

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

Spring 2023

Quick Refresher from Last Class

- ✓ Examine a line chart for trends, seasonality, and cycles.
- ✓ Explain the grammar of graphics and how it can be used to create time series plots in .
- ✓ Create interactive time-series plots by using the [plotly](#) package .

Learning Objectives for Today's Class

- Use numerical summaries to describe a time series.
- Explain what do we mean by correlation.
- Apply transformations to a time series.

A Summarizing Time-Series Data

Measures of Average

Mean: Given a set of n values Y_1, Y_2, \dots, Y_n , the arithmetic mean can be computed as:

$$\bar{Y} = \frac{Y_1 + Y_2 + \dots + Y_n}{n} = \frac{1}{n} \sum_{i=1}^n Y_i.$$

Order Statistics: Given a set of n values Y_1, Y_2, \dots, Y_n , we place them in an ascending order to define the order statistics, written as $Y_{(1)}, Y_{(2)}, \dots, Y_{(n)}$.

Median:

- If n is odd, $n = 2m + 1$ and the median is $Y_{(m+1)}$.
- If n is even, $n = 2m$ and the median is the average of the two middle numbers, i.e., $\frac{1}{2}[Y_{(m)} + Y_{(m+1)}]$.

Measures of Variation

The **range** denotes the difference between the largest and smallest value in a sample:

$$Range = Y_{(n)} - Y_{(1)}.$$

The **deviation** is defined as the difference between a given observation Y_i and the mean \bar{Y} .

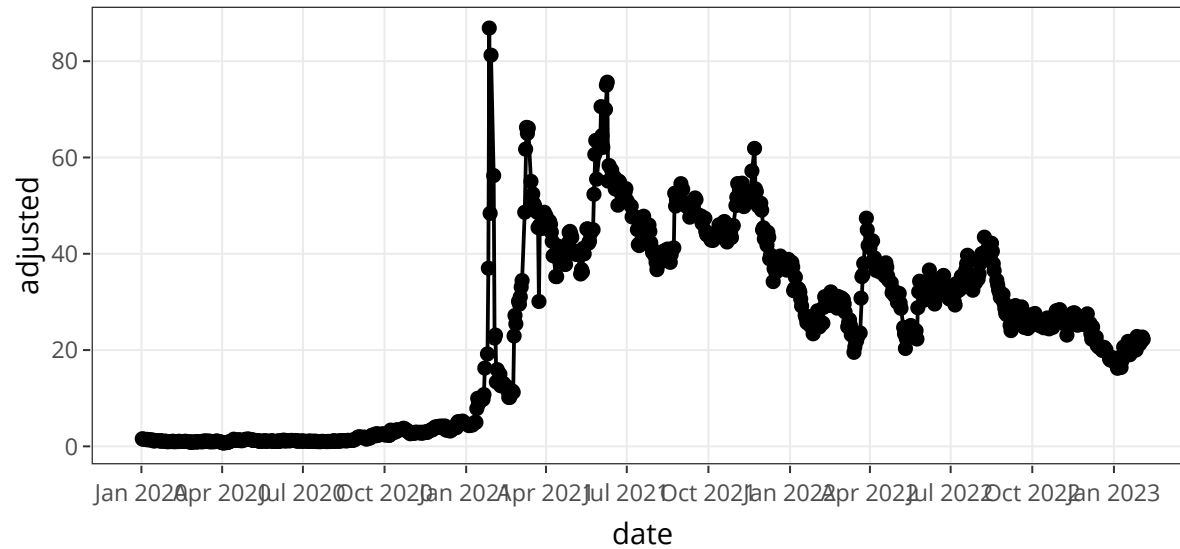
The **mean absolute deviation (MAD)** is the average deviations about the mean, irrespective of their sign:

$$MAD = \frac{\sum_{i=1}^{i=n} |d_i|}{n}.$$

The **variance** is the average of the squared deviations around the mean:

