Exercises 1-5

(For each exercise, place your solution below the exercise description.)

Exercise 1

This exercise demonstrates how to insert an image into an R Markdown notebook.

1. Download any version of a UGA logo image from the internet.
2. Use an image editor (e.g., Paint on Windows) to add your name to the image.
3. Upload the modified image to a publicly accessible web location (e.g., GitHub, OneDrive, or Google Drive).
4. Display the signed image below using your web link.

Note: The key requirement is that others must be able to compile your R Markdown file successfully without having the image file locally. This will be the standard procedure for homework assignments and projects throughout the course.

Exercise 2

Recreate the content on page 47 of the textbook, starting from “If the errors w_t are normally distributed …” through the end of the page. Exact typesetting is not required; however, all mathematical notation and formatting must be correct.

If the errors w_t are normally distributed, $\hat{\beta}$ is also the maximum likelihood estimator for β and is normally distributed with

$$\text{cov}(\hat{\beta}) = \sigma_w^2 C, \tag{2.6}$$

where

$$C = \left(\sum_{t=1}^{n} z_t z_t' \right)^{-1} \tag{2.7}$$

is a convenient notation. An unbiased estimator for the variance σ_w^2 is

$$s_w^2 = MSE = \frac{SSE}{n - (q+1)} \tag{2.8}$$

where MSE denotes the . Under normal assumption,

$$t = \frac{(\hat{\beta}_t - \beta_t)}{s_w \sqrt{c_{tt}}} \tag{2.8}$$

Exercise 3

In this exercise, we explore how to load an external dataset into R.

1. Download daily S&P 500 index closing-price data from 06/01/2025 to 06/30/2025 as a CSV file (e.g., from this link (https://www.marketwatch.com/investing/index/spx/download-data)).
2. Upload the CSV file to a publicly accessible web location.
3. Create a code chunk below that loads the dataset from your web link into R and displays the first few rows using the `head()` function.

```
data_url<-"https://raw.githubusercontent.com/popsicle2003/Applied-Time-Series-Coursework/refs/heads/main/Exercises/S%26P500%20Close%20Data.csv"
sp500_data <- read.csv(data_url)

head(sp500_data)
```

	Date <chr>	Open <chr>	High <chr>	Low <chr>	Close <chr>
1	06/27/2025	6,150.70	6,187.68	6,132.35	6,173.07
2	06/26/2025	6,112.09	6,146.52	6,107.27	6,141.02
3	06/25/2025	6,104.23	6,108.51	6,080.09	6,092.16
4	06/24/2025	6,061.21	6,101.76	6,059.25	6,092.18
5	06/23/2025	5,969.67	6,028.77	5,943.23	6,025.17
6	06/20/2025	5,999.67	6,018.20	5,952.56	5,967.84

6 rows

Exercise 4

In the lecture we saw the function

```
crazy <- function(num)
{
  if (num>10000)
  {
    print('Sample size too large. Set to 10000')
    num <- 10000.
  }
 x <- c() # start with an empty vector
 for (n in 1:num)
 {
   x[n] <- mean(rcauchy(n))  #rcauchy(n) generates n standard Cauchy random numbers
 }
 plot(x, type="l", xlab="sample size", ylab="sample mean")
}
```

