Name: Your name here
Due: 2024/09/23

Homework 2

Be sure to submit **both** the .pdf and .qmd file to Canvas by Monday, September 23rd at 11:59 pm.

0. [1 pt] Did you work with anyone on this homework? If so, who?

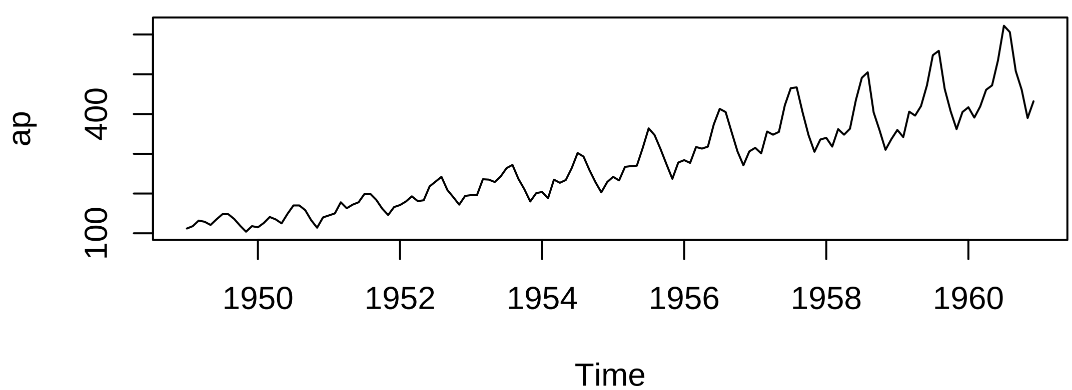
1. [13 pt] For this homework, we continue practicing decomposition of time series, again recreating the output of `decompose`, but this time focusing on a multiplicative decomposition model.

   a) [1 pt] Load the `AirPassengers` data set.

   ```
   data(AirPassengers)
   ap <- AirPassengers
   ```

   b) [1 pt] Plot the raw `AirPassengers` time series and describe what you see in terms of trend and seasonal effect.

