Mid Trimester

Nzambuli Daniel

2024-06-19

The Data

```
library(readxl)
Mid_sem_dataset = read_excel("Mid-sem dataset.xlsx")
Mid_sem_dataset
```

```
## # A tibble: 60 × 4
##    Season Month      Year No_of_tourists
##    <chr>  <chr>     <dbl>          <dbl>
##  1 Summer January    2019          19500
##  2 Summer February   2019          21000
##  3 Autumn March      2019          23000
##  4 Autumn April      2019          25000
##  5 Autumn May        2019          27000
##  6 Winter June       2019          30000
##  7 Winter July       2019          32000
##  8 Winter August     2019          31000
##  9 Spring September  2019          29000
## 10 Spring October    2019          27000
## # ℹ 50 more rows
```

CASE SCENARIO: TOURIST ANALYSIS IN CAPETOWN

Cape Town, a scenic destination celebrated for its breathtaking landscapes and rich cultural heritage, attracts a diverse group of tourists throughout the year. Its charming old town, scenic hiking trails, vibrant local markets, and tranquil beaches offer visitors a wide array of activities. The attached hypothetical data provides insights into the number of tourists visiting Cape Town from 2019 to 2023.

1. Determine the seasonal variation in tourist numbers for each using the ratio-to-moving average approach from 2019 to 2023.

Quarterly seasonal data only

```
seasonal_data = Mid_sem_dataset[, c(1, 3, 4)]
seasonal_data
```

```
## # A tibble: 60 × 3
##    Season  Year No_of_tourists
##    <chr>  <dbl>          <dbl>
##  1 Summer  2019          19500
##  2 Summer  2019          21000
##  3 Autumn  2019          23000
```

```
##  4 Autumn  2019          25000
##  5 Autumn  2019          27000
##  6 Winter  2019          30000
##  7 Winter  2019          32000
##  8 Winter  2019          31000
##  9 Spring  2019          29000
## 10 Spring  2019          27000
## # ℹ 50 more rows
```

Group by seasons in each year

All the summers in 2019 2020 2022 2023 will be under the label summer …

```
library(tidyverse)
```

```
## ── Attaching core tidyverse packages ──────────────────────── tidyverse 2.0.0 ──
## ✔ dplyr     1.1.4     ✔ readr     2.1.5
## ✔ forcats   1.0.0     ✔ stringr   1.5.1
## ✔ ggplot2   3.5.1     ✔ tibble    3.2.1
## ✔ lubridate 1.9.3     ✔ tidyr     1.3.1
## ✔ purrr     1.0.2
## ── Conflicts ────────────────────────────────────────── tidyverse_conflicts() ──
## ✖ dplyr::filter() masks stats::filter()
## ✖ dplyr::lag()    masks stats::lag()
## ℹ Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to beco
me errors
```

```
seasonal_data = seasonal_data %>% group_by(Year, Season) %>% summarise(No_of_tourists = sum(No_
of_tourists))
```

```
## `summarise()` has grouped output by 'Year'. You can override using the
## `.groups` argument.
```

```
seasonal_data
```

```
## # A tibble: 20 × 3
## # Groups:   Year [5]
##     Year Season No_of_tourists
##    <dbl> <chr>           <dbl>
##  1  2019 Autumn          75000
##  2  2019 Spring          81000
##  3  2019 Summer          63500
##  4  2019 Winter          93000
##  5  2020 Autumn          84000
##  6  2020 Spring          87000
##  7  2020 Summer          71000
##  8  2020 Winter          99000
##  9  2021 Autumn          93000
## 10  2021 Spring          96000
## 11  2021 Summer          79500
## 12  2021 Winter         108000
## 13  2022 Autumn         108000
## 14  2022 Spring         111000
## 15  2022 Summer          95000
## 16  2022 Winter         123000
## 17  2023 Autumn         123000
## 18  2023 Spring         103000
```

```
## 19  2023 Summer         102000
## 20  2023 Winter         120000
```

Reorder the column of season to appear like the original

target order is `Summer, Autumn, Winter, Spring`

```
seasonal_data$Season = factor(seasonal_data$Season, levels = c("Summer", "Autumn", "Winter", "Spring"))
seasonal_data = seasonal_data[order(seasonal_data$Year, seasonal_data$Season),]
seasonal_data
```

```
## # A tibble: 20 × 3
## # Groups:   Year [5]
##     Year Season No_of_tourists
##    <dbl> <fct>           <dbl>
##  1  2019 Summer          63500
##  2  2019 Autumn          75000
##  3  2019 Winter          93000
##  4  2019 Spring          81000
##  5  2020 Summer          71000
##  6  2020 Autumn          84000
##  7  2020 Winter          99000
##  8  2020 Spring          87000
##  9  2021 Summer          79500
## 10  2021 Autumn          93000
## 11  2021 Winter         108000
## 12  2021 Spring          96000
## 13  2022 Summer          95000
## 14  2022 Autumn         108000
## 15  2022 Winter         123000
## 16  2022 Spring         111000
## 17  2023 Summer         102000
## 18  2023 Autumn         123000
## 19  2023 Winter         120000
## 20  2023 Spring         103000
```