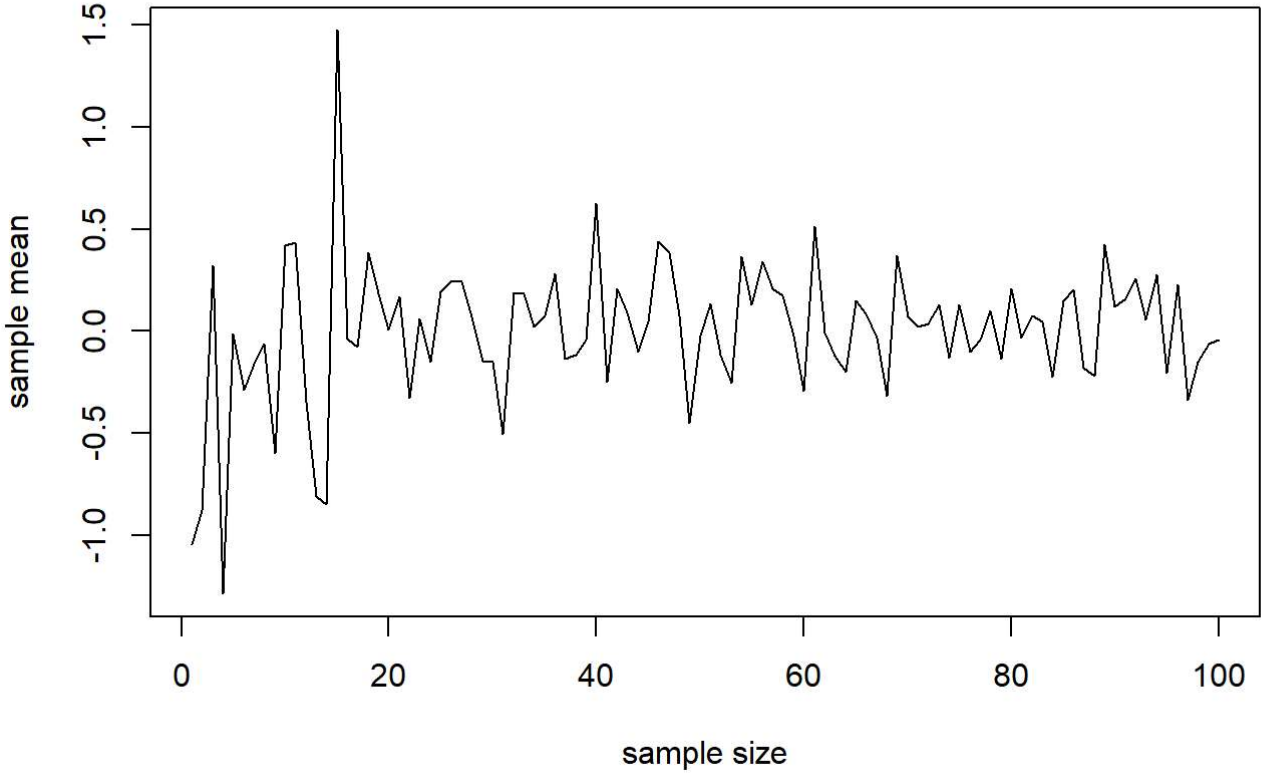
which is used to visually illustrate the failure of Law of Large Numbers when the population mean does not exist. What if we replace the sample mean by sample median? To explore what happens, let's try the following.

1. Create a function called "cool" that does a similar job as the "crazy" function above, which replaces the mean by median. In addition, you are required to achieve the following: if the sample size is less than 100, replace it by 100 and display the message "Sample size too small. Set to 100."
2. Run your function 3 times with the inputs: `cool(100)` , `cool(1000)` , `cool(10000)` and display the results. Does a Law of Large Numbers seem hold for median?

