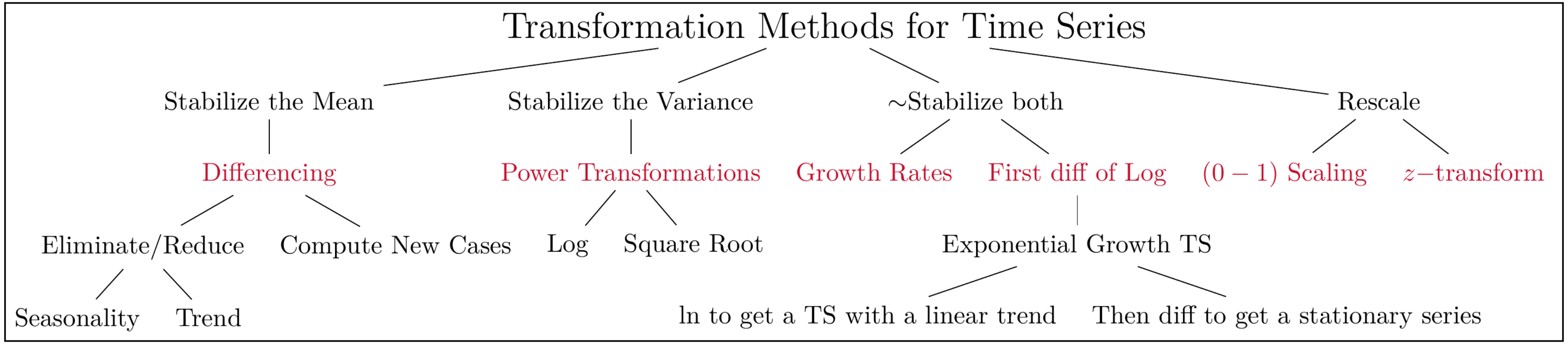
✅ Explain what do we mean by correlation.

Learning Objectives for Today's Class

- Apply transformations to a time series in both R and Python.

Transformations

Guidelines for Transforming Time Series Data



A classification of common transformation approaches for time series data

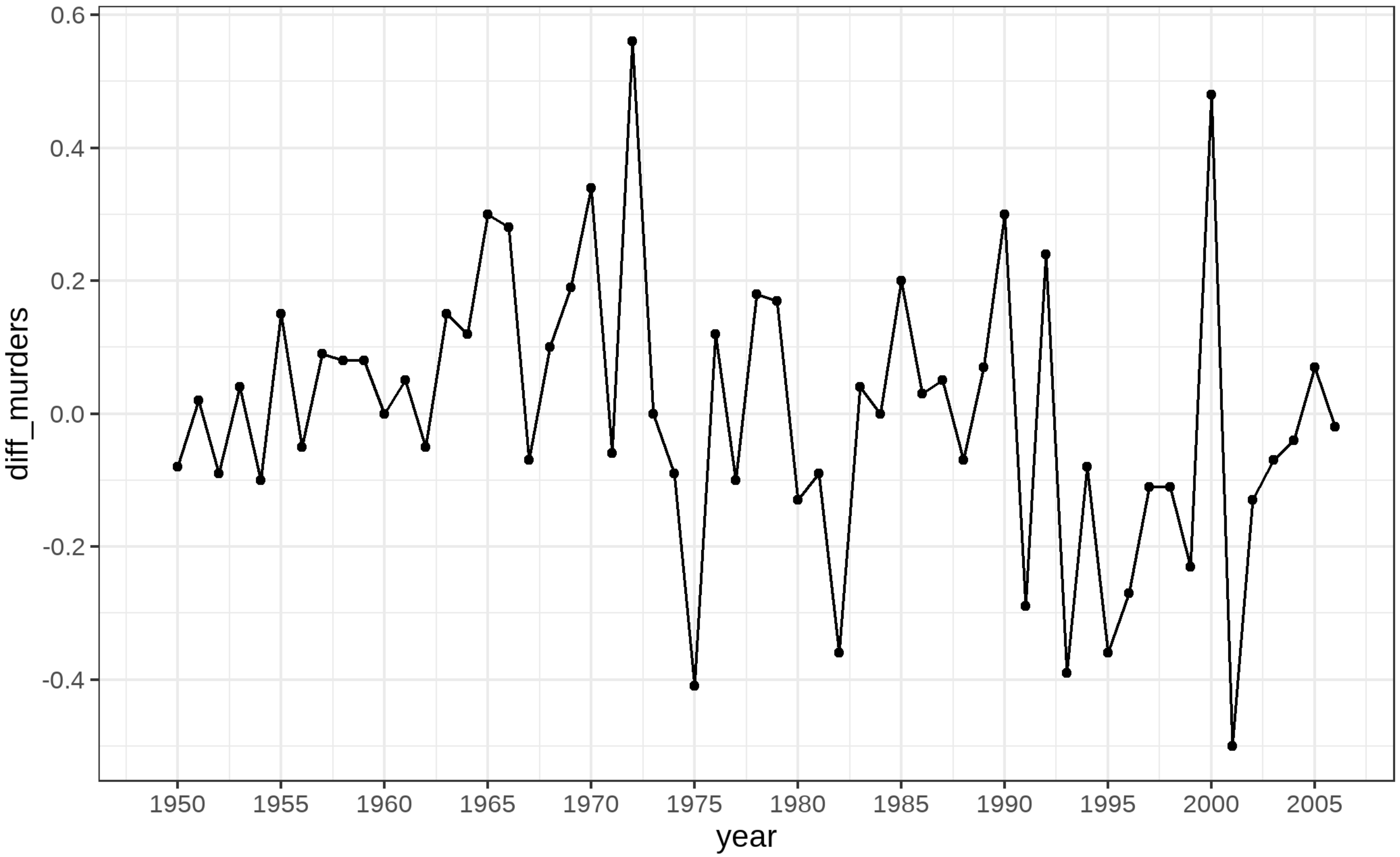
My (incomplete) attempt to provide you with a taxonomy for time series data transformations.

