

***Rotman***

# INTRO TO R – TIME SERIES & FINANCE PACKAGES

R Workshop – Part 4 (2)

March 23, 2023 Prepared by Jay Cao / TDMDAL

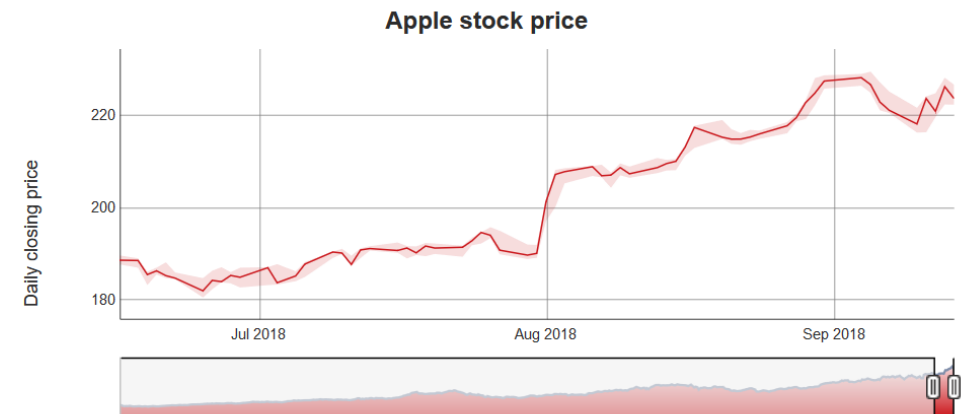
Website: <https://tdmdal.github.io/r-workshop-202223-winter/>



Rotman School of Management  
UNIVERSITY OF TORONTO

# What's Time Series (TS)

- A series of values obtained at successive times
  - A series of numerical values
  - With associated timestamps (or start, end, and frequency if equi-interval)
- Typical operations on a time series
  - lead, lag, difference, rolling window aggregation, etc.
  - time-aware subsetting
- Typical statistics
  - moving average, returns, etc.
  - trend, seasonality, stationarity, etc.