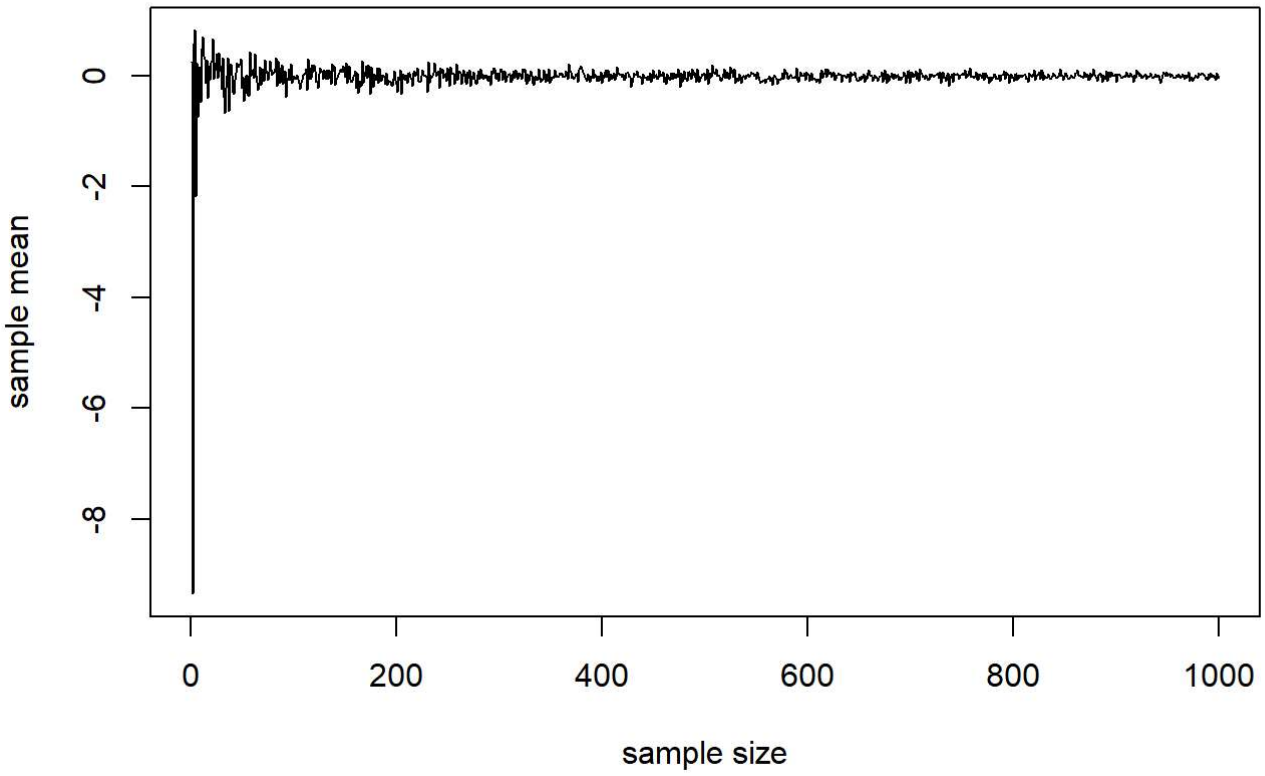
```
cool <- function(num)
{
  if (num>10000)
  {
    print('Sample size too large. Set to 10000')
    num <- 10000.
  }
  if (num<100)
  {
    print('Sample size too small. Set to 100')
    num <- 100.
  }
 x <- c() # start with an empty vector
 for (n in 1:num)
 {
   x[n] <- median(rcauchy(n))  #rcauchy(n) generates n standard Cauchy random numbers
 }
 plot(x, type="l", xlab="sample size", ylab="sample mean")
}
```