Stablize the Mean: Differencing

The plot below shows the number of murdered women per 100,000 people in the U.S. From the plot, we can see that the ts is not stationary.

Stablize the Mean: Differencing

The plot below shows the **first nonseasonal difference**. From the plot, we can see that differencing has reduced the nonstationary nature of the time-series.



Computing the First Nonseasonal Difference

The change in the time series from one period to the next is known as the first nonseasonal difference. It can be computed as follows:

$$DY_t = Y_t - Y_{t-1}$$

```r
women_murdered_filtered =
  women_murdered |> # dataset read in previous lines of code
  dplyr::filter(year > 2000)

print(women_murdered_filtered)
```

Output

```
## # A tibble: 6 × 3
##   country        year murders_
##   <chr>         <dbl>
## 1 United States  2001
## 2 United States  2002
## 3 United States  2003
## 4 United States  2004
## 5 United States  2005
## 6 United States  2006
```

Computing the First Nonseasonal Difference

The change in the time series from one period to the next is known as the first nonseasonal difference. It can be computed as follows:

$$DY_t = Y_t - Y_{t-1}$$

```r
women_murdered_filtered =
  women_murdered |> # dataset read in previous lines of code
  dplyr::filter(year > 2000)

women_murdered_filtered =
  women_murdered_filtered |>
  dplyr::mutate(
    diff1 = murders_per_100000 - dplyr::lag(murders_per_100000, n = 1
    diff2 = c(NA, diff(murders_per_100000, lag = 1))
  )

print(women_murdered_filtered)
```

Output

```
## # A tibble: 6 × 5
##   country        year murders_
##   <chr>         <dbl>
## 1 United States  2001
## 2 United States  2002
## 3 United States  2003
## 4 United States  2004
## 5 United States  2005
## 6 United States  2006
```

Computing the First Nonseasonal Difference

The change in the time series from one period to the next is known as the first nonseasonal difference. It can be computed as follows:

$$DY_t = Y_t - Y_{t-1}$$

```python
import pandas as pd
import numpy as np

# Reading the CSV file (Equivalent to readr::read_csv in R)
women_murdered = pd.read_csv('../../data/murdered_women_per_100000_people.csv')

# Filtering for 'United States' (Equivalent to dplyr::filter in R)
women_murdered = women_murdered[women_murdered['country'] == 'United States']

# Melting the DataFrame to make it longer (Equivalent to tidyr::pivot_longer in R)
women_murdered = pd.melt(women_murdered, id_vars=['country'], var_name='year', value_name='murders_per_100000')

# Converting 'year' column to numeric (Equivalent to as.numeric in R)
women_murdered['year'] = pd.to_numeric(women_murdered['year'], errors='coerce')

# Removing rows with NA values (Equivalent to na.omit in R)
women_murdered = women_murdered.dropna()

# Filtering for years greater than 2000 and creating a copy to avoid warnings
women_murdered_filtered = women_murdered[women_murdered['year'] > 2000].copy()

# Calculating the differences in 'murders_per_100000' (Equivalent to dplyr::mutate and diff in R)
women_murdered_filtered['diff1'] = women_murdered_filtered['murders_per_100000'].diff()
women_murdered_filtered['diff2'] = np.append(np.nan, np.diff(women_murdered_filtered['murders_per_100000']))

are_columns_identical = women_murdered_filtered['diff1'].equals(women_murdered_filtered['diff2'])
```

Computing the First Seasonal Difference

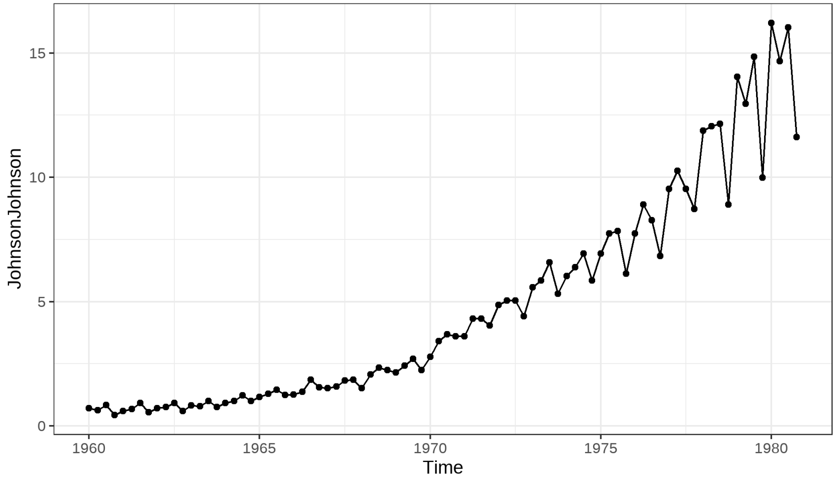
If your data exhibits a seasonal pattern, as illustrated in 04_ts_eda.html, you should employ a **seasonal differencing approach**, you should subtract the difference between an observation and the previous observation from the same season. Let m denote the number of seasons, e.g. $m = 4$ for quarterly data. In such a case, the seasonal difference is computed as follows:

$$DY_{t-m} = Y_t - Y_{t-m}$$

Note: In R, this can be computed by assigning the x argument in the `dplyr::lag()` to m, or by setting the lag argument in the `diff()` to m. In Python, this can be computed by setting the 'periods' argument in the `DataFrame.diff()` method to m.

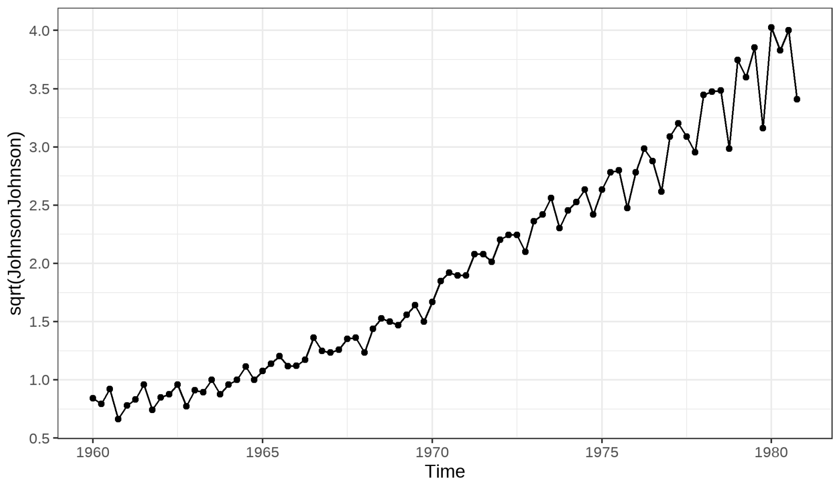
Stablize the Variance: Power Transformations

```
# The built-in JohnsonJohnson dataset

forecast::autoplot(JohnsonJohnson) +
  ggplot2::geom_point() + # adding points
  ggplot2::scale_x_continuous(breaks = scal
  ggplot2::scale_y_continuous(breaks = scal
  ggplot2::theme_bw()
```