$$S^2 = \frac{\sum_{i=1}^{i=n} d_i^2}{n - 1}.$$

The GameStop Short Squeeze



Summarizing the GME Short Squeeze: Avg/Var Measures

```
gme_get =  
  tidyquant::tq_get(x = 'GME', from = '2020-01-01') |>  
  dplyr::select(date, adjusted) |>  
  dplyr::mutate(  
    year = lubridate::year(date),  
    month = lubridate::month(date, label = T)  
  )  
gme_get
```

```
## # A tibble: 779 × 4  
##   date          adjusted year  
##   <date>          <dbl> <dbl>  
## 1 2020-01-02      1.58  2020  
## 2 2020-01-03      1.47  2020  
## 3 2020-01-06      1.46  2020  
## 4 2020-01-07      1.38  2020  
## 5 2020-01-08      1.43  2020  
## 6 2020-01-09      1.39  2020  
## 7 2020-01-10      1.36  2020  
## 8 2020-01-13      1.36  2020  
## 9 2020-01-14      1.18  2020  
## 10 2020-01-15      1.15  2020  
## # ... with 769 more rows
```


Summarizing the GME Short Squeeze: Avg/Var Measures

```
gme_get =  
  tidyquant::tq_get(x = 'GME', from = '2020-01-01') |>  
  dplyr::select(date, symbol, adjusted) |>  
  dplyr::mutate(  
    year = lubridate::year(date),  
    month = lubridate::month(date, label = T)  
  )  
  
gme_summary =  
  gme_get |>  
  dplyr::group_by(symbol)  
  
gme_summary
```

```
## # A tibble: 779 × 5  
## # Groups:   symbol [1]  
##   date      symbol adjusted  
##   <date>     <chr>      <dbl>  
## 1 2020-01-02 GME         1.58  
## 2 2020-01-03 GME         1.47  
## 3 2020-01-06 GME         1.46  
## 4 2020-01-07 GME         1.38  
## 5 2020-01-08 GME         1.43  
## 6 2020-01-09 GME         1.39  
## 7 2020-01-10 GME         1.36  
## 8 2020-01-13 GME         1.36  
## 9 2020-01-14 GME         1.18  
## 10 2020-01-15 GME         1.15  
## # ... with 769 more rows
```

Summarizing the GME Short Squeeze: Avg/Var Measures

```
gme_get =  
  tidyquant::tq_get(x = 'GME', from = '2020-01-01') |>  
  dplyr::select(date, symbol, adjusted) |>  
  dplyr::mutate(  
    year = lubridate::year(date),  
    month = lubridate::month(date, label = T)  
  )  
  
gme_summary =  
  gme_get |>  
  dplyr::group_by(symbol) |>  
  dplyr::summarise(  
    adjusted_avg = mean(adjusted),  
    adjusted_med = median(adjusted),  
    adjusted_var = var(adjusted),  
    adjusted_sd = sd(adjusted)  
  )  
  
gme_summary |> t() # transposing for printout
```

```
##           [,1]  
## symbol      "GME"  
## adjusted_avg "24.44359"  
## adjusted_med "26.05"  
## adjusted_var "358.923"  
## adjusted_sd  "18.94526"
```

Summarizing the GME Short Squeeze: Avg/Var Measures

```
gme_get =  
  tidyquant::tq_get(x = 'GME', from = '2020-01-01') |>  
  dplyr::select(date, symbol, adjusted) |>  
  dplyr::mutate(  
    year = lubridate::year(date),  
    month = lubridate::month(date, label = T)  
  )  
  
gme_summary =  
  gme_get |>  
  dplyr::group_by(symbol, year) |>  
  dplyr::summarise(  
    adjusted_avg = mean(adjusted),  
    adjusted_med = median(adjusted),  
    adjusted_var = var(adjusted),  
    adjusted_sd = sd(adjusted)  
  )  
  
gme_summary
```

```
## # A tibble: 4 × 6  
## # Groups:   symbol [1]  
##   symbol  year adjusted_avg ad:  
##   <chr>   <dbl>         <dbl>  
## 1 GME     2020          1.79  
## 2 GME     2021         42.4  
## 3 GME     2022         29.6  
## 4 GME     2023         19.9
```