Testing

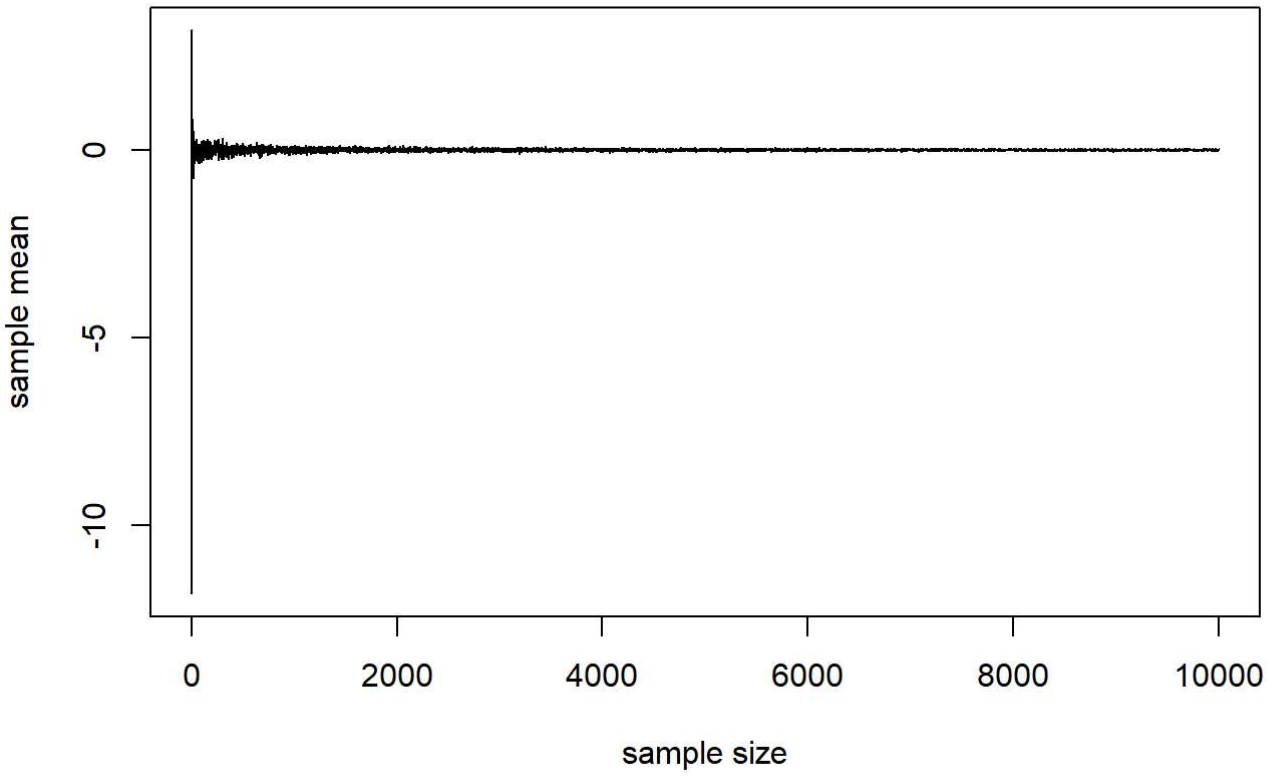
```
cool(100)
```




```
cool(1000)
```




```
cool(10000)
```



Yes.