   ```
   plot(ap)
   ```



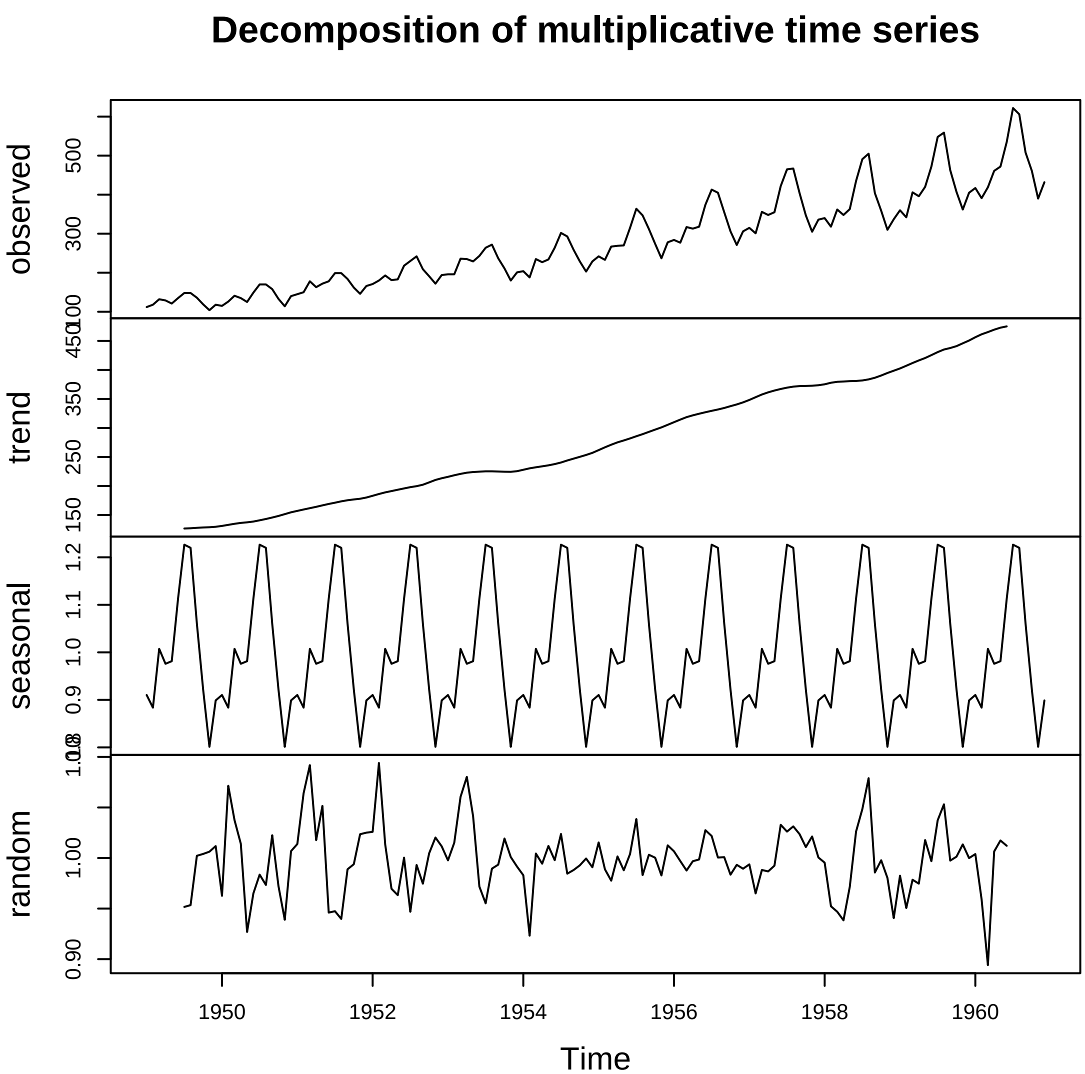
   The series has a strong positive trend and an equally strong seasonal component that increases in magnitude as the trend increases.

   c) [1 pt] Is an additive decomposition model appropriate for these data? Why or why not?

> No. The magnitude of the seasonal effect increases as the trend increases, suggesting a multiplicative model would be more appropriate.

d) [1 pt] Regardless of your answer to the previous question, fit a multiplicative decomposition model to the `AirPassengers` data set and plot the results.

```
ap_decompose <- decompose(ap, type = "multiplicative")
plot(ap_decompose)
```



e) [3 pt] Calculate the trend component for the multiplicative model by hand, and compare your estimated trend effects to those obtained by `decompose` to ensure they are correct.

```r
ap_trend <- matrix(NA, nrow = length(ap) - 12, ncol = 1)
for(t in 7:(length(ap)-6)){
  ap_trend[t-6,] <- (
    .5*ap[t-6] + ap[t-5] + ap[t-4] + ap[t-3] +
      ap[t-2] + ap[t-1] + ap[t] + ap[t+1] +
      ap[t+2] + ap[t+3] + ap[t+4] + ap[t+5] + .5*ap[t+6]
  )*(1/12)
}

# append the NAs we skipped
ap_trend_full <- c(rep(NA, 6), ap_trend, rep(NA, 6))

# check if hand values are equal, up to numeric tolerance
all(abs(ap_trend_full - ap_decompose$trend) <= 1e-10, na.rm = T)
```

```
[1] TRUE
```

f) [3 pt] Calculate the average seasonal effect associated with each month and center the resulting estimates. Compare these 12 values to those obtained from the `decompose` function to ensure you have calculated them correctly.

```r
s_hat <- ap / ap_trend_full
s_bar_tmp <- colMeans(matrix(s_hat, ncol = 12, byrow = T), na.rm = T)
s_bar <- s_bar_tmp/mean(s_bar_tmp)

# check if hand values are equal, up to numeric tolerance
all(abs(s_bar - ap_decompose$figure) <= 1e-10)
```

```
[1] TRUE
```

g) [3 pt] Calculate the residual error series from the previously created objects representing the trend and seasonal effects. Compare the by-hand calculation to the results from `decompose` to ensure that you have calculated the series correctly.