Moving average

$$MA_t = \frac{1}{no.\ seasons} \sum_{i=1}^{t+3} x_i$$

$$= \frac{1}{4} \sum_{i=1}^{t+3} x_i$$

```
mov_avg = zoo::rollmean(seasonal_data$No_of_tourists, k = 4, fill = NA, align = "center")
mov_avg
```

```
##  [1]     NA  78125  80000  82250  83750  85250  87375  89625  91875  94125
## [11]  98000 101750 105500 109250 111000 114750 114000 112000     NA     NA
```

```
seasonal_data$mov_avg = mov_avg
seasonal_data
```

```
## # A tibble: 20 × 4
## # Groups:   Year [5]
```

```
##     Year Season No_of_tourists mov_avg
##    <dbl> <fct>           <dbl>   <dbl>
##  1  2019 Summer          63500      NA
##  2  2019 Autumn          75000   78125
##  3  2019 Winter          93000   80000
##  4  2019 Spring          81000   82250
##  5  2020 Summer          71000   83750
##  6  2020 Autumn          84000   85250
##  7  2020 Winter          99000   87375
##  8  2020 Spring          87000   89625
##  9  2021 Summer          79500   91875
## 10  2021 Autumn          93000   94125
## 11  2021 Winter         108000   98000
## 12  2021 Spring          96000  101750
## 13  2022 Summer          95000  105500
## 14  2022 Autumn         108000  109250
## 15  2022 Winter         123000  111000
## 16  2022 Spring         111000  114750
## 17  2023 Summer         102000  114000
## 18  2023 Autumn         123000  112000
## 19  2023 Winter         120000      NA
## 20  2023 Spring         103000      NA
```

```
rat_mn = zoo::rollmean(seasonal_data$mov_avg, k = 2, fill = NA, align = "center")
rat_mn = c(NA, rat_mn[1:19])
rat_mn
```

```
##  [1]       NA       NA  79062.5  81125.0  83000.0  84500.0  86312.5  88500.0
##  [9]  90750.0  93000.0  96062.5  99875.0 103625.0 107375.0 110125.0 112875.0
## [17] 114375.0 113000.0       NA       NA
```

Centered Moving Average

$$CMA_t = \frac{1}{2}(x_t + x_{t+1})$$

```
seasonal_data$rat_mn = rat_mn
seasonal_data
```

```
## # A tibble: 20 × 5
## # Groups:   Year [5]
##     Year Season No_of_tourists mov_avg  rat_mn
##    <dbl> <fct>           <dbl>   <dbl>   <dbl>
##  1  2019 Summer          63500      NA     NA
##  2  2019 Autumn          75000   78125     NA
##  3  2019 Winter          93000   80000  79062.
##  4  2019 Spring          81000   82250  81125
##  5  2020 Summer          71000   83750  83000
##  6  2020 Autumn          84000   85250  84500
##  7  2020 Winter          99000   87375  86312.
##  8  2020 Spring          87000   89625  88500
##  9  2021 Summer          79500   91875  90750
## 10  2021 Autumn          93000   94125  93000
## 11  2021 Winter         108000   98000  96062.
## 12  2021 Spring          96000  101750  99875
## 13  2022 Summer          95000  105500 103625
## 14  2022 Autumn         108000  109250 107375
## 15  2022 Winter         123000  111000 110125
```

```
## 16  2022 Spring         111000  114750 112875
## 17  2023 Summer         102000  114000 114375
## 18  2023 Autumn         123000  112000 113000
## 19  2023 Winter         120000      NA     NA
## 20  2023 Spring         103000      NA     NA
```

Seasonal Ratio

$$seasonal\ relatives = \frac{actual}{centralized\ avg} * 100$$

```
seasonal_data$sn_rel = seasonal_data$No_of_tourists/ seasonal_data$rat_mn * 100
seasonal_data
```

```
## # A tibble: 20 × 6
## # Groups:   Year [5]
##     Year Season No_of_tourists mov_avg  rat_mn sn_rel
##    <dbl> <fct>           <dbl>   <dbl>   <dbl>  <dbl>
##  1  2019 Summer          63500      NA     NA    NA
##  2  2019 Autumn          75000   78125     NA    NA
##  3  2019 Winter          93000   80000  79062.  118.
##  4  2019 Spring          81000   82250  81125    99.8
##  5  2020 Summer          71000   83750  83000    85.5
##  6  2020 Autumn          84000   85250  84500    99.4
##  7  2020 Winter          99000   87375  86312.  115.
##  8  2020 Spring          87000   89625  88500    98.3
##  9  2021 Summer          79500   91875  90750    87.6
## 10  2021 Autumn          93000   94125  93000   100
## 11  2021 Winter         108000   98000  96062.  112.
## 12  2021 Spring          96000  101750  99875    96.1
## 13  2022 Summer          95000  105500 103625    91.7
## 14  2022 Autumn         108000  109250 107375   101.
## 15  2022 Winter         123000  111000 110125   112.
## 16  2022 Spring         111000  114750 112875    98.3
## 17  2023 Summer         102000  114000 114375    89.2
## 18  2023 Autumn         123000  112000 113000   109.
## 19  2023 Winter         120000      NA     NA    NA
## 20  2023 Spring         103000      NA     NA    NA
```