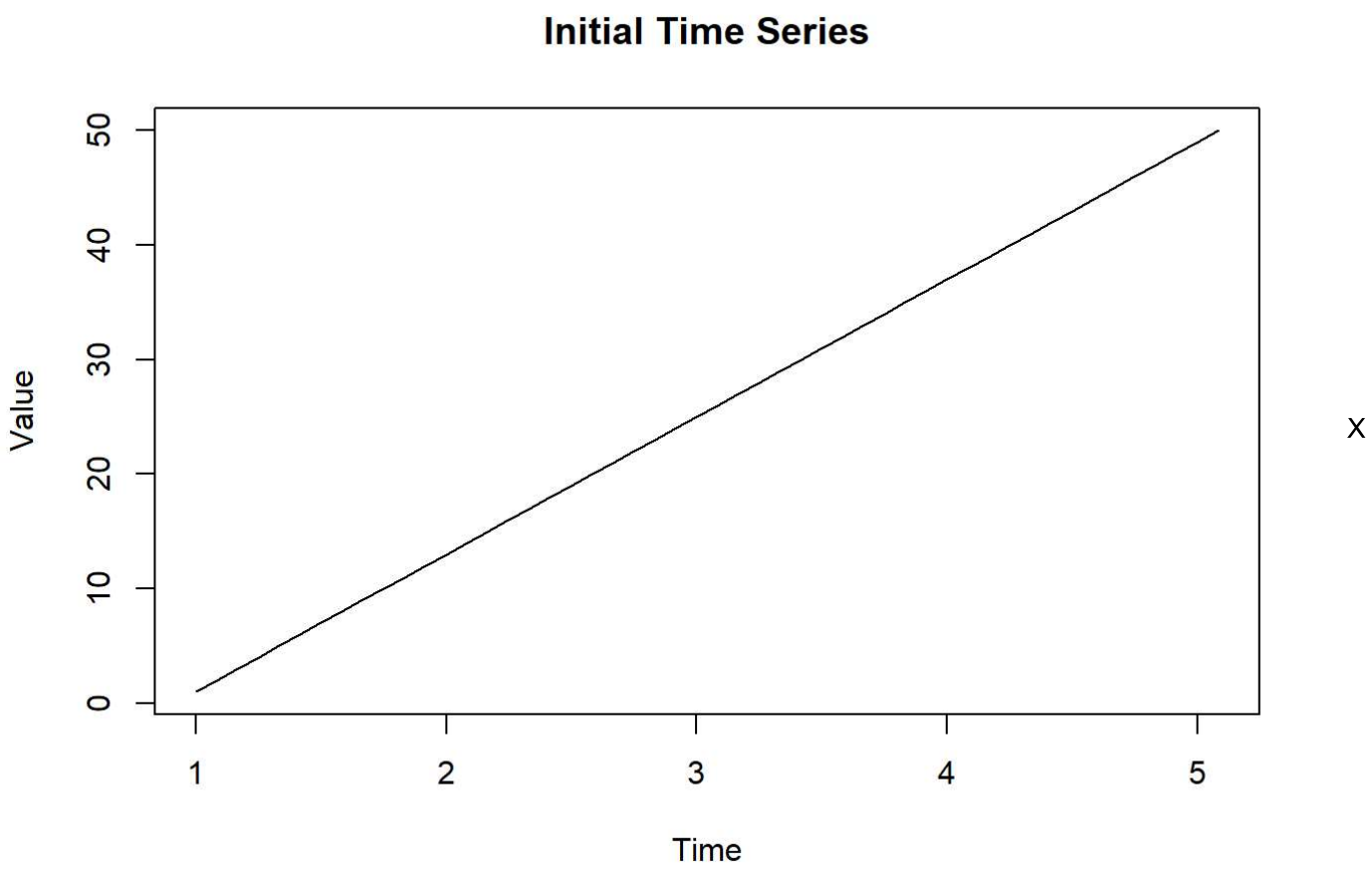
We see that law of large numbers also applies to median. # Exercise 5

1. Create a *ts* object containing consecutive integers from 1 to 50, whose `frequency` is set to 12.
2. Display and plot the data. In the plot, what do the integers in the x-axis tick labels stand for?
3. Modify the attributes of the data, so that it represents a monthly time series starting in June 1990. Display the modified data.
4. Based on the previous item, extract the sub time series from December 1990 to April 1992.

```
my_ts <- ts(1:50, frequency = 12)
print(my_ts)
```

```
##    Jan Feb Mar Apr May Jun Jul Aug Sep Oct Nov Dec
## 1    1   2   3   4   5   6   7   8   9  10  11  12
## 2   13  14  15  16  17  18  19  20  21  22  23  24
## 3   25  26  27  28  29  30  31  32  33  34  35  36
## 4   37  38  39  40  41  42  43  44  45  46  47  48
## 5   49  50
```

```
plot(my_ts, main = "Initial Time Series", ylab = "Value", xlab = "Time")
```