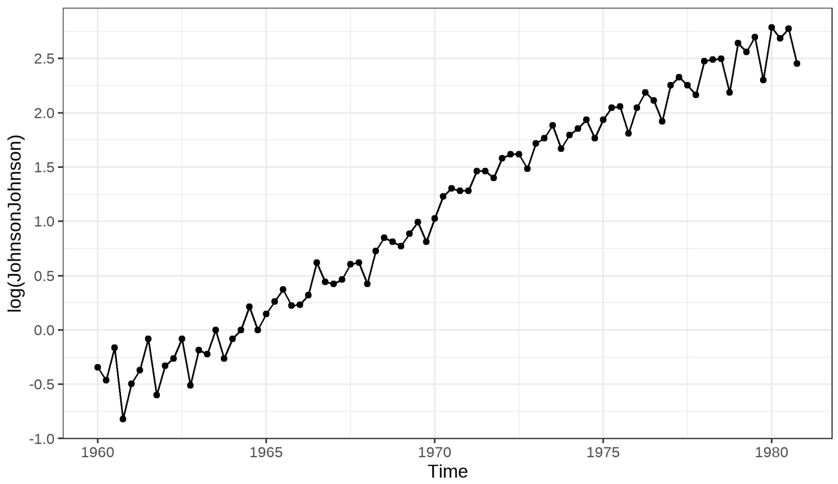
Stablize the Variance: Power Transformations

```r
# The built-in JohnsonJohnson dataset

forecast::autoplot(sqrt(JohnsonJohnson)) +
  ggplot2::geom_point() + # adding points
  ggplot2::scale_x_continuous(breaks = scal
  ggplot2::scale_y_continuous(breaks = scal
  ggplot2::theme_bw()
```



Stablize the Variance: Power Transformations

```r
# The built-in JohnsonJohnson dataset

forecast::autoplot(log(JohnsonJohnson)) +
  ggplot2::geom_point() + # adding points
  ggplot2::scale_x_continuous(breaks = scal
  ggplot2::scale_y_continuous(breaks = scal
  ggplot2::theme_bw()
```



A Note on the Log Transform

The log transformation can be computed as follows:

$$L_t = \ln\left(Y_t\right)$$

Note that the `log()` in both R and Python takes the natural logarithm as its default base, i.e., would transform a variable/statistic based on the above equation.

The reverse transformation using the exponential function is:

$$e^{L_t} = e^{\ln\left(Y_t\right)} = Y_t$$

The Log Transform

- The primary purpose of the log transform is to **convert exponential growth into linear growth.**
- The transform often has the **secondary purpose of balancing the variance.**
- Difference in logs and growth rate transformations produce similar results and interpretations (see next slides).

Stabilizing the Mean and Variance

The **first nonseasonal difference in logarithms** represents the logarithm of the ratio

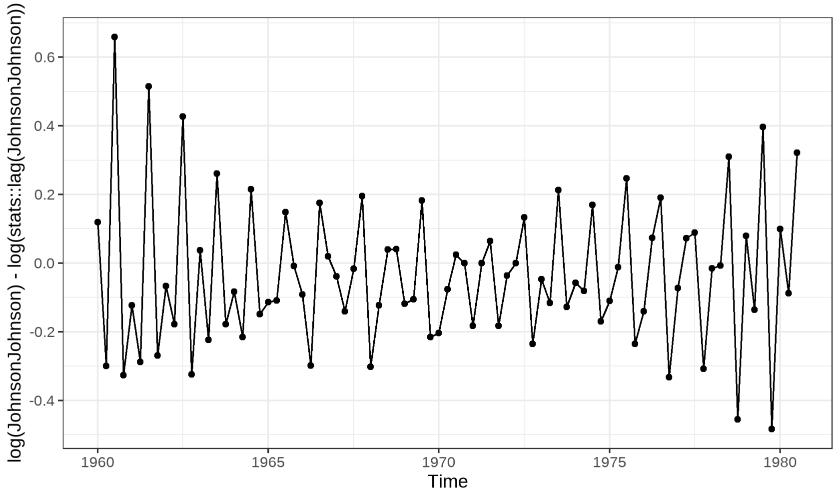
$$L_t = \ln\left(\frac{Y_t}{Y_{t-1}}\right) = \ln(Y_t) - \ln(Y_{t-1})$$