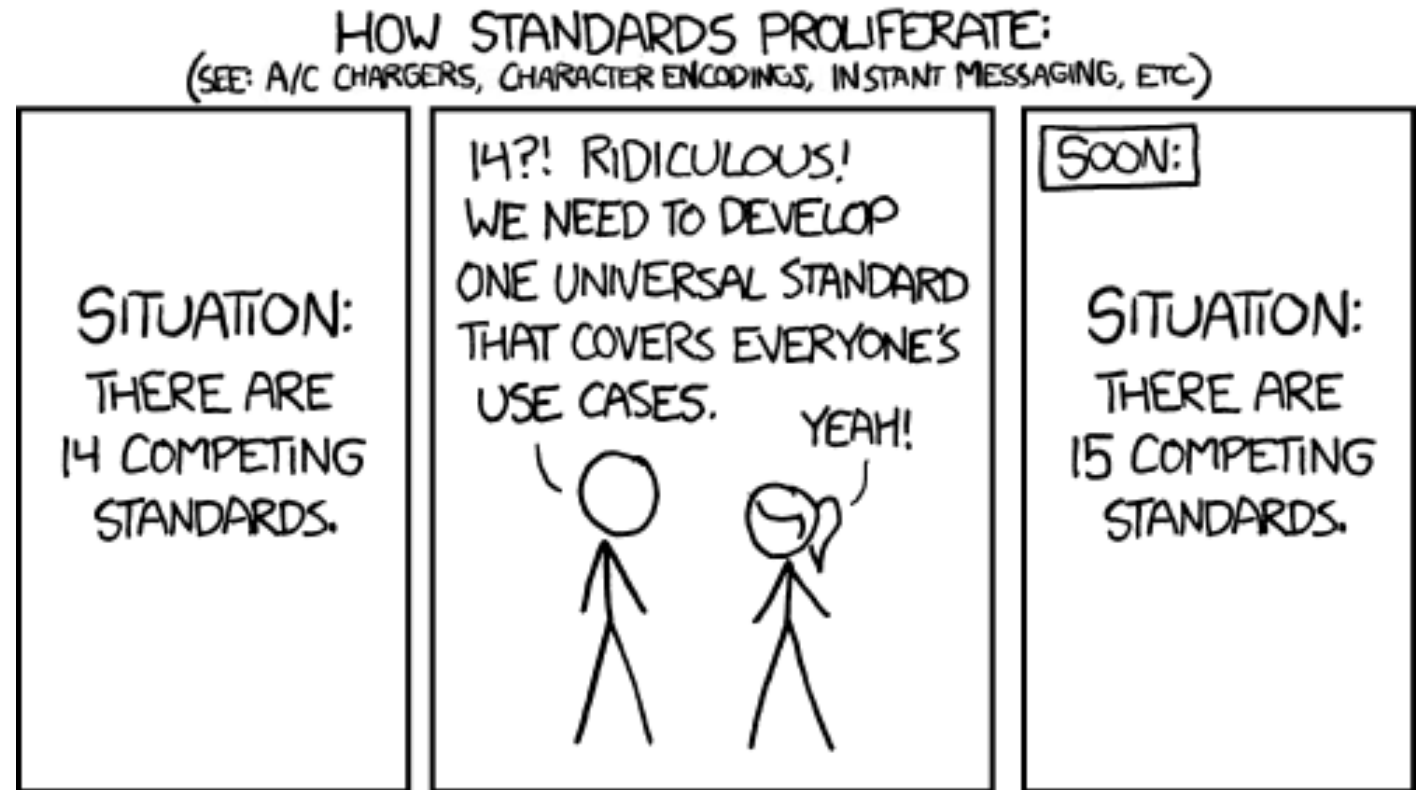


# How to Store Time Series (TS) in R

- From what we have seen so far
  - Vectors (with names as timestamps)
  - Matrices (with row names as timestamps)
  - **Dataframes/tibbles** with a timestamp column
- What we really need
  - Store time series efficiently
  - More importantly, be able to manipulate time series efficiently
    - i.e. need associated functions/packages that can efficiently operate on stored time series (lead, lag, smooth, moving average, etc.)

# Current Status of R TS Data Structures

- A vast number of Time Series Data Structures
- Each has associated packages
  - for TS manipulation
  - for TS analysis/modeling



# Oldies but Goodies

- **ts** class: a class for equi-spaced time series
  - what's a "class": a data structure with associated operations (methods)
- **zoo** class
  - can handle regular- and irregular-spaced time series
  - can use arbitrary classes for the timestamps
- **xts** class
  - built on **zoo** with more functions for data processing
  - uniform handling of R's time-based data classes (e.g. **zoo**, **timeSeries**, etc.)
- Many more
  - **timeSeries** class in **timeSeries** package
  - **tis** class from **tis** package