Summarizing the GME Short Squeeze: Avg/Var Measures

```
gme_get =  
  tidyquant::tq_get(x = 'GME', from = '2020-01-01') |>  
  dplyr::select(date, symbol, adjusted) |>  
  dplyr::mutate(  
    year = lubridate::year(date),  
    month = lubridate::month(date, label = T)  
  )  
  
gme_summary =  
  gme_get |>  
  dplyr::group_by(symbol, year, month) |>  
  dplyr::summarise(  
    adjusted_avg = mean(adjusted),  
    adjusted_med = median(adjusted),  
    adjusted_var = var(adjusted),  
    adjusted_sd = sd(adjusted)  
  )  
  
print(gme_summary, n=15)
```

```
## # A tibble: 38 × 7  
## # Groups:   symbol, year [4]  
##   symbol  year month adjusted_  
##   <chr>  <dbl> <ord>      <dbl>  
## 1 GME    2020 Jan      1.0  
## 2 GME    2020 Feb      0.0  
## 3 GME    2020 Mar      1.0  
## 4 GME    2020 Apr      1.0  
## 5 GME    2020 May      1.0  
## 6 GME    2020 Jun      1.0  
## 7 GME    2020 Jul      1.0  
## 8 GME    2020 Aug      1.0  
## 9 GME    2020 Sep      2.0  
## 10 GME   2020 Oct      3.0  
## 11 GME   2020 Nov      3.0  
## 12 GME   2020 Dec      4.0  
## 13 GME   2021 Jan     19.0  
## 14 GME   2021 Feb     17.0  
## 15 GME   2021 Mar     47.0  
## # ... with 23 more rows
```

Correlation

The Pearson Correlation Coefficient

- **Correlation:** measures the strength of the **linear relationship** between two quantitative variables.
- It can be computed using the `cor()` from base R. Mathematically speaking, the pearson correlation coefficient, r , can be computed as

$$r = \frac{\sum_{i=1}^n (X_i - \bar{X})(Y_i - \bar{Y})}{\sqrt{\sum_{i=1}^n (X_i - \bar{X})^2 \sum_{i=1}^n (Y_i - \bar{Y})^2}}$$

- Do **not** use the Pearson Correlation coefficient if both variables are not quantitative. Instead, refer to the `mixed.cor()` from the **psch package** to compute the correlations for mixtures of continuous, polytomous, and/or dichotomous variables.
- You should supplement **any descriptive summaries with visualizations** to ensure that you are able to interpret the computations correctly.

Supplement Summaries with Viz: Anscombe's Dataset

In a seminal paper, Anscombe stated:

Few of us escape being indoctrinated with these notions:

- numerical **calculations are exact, but graphs are rough**;
- for any particular kind of **statistical data there is just one set of calculations constituting a correct statistical analysis**;
- performing **intricate calculations is virtuous**, whereas **actually looking at the data is cheating**.

He proceeded by stating that

a computer should **make both calculations and graphs**. Both sorts of output should be studied; each will contribute to understanding.

Now, let us consider his four datasets, each consisting of eleven (x,y) pairs.

Supplement Summaries with Viz: Anscombe's Dataset

| x1 ▾ | x2 ▾ | x3 ▾ | x4 ▾ | y1 ▾ | y2 ▾ | y3 ▾ | y4 ▾ |
|------|------|------|------|-------|------|-------|------|
| 10 | 10 | 10 | 8 | 8.04 | 9.14 | 7.46 | 6.58 |
| 8 | 8 | 8 | 8 | 6.95 | 8.14 | 6.77 | 5.76 |
| 13 | 13 | 13 | 8 | 7.58 | 8.74 | 12.74 | 7.71 |
| 9 | 9 | 9 | 8 | 8.81 | 8.77 | 7.11 | 8.84 |
| 11 | 11 | 11 | 8 | 8.33 | 9.26 | 7.81 | 8.47 |
| 14 | 14 | 14 | 8 | 9.96 | 8.1 | 8.84 | 7.04 |
| 6 | 6 | 6 | 8 | 7.24 | 6.13 | 6.08 | 5.25 |
| 4 | 4 | 4 | 19 | 4.26 | 3.1 | 5.39 | 12.5 |
| 12 | 12 | 12 | 8 | 10.84 | 9.13 | 8.15 | 5.56 |
| 7 | 7 | 7 | 8 | 4.82 | 7.26 | 6.42 | 7.91 |
| 5 | 5 | 5 | 8 | 5.68 | 4.74 | 5.73 | 6.89 |

