Assignment 3B- Window Functions

Kiera Griffiths

Approach:

For this assignment, I used dplyr in R to analyze CPI (Consumer Price Index) data from the FRED (Federal Reserve Economic Data). Original CPI data is collected by the Bureau of Labor Statistics and maintained by the Federal Reserve Bank of St. Louis (FB St. Louis). I used data about the average price of a dozen eggs and a gallon of milk to calculate a six-month moving average (starting at January 1, 2022) which smooths short-term fluctuations and better understands pricing over time. I formatted both datasets to present prices by month, then merged both to be analyzed collectively.

 # 1. Load libraries

Dplyr helps transform data from datasets by grouping and calculating rows.

```
library(dplyr)
library(readr)
library(lubridate)
library(tidyr)
library(slider)
```

2. Load CSV files from URLs

```
eggs <- read_csv("https://raw.githubusercontent.com/KieraG2026/Window-Functions/refs/heads/main/FRED%20CPI%20Data-%20Dozen%20Eggs.csv")
milk <- read_csv("https://raw.githubusercontent.com/KieraG2026/Window-Functions/refs/heads/main/FRED%20CPI%20Data-%20Gallon%20Milk.csv")
```

3. Convert observation_date columns to date format

```
eggs <- eggs %>% mutate(observation_date = ymd(observation_date))
milk <- milk %>% mutate(observation_date = ymd(observation_date))
```

```
# 4. Filter data from Jan 1, 2022 onwards
eggs <- eggs %>% filter(observation_date >= as.Date("2022-01-01"))
milk <- milk %>% filter(observation_date >= as.Date("2022-01-01"))

# 5. Rename price columns
colnames(eggs)[2] <- "Eggs"
colnames(milk)[2] <- "Milk"

# 6. Merge datasets by observation_date
df <- left_join(eggs, milk, by = "observation_date")

# 7. Convert to long format
df <- df %>%
  pivot_longer(cols = c(Eggs, Milk),
               names_to = "Item",
               values_to = "Price")

# 8. Add Year column
df <- df %>%
  mutate(Year = year(observation_date))

# 9. Calculate Year-to-Date (YTD) average
df <- df %>%
  arrange(observation_date) %>%
  group_by(Item, Year) %>%
  mutate(YTD_avg = cummean(Price))

# 10. Calculate 6-Observation Moving Average
# Note: CPI data is monthly, so this represents a 6-month moving average
df <- df %>%
  group_by(Item) %>%
  mutate(MA_6 = slide_dbl(Price,
```

```
        mean,
        .before = 5,
        .complete = TRUE))

# 11. View final table
print(df, n=14)
```

```
# A tibble: 96 x 6
# Groups:   Item [2]
   observation_date Item  Price  Year YTD_avg  MA_6
   <date>           <chr> <dbl> <dbl>   <dbl> <dbl>
 1 2022-01-01       Eggs   1.93  2022    1.93 NA
 2 2022-01-01       Milk   3.79  2022    3.79 NA
 3 2022-02-01       Eggs   2.00  2022    1.97 NA
 4 2022-02-01       Milk   3.88  2022    3.83 NA
 5 2022-03-01       Eggs   2.05  2022    1.99 NA
 6 2022-03-01       Milk   3.92  2022    3.86 NA
 7 2022-04-01       Eggs   2.52  2022    2.12 NA
 8 2022-04-01       Milk   4.01  2022    3.90 NA
 9 2022-05-01       Eggs   2.86  2022    2.27 NA
10 2022-05-01       Milk   4.20  2022    3.96 NA
11 2022-06-01       Eggs   2.71  2022    2.35  2.34
12 2022-06-01       Milk   4.15  2022    3.99  3.99
13 2022-07-01       Eggs   2.94  2022    2.43  2.51
14 2022-07-01       Milk   4.16  2022    4.01  4.05
```

The dataset shows the monthly prices of eggs and milk in 2022 and later. Each row includes the observation_date (the date of the price observation), Item (either Eggs or Milk), Price (the monthly CPI price), Year (extracted from the date), YTD_avg (the year-to-date average), and MA_6 (the six-period moving average). The YTD average is calculated using a window function, “cummean()”, which computes a running average from January 2022 to the current month for each item separately. The six-period moving average

smooths short-term fluctuations by averaging the current month with the previous five months, calculated with “slide_dbl()”. Since the data is monthly, this represents a six-month moving average rather than a daily one. Window functions like these are powerful because they perform calculations across a defined set of rows without collapsing the dataset, keeping one result per row while respecting the grouping by item or year. In the dataset, early months show NA for the moving average because there aren’t enough prior months to form a full six-month window. It’s also important to note the moving average calculations use only the previous six months, so the July calculations only go back to FEBRUARY, not January. Overall, YTD averages show cumulative trends within the year, while MA_6 highlights smoothed price trends for each item separately, revealing the gradual price increases over time.