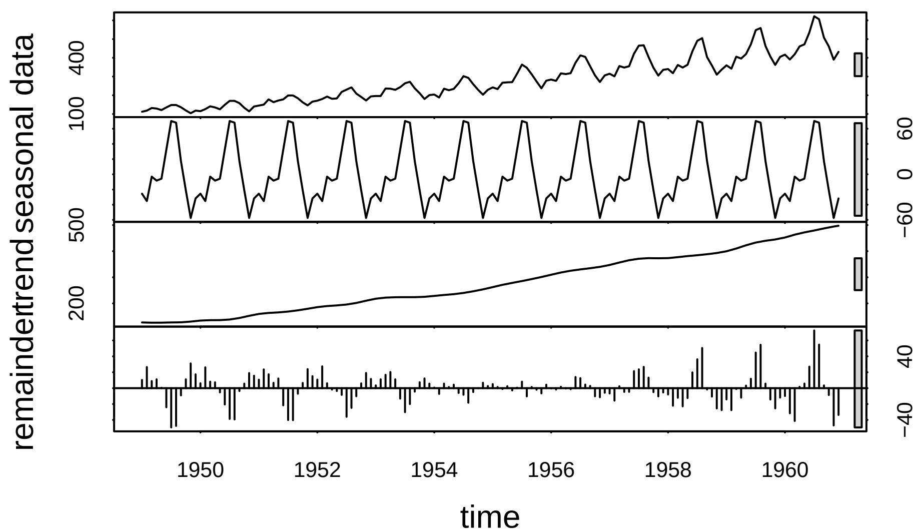
```r
# calculate residual error series
res <- ap / (ap_trend_full * s_bar)

# check if hand values are equal, up to numeric tolerance
all(abs(res - ap_decompose$random) <= 1e-10, na.rm = T)
```

```
[1] TRUE
```

2. [5 pt] The moving average calculated by the `decompose` function is a type of **smoother** or **filter**. Smoothing/filtering algorithms generally seek to predict a response at time t based on observations from both before and after t, as we saw with the moving average in question 1. Another popular smoother is the **loess** smoother, which stands for locally estimated scatterplot smoothing, and can be performed in R with the `stl` function.

   a) [2 pt] The following code fits a loess smoother to the `AirPassengers` data and prints the trend estimate. How does this compare to the trend created by `decompose`? Do you notice any major differences?

```
loess_periodic <- stl(ap, s.window = "periodic")
plot(loess_periodic)
```




```
loess_periodic$time.series[,2]
```

```
          Jan      Feb      Mar      Apr      May      Jun      Jul      Aug
1949 127.1873 126.6495 126.1117 126.1989 126.2861 126.7330 127.1799 127.4162
1950 134.3390 135.1084 135.8777 135.9454 136.0131 137.2093 138.4055 141.3114
1951 159.7945 161.6914 163.5884 164.6017 165.6150 167.2285 168.8420 171.2070
1952 185.7223 187.8163 189.9103 191.1312 192.3520 194.0658 195.7796 198.7814
1953 218.0156 220.1263 222.2371 222.9376 223.6380 223.7928 223.9476 223.9443
1954 228.7944 230.5535 232.3126 233.8328 235.3531 237.7041 240.0551 243.2465
1955 262.3979 266.7545 271.1111 274.8384 278.5657 282.0862 285.6067 289.3515
1956 309.7570 314.0295 318.3020 321.7071 325.1122 327.6514 330.1906 332.4831
1957 347.2005 351.9113 356.6220 360.9900 365.3580 368.2253 371.0926 372.2017
```