Showing 1 to 11 of 11 entries

Previous

1

Next

Supplement Summaries with Viz: Anscombe's Dataset

| set | x.mean | x.sd | y.mean | y.sd | corr |
|-----|--------|------|--------|------|------|
| I | 9 | 3.32 | 7.5 | 2.03 | 0.82 |
| II | 9 | 3.32 | 7.5 | 2.03 | 0.82 |
| III | 9 | 3.32 | 7.5 | 2.03 | 0.82 |
| IV | 9 | 3.32 | 7.5 | 2.03 | 0.82 |

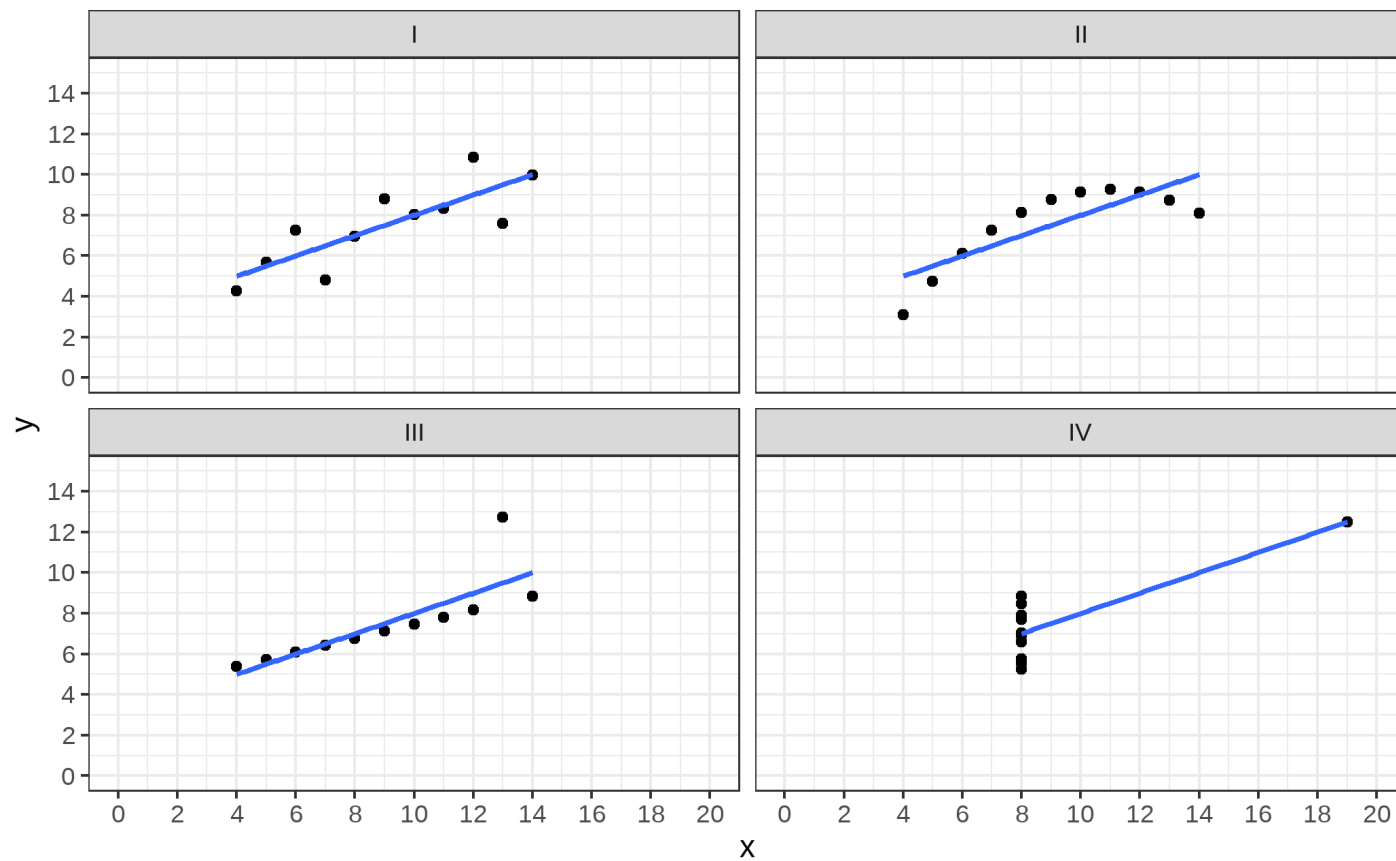
Showing 1 to 4 of 4 entries

Previous

1

Next

Supplement Summaries with Viz: Anscombe's Dataset



Kahoot Competition #02

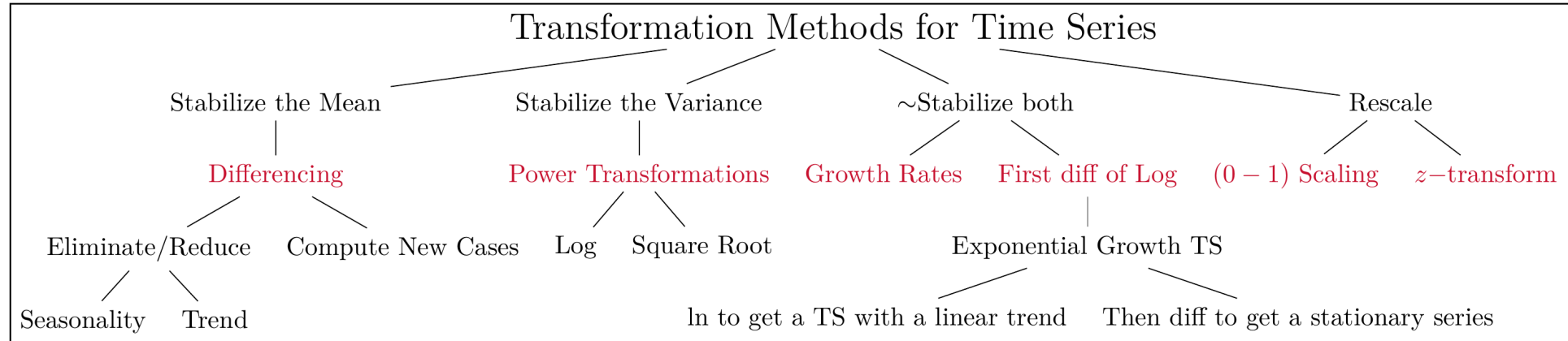
To assess your understanding and retention of the topics covered so far, you will **compete in a Kahoot competition (consisting of 6 questions)**:

- Go to <https://kahoot.it/>
- Enter the game pin, which will be shown during class
- Provide your first (preferred) and last name
- Answer each question within the allocated time window (**fast and correct answers provide more points**)

Winning the competition involves having as many correct answers as possible AND taking the shortest duration to answer these questions. The winner 🏆 of the competition will receive a \$10 Starbucks gift card. Good luck!!!