<https://cran.r-project.org/web/views/TimeSeries.html>

# New Kids in Town – Tidy Data Converts

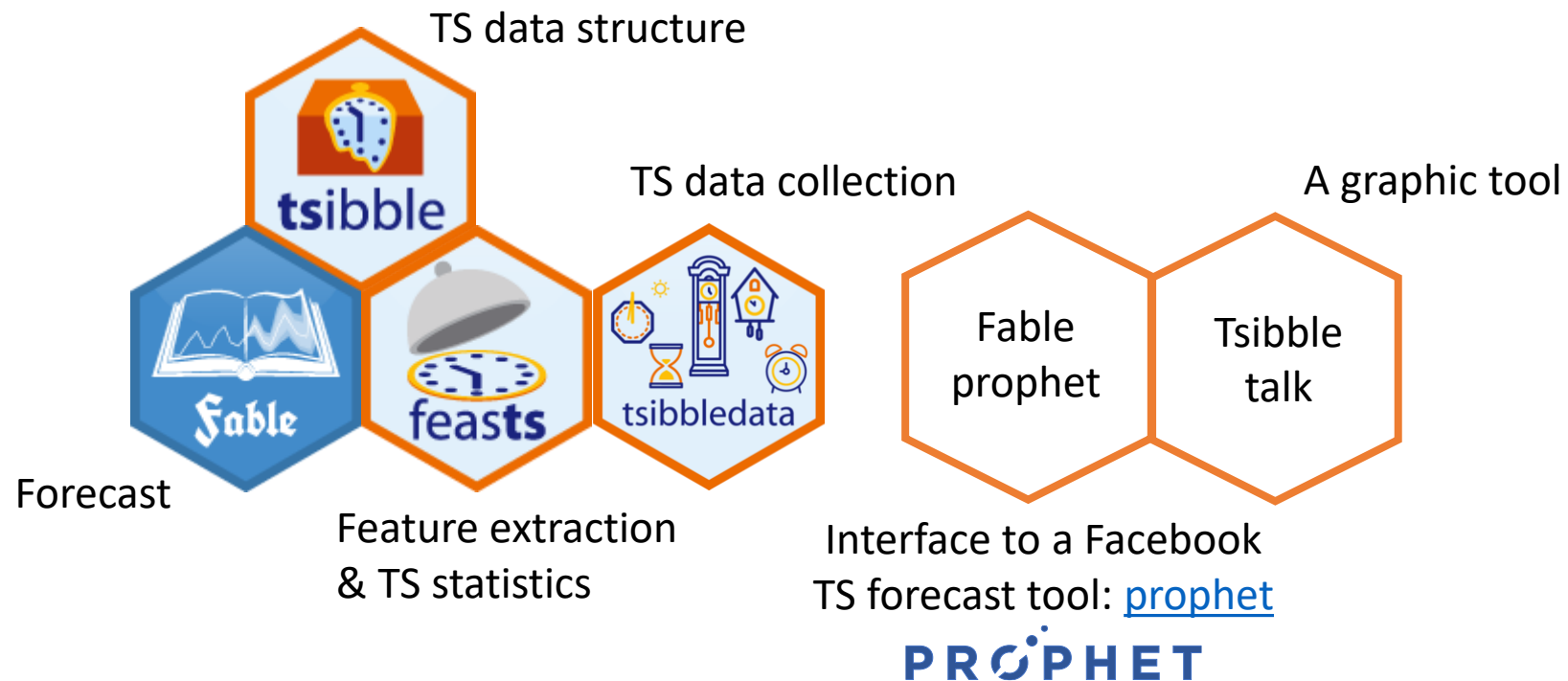
- [tsibble](#), a new time series class (`tbl_ts`) built on [tibble](#)
  - from tsibble package: time-based dataframe/tibble
  - Part of [tidyverts](#) ecosystem (not tidyverse) for time series
- [tibble](#), but make it “time-aware” whenever needed
  - [tidyquant](#)
    - e.g., convert `tibble` to `xts` when interfacing with other packages that operates on `xts`
  - [modetime](#) and its ecosystem
    - a consistent way to do time series analysis using many other time series packages
    - same team behind tidyquant
  - [prophet](#)
    - time series forecast (from Facebook) based on additive model
    - work directly with `tibble` from tidyverse

# Which One Should You Use

- Try “New Kids” and their eco-system first
  - They mostly follow the tidy data & tidyverse framework for data wrangling, something you are familiar with from tabular data
  - Not our focus today, but we will mention a few packages
- If “New Kids” can’t do the job, fall back to “Oldies but Goodies”
  - Many old but potentially useful TS packages still uses traditional TS DS
  - We will focus on ts and xts and a few related TS and finance packages today
- Use both (“hybrid-mode”)
  - convert between time series data structures (e.g. using [tsbox](#), etc.)

# New Kids TS Ecosystem (1)

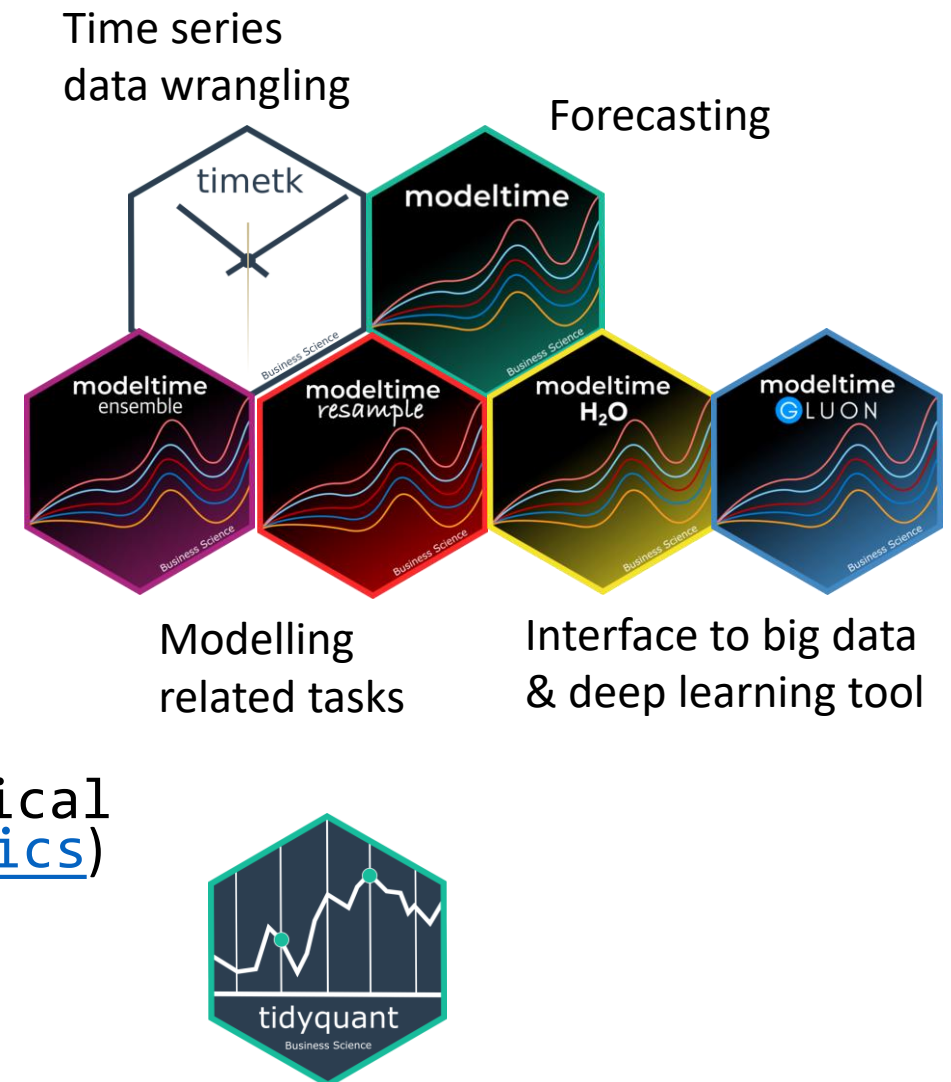
- [tsibble](#), a new time series class (`tbl_ts`) built on `tibble`
  - from `tsibble` package: time-based dataframe/tibble
  - Part of [tidyverts](#) ecosystem (not tidyverse) for time series