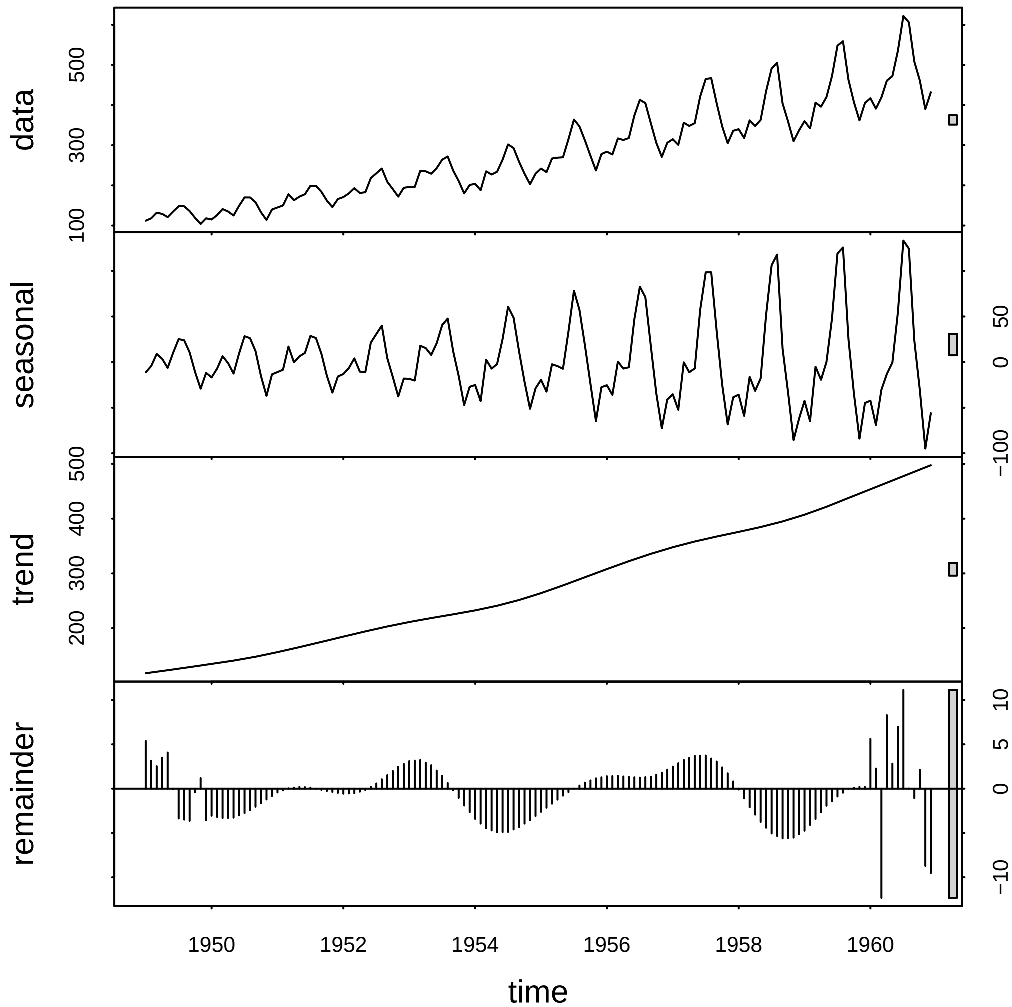
```
1958 373.5057 375.3037 377.1017 379.1766 381.2514 382.8734 384.4954 386.4129
1959 399.6645 404.9035 410.1425 416.3450 422.5474 427.7897 433.0319 436.4362
1960 452.3700 457.8121 463.2543 467.6104 471.9665 475.6083 479.2501 483.1894
          Sep      Oct      Nov      Dec
1949 127.6525 129.0186 130.3846 132.3618
1950 144.2173 148.3371 152.4568 156.1257
1951 173.5721 176.5526 179.5332 182.6278
1952 201.7832 206.0687 210.3542 214.1849
1953 223.9409 224.5947 225.2486 227.0215
1954 246.4380 250.1037 253.7693 258.0836
1955 293.0963 297.0760 301.0557 305.4064
1956 334.7756 337.3493 339.9230 343.5618
1957 373.3108 373.0689 372.8270 373.1663
1958 388.3304 390.6746 393.0187 396.3416
1959 439.8406 442.3755 444.9104 448.6402
1960 487.1288 490.7530 494.3773 497.4299
```

> The values differ by a little, and you are able to obtain trend estimates for the first and last 6 months.