Transformations

Guidelines for Transforming Time Series Data



A classification of common transformation approaches for time series data

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:

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

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

```
## # 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:

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

```
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)
```

```
## # 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 Seasonal Difference

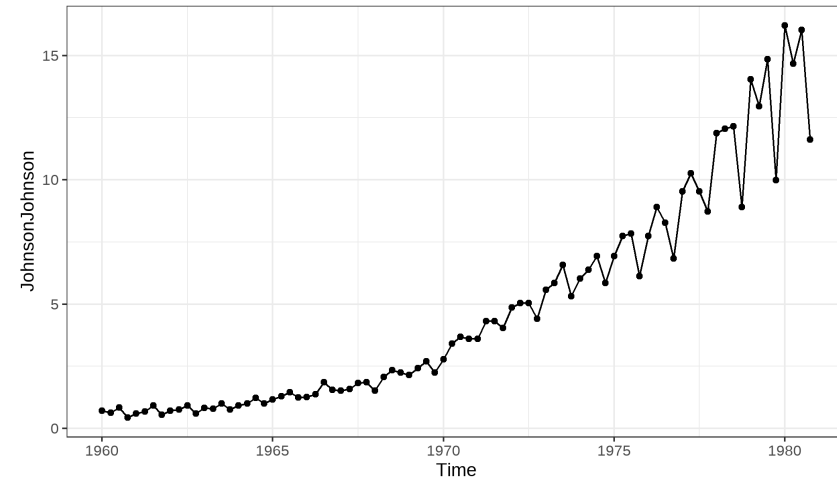
If your data exhibits a seasonal pattern, as illustrated in Slides 6 and 18 in [03_ts_viz.html](#), you should employ a **seasonal differencing approach**, you should subtract the difference between an observation and the previous observation from the same season. Let m denote the number of seasons, 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 .