# New Kids TS Ecosystem (2)

- [modeltime](#) and its ecosystem
  - a consistent way to do time series analysis
  - build on many other time series packages
    - i.e., a super “wrapper” of other R packages
  - kind of like tidymodel for TS
  - very new so...
- [tidyquant](#)
  - integrates resources for collecting and analyzing financial data (xts, [quantmod](#), [TTR](#) (Technical Trading Rule) and [PerformanceAnalytics](#))
    - i.e., a super “wrapper” too
  - work with `tibble` from tidyverse



# New Kids TS Ecosystem (3)

- [prophet](#)
  - a specific time series forecast procedure based on additive model
    - robust to outliers, missing data, and dramatic changes in your time series
    - tunable
  - work directly with **tibble** from tidyverse
  - built by Facebook (Facebook Open Source)

The logo for the Prophet time series forecasting library. It features the word "PROPHET" in a bold, blue, sans-serif font. The letter "O" is stylized as a blue circle with a white dot in the center, resembling a prophetic symbol or a target.

# ts class

- A class for equi-spaced time series supported by base R
- Data is stored as
  - a vector (univariate) or matrix (multi-variate) with attributes...
  - “class”: ts
  - “tsp” (time series parameters): a numerical vector recording (start, end, freq)
- Many functions/packages work well with ts object
  - e.g., [forecast](#) package

# ts class – how are data stored / 1

```
> ts_obj <- ts(1:10, frequency = 4, start = c(2017, 2)) # 2nd Quarter of 2017
> ts_obj
      Qtr1 Qtr2 Qtr3 Qtr4
2017         1    2    3
2018    4    5    6    7
2019    8    9   10
>
> typeof(ts_obj)
[1] "integer"
>
> class(ts_obj)
[1] "ts"
```

# ts class – how are data stored / 2

```
> attributes(ts_obj)
$tsp
[1] 2017.25 2019.50    4.00

$class
[1] "ts"
```

# ts class – associated time-aware operations

```
> cycle(ts_obj)
      Qtr1 Qtr2 Qtr3 Qtr4
2017      2   3   4
2018   1   2   3   4
2019   1   2   3

>
> diff(ts_obj, 4)
      Qtr1 Qtr2 Qtr3 Qtr4
2018      4   4   4
2019   4   4   4

>
# see notebook for more, and the "forecast" packages intro
```

```
> ts_obj
      Qtr1 Qtr2 Qtr3 Qtr4
2017      1   2   3
2018   4   5   6   7
2019   8   9  10
```

# xts class

- xts extends zoo
  - zoo can handle regular- and irregular-spaced time series; so does xts
- xts can use arbitrary classes for timestamps
- Compatible with zoo and other time-series classes in other packages
- Many functions/packages work well with xts object
  - e.g., [forecast](#), [quantmod](#), and [PerformanceAnalytics](#)