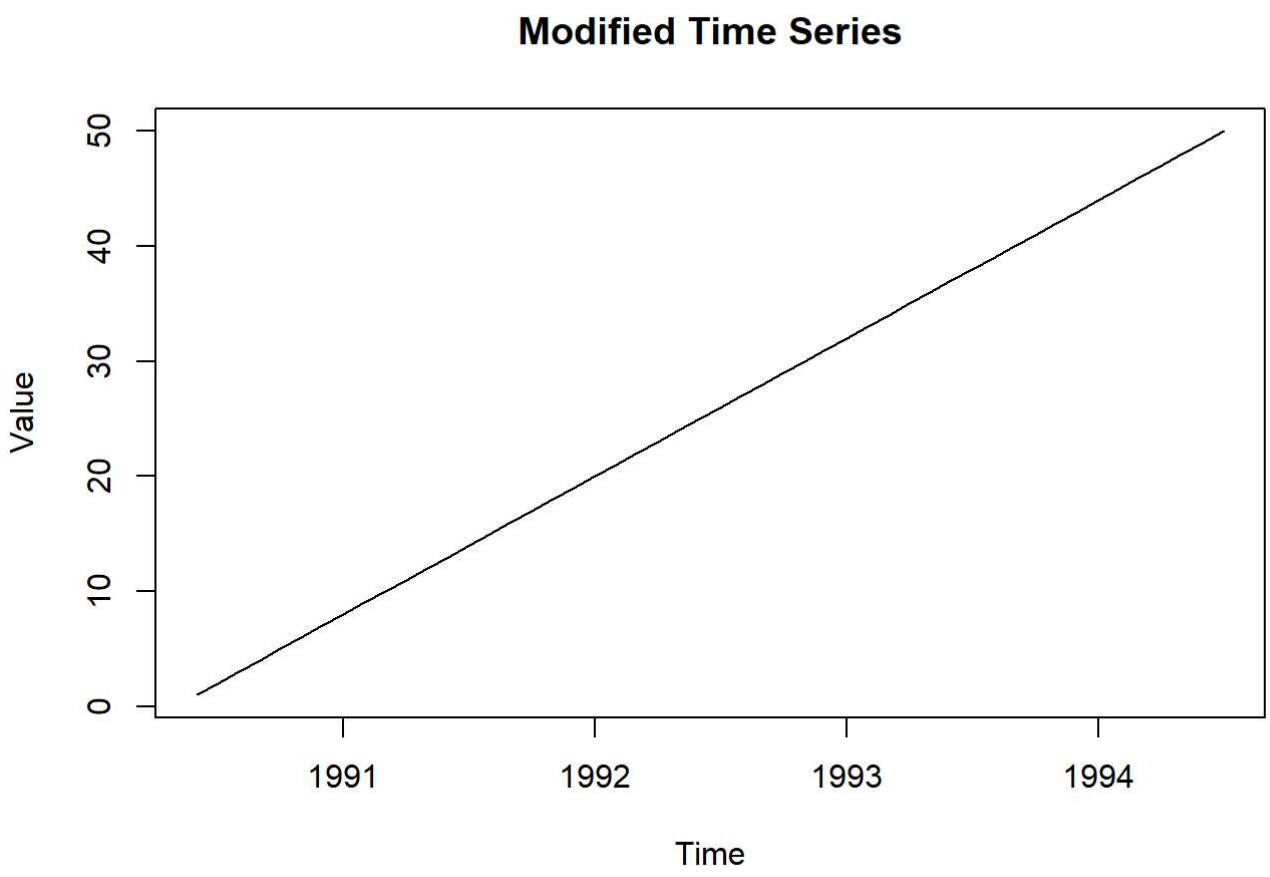
X

axis represents time units.

