Manipulating Time Series Data in R

2022-05-16

*This course will introduce learners to working with time series data in R. Learners will explore how to store and format data in date and time objects as well as how to manipulate time series datasets through subsetting, indexing, and extraction. Examples of time series data across a variety of fields in business and science should be discussed. The course will cover summarization, frequency, missing data, resampling, and comparison techniques as well as window functions for both rolling and expanding windows.*

Packages Used:

* base and stats (default libraries, but I wanted to name them explicitly)
* zoo

# Chapter 1: Introduction to Time Series Data

* Lesson 1.1: *What is Time Series Data*
  + LO: Learner will be able to understand the foundations of time series data: rather than just analyzing a variable over time, study *how* that variable changes with time.
* Lesson 1.2: *Interpreting a Time Series*
  + LO: Learner will be able to interpret a time series graph, understanding the x- and y-axes, trend, periods, etc. at an introductory level
* Lesson 1.3: *Temporal data classes in R*
  + LO: Introduction to different formats for temporal data in R, such as the Date, numeric, and character formats:
    - e.g.: 2022-01-30, 19022, and “2022-01-30” share the same information, but in different formats
  + LO: Learners will be able to check classes of data stored as vectors or as columns in a dataframe or tibble.
    - class()
* Lesson 1.4: *Converting between data classes*
  + LO: Learners will be able to convert between classes in R, such as converting a character vector to a Date vector
    - as.Date()
    - as.numeric()
    - as.character()

# Chapter 2: Time Series objects in R

* Lesson 2.1: *How does R store Time Series Data?*
  + LO: Learners will be introduced to ts objects in R, and how they differ from objects like vectors or data frames
  + LO: Retrieve the temporal attributes (start, end, and frequency) of a time series object.
    - start()
    - end()
    - frequency()
* Lesson 2.2: *Create a Time Series object*
  + LO: Convert a vector of observations into a ts object, specifying start time and frequency
    - ts()
* Lesson 2.3: *Using the Zoo Package to store time series data*
  + LO: What is zoo and why is it different from base ts?
    - Zoo can use irregular time intervals
  + LO: Create and coerce time series objects with the zoo package:
    - zoo::zoo()
    - zoo::as.zoo()
* Lesson 2.4: *Using Zoo to extract time and data vectors*
  + LO: Extract “core data” and time data from a ts or zoo object:
    - time()
    - zoo::coredata()

# Chapter 3: Subsetting, Extracting, and Resampling

* Lesson 3.1: *Subsetting a window of observations*
  + LO: Learner will be able to extract a window of observations between a set of time intervals
    - window()
    - as.Date()
    - zoo::as.yearmon()
  + LO: Use the '[' operator with as.Date() to extract a specific date’s observation
    - '['
    - as.Date()
    - zoo::as.yearmon()
* Lesson 3.2: *Retrieving observations by index*
  + LO: Use standard R '[' operator to extract one or more observations by numerical index
    - '['
    - e.g.: data[1:20] retrieves observations 1 through 20
* Lesson 3.3: *Resampling observations*
  + LO: Learner will be able to resample observations to any interval of time (yearly, monthly, quarterly, etc.)
    - aggregate()
    - e.g.: aggregate(data, nfrequency = 12, FUN = sum) finds sums of observations within each month.
* Lesson 3.4: *Imputing Missing Values*
  + LO: Use the zoo package to impute missing values with either linear interpolation or cubic spline interpolation
    - zoo::na.approx() and zoo::na.spline(), respectively

# Chapter 4: Rolling and Expanding Windows

* Lesson 4.1: *What are windows?*
  + LO: Learner will understand the utility of rolling and expanding windows: finding moving averages, cumulative sums, etc.
* Lesson 4.2: *Calculating a Rolling Window*
  + LO: Learner will be able to perform a rolling window operation on a time series, creating a moving average (or moving sum) of any length
    - zoo::rollapply()
    - zoo::rollapplyr() (convenience wrapper for zoo::rollapply(align = "right"))
    - e.g.: zoo::rollapplyr(daily\_data, FUN = mean, width = 7) to create a 7-day rolling average from daily\_data
* Lesson 4.3: *Calculating an Expanding Window*
  + LO: Learner will be able to create an expanding window: a rolling window where the “start” is fixed and the “end” moves
    - cumsum()
    - seq\_along()
    - e.g.: cumsum(x) / seq\_along(x) calculates a cumulative mean
    - *this function exists in dplyr::cummean(), but I didn’t want an entire package dependency for something so simple*
* Lesson 4.4: *Plotting windows alongside Data*
  + LO: Learner will be able to plot the rolling/expanding window alongside the original data, in order to visually assess how these operations affect the data
    - plot()
    - lines()