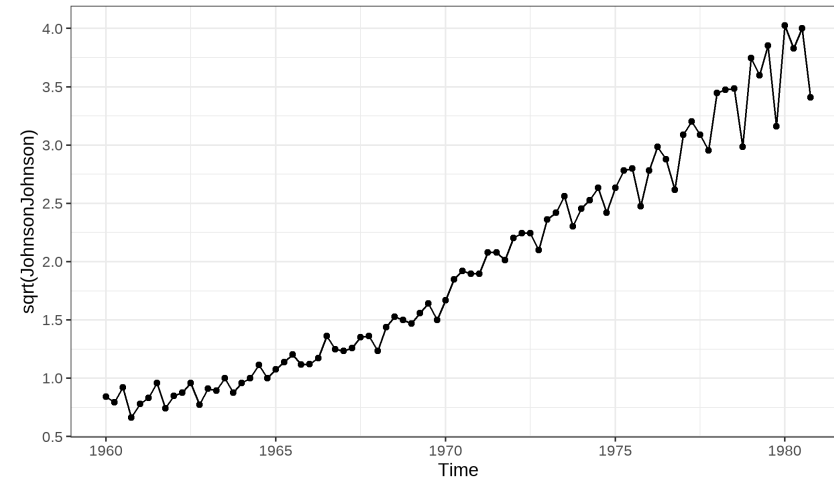
Stablize the Variance: Power Transformations

```
# 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

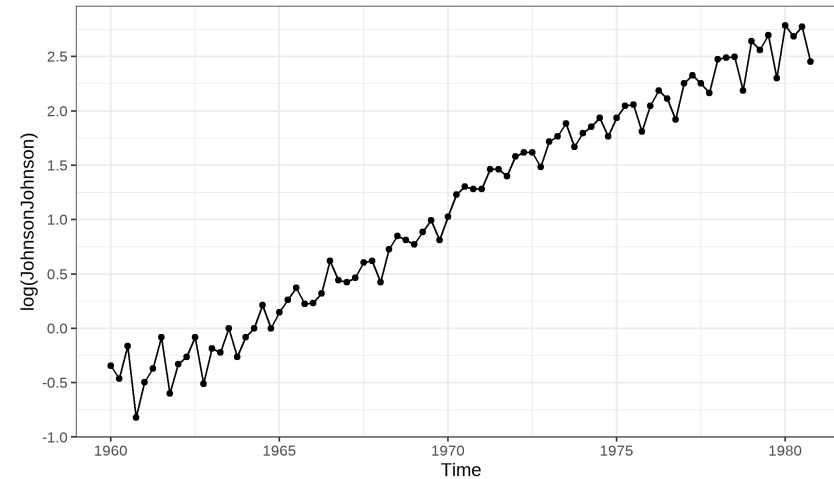
```
# The built-in JohnsonJohnson dataset  
forecast::autoplot(sqrt(JohnsonJohnson)) +  
  ggplot2::geom_point() + # adding points  
  ggplot2::scale_x_continuous(breaks = scal  
  ggplot2::scale_y_continuous(breaks = scal  
  ggplot2::theme_bw()
```



Stablize the Variance: Power Transformations

```
# 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:

$$L_t = \ln(Y_t)$$

Note that the `log()` in R 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:

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

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

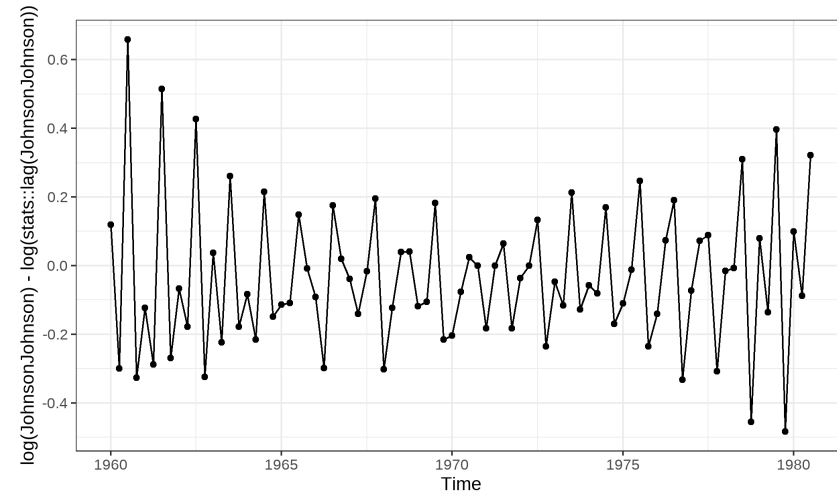
$$L_t = \ln\left(\frac{Y_t}{Y_{t-1}}\right) = \ln(Y_t) - \ln(Y_{t-1})$$

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

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

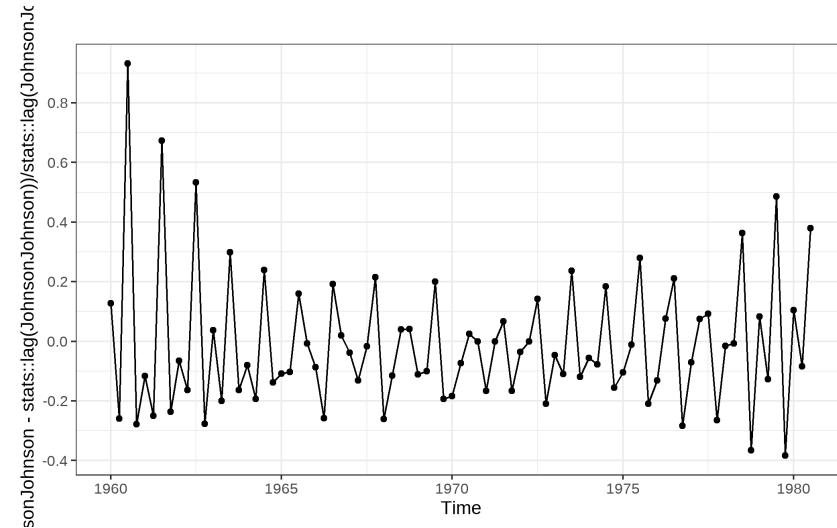
Stabilizing the Mean and Variance

```
# 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(JohnsonJohnson, 1))  
  ) +  
  ggplot2::geom_point() + # adding points  
  ggplot2::scale_x_continuous(breaks = scale_x_year_major())  
  ggplot2::scale_y_continuous(breaks = scale_y_continuous())  
  ggplot2::theme_bw()
```



05:00

A Practical Note about Growth Rates

| Activity | Q1 | Q2 |
|----------|----|----|
|----------|----|----|

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

A Practical Note about Growth Rates

| Activity | Q1 | Q2 |
|----------|----|----|
|----------|----|----|

- **Question 1:** Let us say that an investor purchased 10 stocks of $\backslash \$GME$, 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

| 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 capitalize on the `mutate()` from `tidyverse` to create transformations for multiple time series. Specifically, we will use the `tq_get()` from `tidyquant` to extract data about the following cryptocurrencies (a) `Cardano` (ADA), (b) `Chainlink` (LINK), and (c) `Zilliqa` (ZIL). We will compute:

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

Obviously, we will have to ensure that these transformations are computed for each coin separately. For the purpose of this activity, let us extract the data from 2023-01-01 to 2023-02-05.

Recap

Summary of Main Points

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

- Use numerical summaries to describe a time series.
- Explain what do we mean by correlation.
- Apply transformations to a time series.

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 04](#) on Canvas.