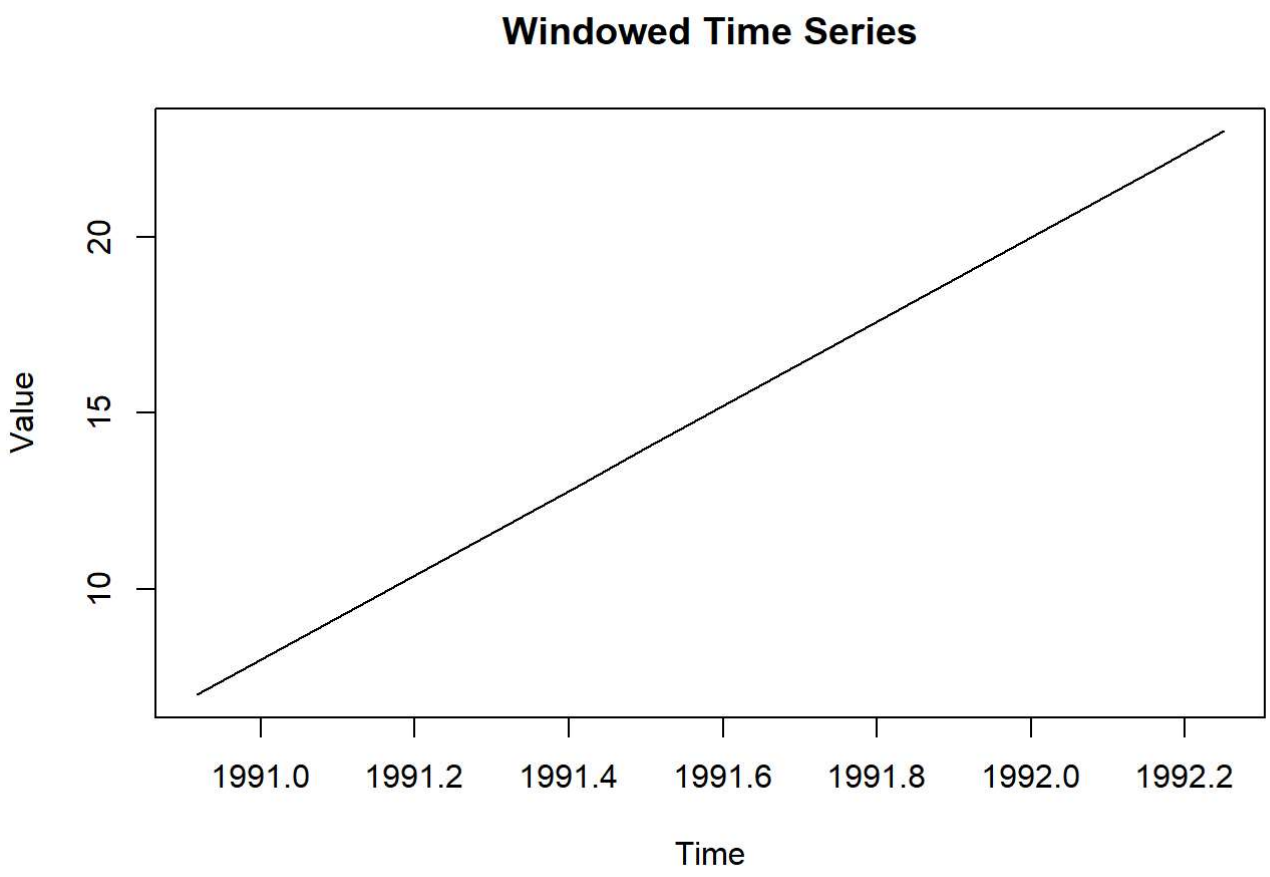
```
my_ts_modified <- ts(1:50, start = c(1990, 6), frequency = 12)
print(my_ts_modified)
```

```
##      Jan Feb Mar Apr May Jun Jul Aug Sep Oct Nov Dec
## 1990                       1   2   3   4   5   6   7
## 1991   8   9  10  11  12  13  14  15  16  17  18  19
## 1992  20  21  22  23  24  25  26  27  28  29  30  31
## 1993  32  33  34  35  36  37  38  39  40  41  42  43
## 1994  44  45  46  47  48  49  50
```

```
plot(my_ts_modified, main = "Modified Time Series", ylab = "Value", xlab = "Time")
```




```
sub_ts <- window(my_ts_modified, start = c(1990, 12), end = c(1992, 4))
plot(sub_ts, main = "Windowed Time Series", ylab = "Value", xlab = "Time")
```




```
print(sub_ts)
```

```
##      Jan Feb Mar Apr May Jun Jul Aug Sep Oct Nov Dec
## 1990                                                7
## 1991   8   9  10  11  12  13  14  15  16  17  18  19
## 1992  20  21  22  23
```