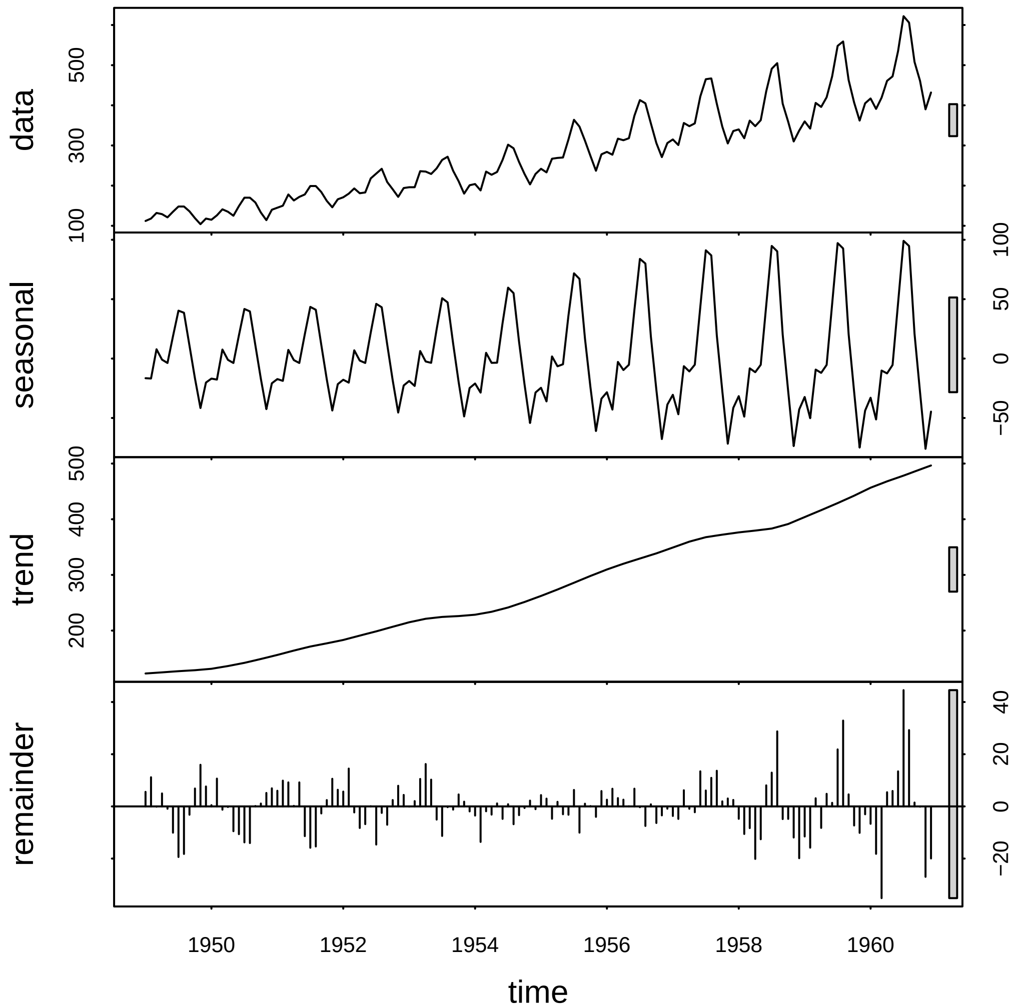
b) [2 pt] The code above assumes the window for calculating the loess smoother is based on the period of the time series. Play around with changing the `s.window` argument (which can be an integer) in the `stl` function. What do you notice about the resulting decomposition?

> Changing the values can have a fairly significant impact on the magnitude of the estimate of the seasonal component.