In the absence of seasonality, the **growth rate** for a time series is given by

$$GY_t = 100\frac{Y_t - Y_{t-1}}{Y_{t-1}}$$

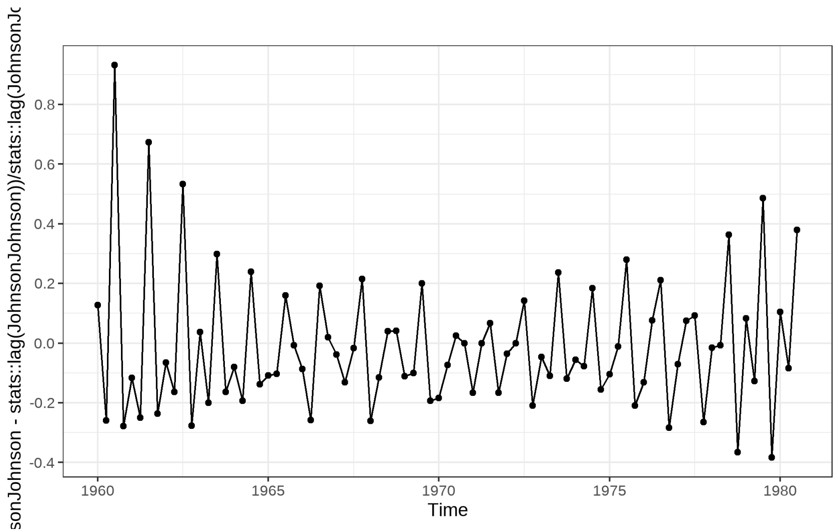
Stabilizing the Mean and Variance

```r
# The built-in JohnsonJohnson dataset

forecast::autoplot(
  log(JohnsonJohnson) - log(stats::lag(John
  ) +
  ggplot2::geom_point() + # adding points
  ggplot2::scale_x_continuous(breaks = scal
  ggplot2::scale_y_continuous(breaks = scal
  ggplot2::theme_bw()
```



Stabilizing the Mean and Variance

```
# The built-in JohnsonJohnson dataset

forecast::autoplot(
  (JohnsonJohnson - stats::lag(JohnsonJohns
  ) +
  ggplot2::geom_point() + # adding points
  ggplot2::scale_x_continuous(breaks = scal
  ggplot2::scale_y_continuous(breaks = scal
  ggplot2::theme_bw()
```



A Practical Note about Growth Rates

05:00

Activity Q1 Q2

> Over the next 5 minutes, please answer the question in each tab.

A Practical Note about Growth Rates

05:00

Activity | Q1 | Q2

- **Question 1:** Let us say that an investor purchased 10 stocks of \$GME, on 2021-01-29, at 325/stock. The next trading day, 2021-02-01, the GME stock closed at $225. Compute the growth rate in their portfolio worth (assuming it only has the GME stock) over this time period.

What is their growth rate? (Insert below)

- Edit me

A Practical Note about Growth Rates

05:00

Activity Q1 Q2

- **Question 2:** Let us say that the growth rate, $GY_t = -g$. Now let us assume that the GME stock went up by g (i.e., if it went down 10%, it increased by 10% over the next trading day). What is the value of the investor's portfolio by stock market closing on 2021-02-02?

What is their growth rate? (Insert below)

- Edit me

A Live Demo

In this live coding session, we will perform the following transformations on the AAPL stock [YTD] in both and :

- Growth Rates
- Natural log
- Log Differences
- $[0-1]$ Scaling

Recap

Summary of Main Points

By now, you should be able to do the following:

- Apply transformations to a time series in both R and Python.

Things to Do to Prepare for Our Next Class

- Go over your notes and read through Chapter 2.1-2.5 of our reference book.
- Complete Assignment 03 on Canvas.