# xts class – how are data stored / 1

```
> x <- matrix(1:6, ncol = 2)
> print(x)
      [,1] [,2]
[1,]    1    4
[2,]    2    5
[3,]    3    6
>
> idx <- as.Date(c("2019-01-01", "2019-01-02", "2019-01-05"))
> print(idx)
[1] "2019-01-01" "2019-01-02" "2019-01-05"
```



## xts class – how are data stored / 2

```
> xts_obj <- xts(x, order.by = idx)
```

```
> xts_obj
```

```
          [,1] [,2]
2019-01-01     1     4
2019-01-02     2     5
2019-01-05     3     6
```

```
> typeof(xts_obj)
```

```
[1] "integer"
```

```
> class(xts_obj)
```

```
[1] "xts" "zoo"
```

# xts class – how are data stored / 3

```
> str(attributes(xts_obj))
```

```
List of 3
```

```
$ dim : int [1:2] 3 2
```

```
$ index: num [1:3] 1.55e+09 1.55e+09 1.55e+09
```

```
..- attr(*, "tzone")= chr "UTC"
```

```
..- attr(*, "tclass")= chr "Date"
```

```
$ class: chr [1:2] "xts" "zoo"
```

# xts class – associated time-aware operations

```
> # use quantmod package to get data from yahoo finance
> library(quantmod)
> msft <- getSymbols("MSFT",
                    from = "2018-12-31",
                    to = "2019-12-31",
                    auto.assign = FALSE)

>
> # msft_xts is an xts object
> class(msft_xts)
[1] "xts" "zoo"
```

# xts class – associated time-aware operations

```
> # get data for all Monday in 2019 (time-aware subsetting)
> msft[.indexyear(msft) == (2019 - 1900) & .indexwday(msft) == 1]
```

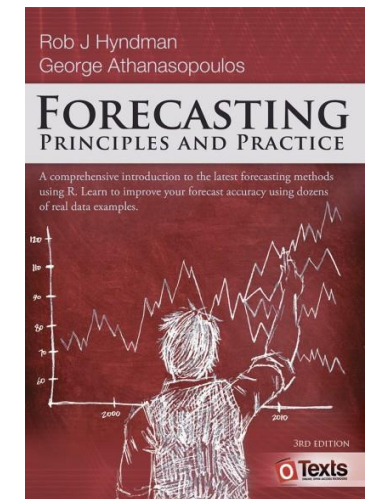
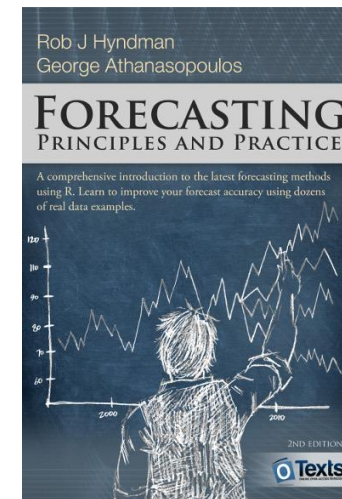
	MSFT.Open	MSFT.High	MSFT.Low	MSFT.Close	MSFT.Volume	MSFT.Adjusted
2019-01-07	101.64	103.27	100.98	102.06	35656100	102.06
2019-01-14	101.90	102.87	101.26	102.05	28437100	102.05
2019-01-28	106.26	106.48	104.66	105.08	29476700	105.08
2019-02-04	102.87	105.80	102.77	105.74	31315100	105.74

...

```
# see notebook for more, and the "PerformanceAnalytics" package intro
```

# Resources

- [a Little book of R for Time Series](#)
  - uses traditional time series data structures & related packages
- [Forecasting: Principles and Practice \(2<sup>nd</sup> ed\)](#)
  - uses forecast packages
- [Forecasting: Principles and Practice \(3<sup>rd</sup> ed.\)](#)
  - uses tsibble and fable packages
- [Financial Engineering Analytics: A Practice Manual Using R](#)



# Appendix 1: Too Many TS Data Structures!

- [tsbox](#)
  - provides conversion between many time series data structures
  - an attempt to unite time series data structure in R

# Appendix 2: RQuantLib

- QuantLib
  - a comprehensive free/open-source library for quantitative finance
  - Written in C++
- RQuantLib
  - makes part of QuantLib accessible from R
    - i.e., RQuantLib is an R interface to QuantLib
  - So far, good with
    - Option pricing
    - Fixed incomes