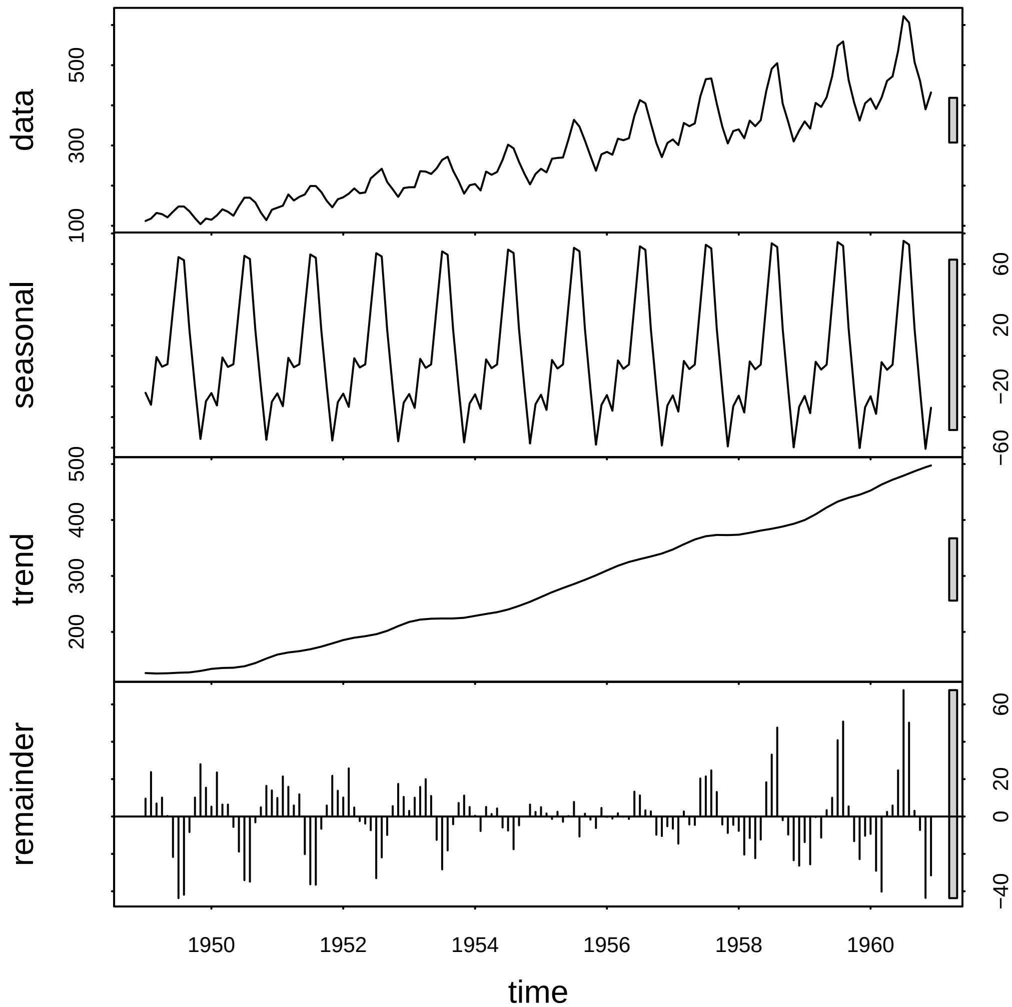
```
plot(stl(ap, s.window = 3))
```




```r
plot(stl(ap, s.window = 10))
```




```
plot(stl(ap, s.window = 30))
```



c) [1 pt] The smoothing and filtering algorithms provide a nice way to summarize a time series in retrospect. Can you think of a shortcoming for using these tools to estimate time series and create forecasts?

> You cannot create forecasts with them! They rely on data from time points both before and after the point of prediction, meaning you cannot forecast into the future with them. Generally, they do not provide a model that you can write down to perform forecasts.

3. [6 pt] Find an interesting **discrete** time series data set and create a decomposition of it below. Note that Kaggle and Data.world often provide some nicely manicured, free data.