Pivot the seasonal indexes wider

```
seas_wide = pivot_wider(seasonal_data[, c(1, 2, 6)], names_from = Year, values_from = sn_rel)
seas_wide
```

```
## # A tibble: 4 × 6
##   Season `2019` `2020` `2021` `2022` `2023`
##   <fct>   <dbl>  <dbl>  <dbl>  <dbl>  <dbl>
## 1 Summer   NA     85.5   87.6   91.7   89.2
## 2 Autumn   NA     99.4  100    101.   109.
## 3 Winter  118.   115.   112.   112.    NA
## 4 Spring   99.8   98.3   96.1   98.3   NA
```

find the median value

```
seas_wide$median = apply(seas_wide[,2:6], 1, function(x) median(x, na.rm = T))
seas_wide
```

```
## # A tibble: 4 × 7
##   Season `2019` `2020` `2021` `2022` `2023` median
##   <fct>   <dbl>  <dbl>  <dbl>  <dbl>  <dbl>  <dbl>
## 1 Summer   NA     85.5   87.6   91.7   89.2   88.4
## 2 Autumn   NA     99.4  100    101.   109.   100.
## 3 Winter  118.   115.   112.   112.    NA    114.
## 4 Spring   99.8   98.3   96.1   98.3   NA     98.3
```

Adjusted Seasonal indexes

```
tot_seas_median = mean(seas_wide$median)
seas_wide$seas_ind = seas_wide$median / tot_seas_median * 100
seas_wide
```

```
## # A tibble: 4 × 8
##   Season `2019` `2020` `2021` `2022` `2023` median seas_ind
##   <fct>   <dbl>  <dbl>  <dbl>  <dbl>  <dbl>  <dbl>    <dbl>
## 1 Summer   NA     85.5   87.6   91.7   89.2   88.4     88.3
## 2 Autumn   NA     99.4  100    101.   109.   100.     100.
## 3 Winter  118.   115.   112.   112.    NA    114.     113.
## 4 Spring   99.8   98.3   96.1   98.3   NA     98.3     98.2
```

Monthly seasonal data

Combine the initial code to a function

```
present_month = function(data, x, season_size){
  mov_avg = zoo::rollmean(data[[x]], k = season_size, fill = NA, align = "center")
  print("The moving average is:")
  print(mov_avg)
  rat_mn = zoo::rollmean(mov_avg, k = 2, fill = NA, align = "center")
  rat_mn = c(NA, rat_mn[1:(length(rat_mn) -1)])
  print("The centralized moving average is:")
  print(rat_mn)
  sn_rel = month_data$No_of_tourists/ rat_mn * 100
  print("The seasonal relative is:")
  print(sn_rel)
  long = data.frame(Year = data[,2],
                    Season = data[, 1],
                    sn_rel = sn_rel)
  wide = pivot_wider(long, names_from = Year, values_from = sn_rel)
  print("The data in wide format is:")
  print(wide)
  wide$median = apply(wide[,2:ncol(wide)], 1, function(x) median(x, na.rm = T))
  tot_seas_median = mean(wide$median)
  wide$seas_ind = wide$median / tot_seas_median * 100
  print("With seasonal index")
  return(wide)
}
```

The annual data

```
month_data = Mid_sem_dataset[, c(2, 3, 4)]
colnames(month_data)
```

```
## [1] "Month"          "Year"           "No_of_tourists"
```

```
month_wide = present_month(month_data,"No_of_tourists", 12)
```

```
## [1] "The moving average is:"
##  [1]       NA       NA       NA       NA       NA 26041.67 26250.00 26500.00
##  [9] 26750.00 27000.00 27250.00 27416.67 27583.33 27750.00 27916.67 28083.33
## [17] 28250.00 28416.67 28625.00 28875.00 29125.00 29375.00 29625.00 29875.00
## [25] 30125.00 30375.00 30625.00 30875.00 31125.00 31375.00 31833.33 32250.00
## [33] 32666.67 33083.33 33500.00 33916.67 34333.33 34750.00 35166.67 35583.33
## [41] 36000.00 36416.67 36833.33 37250.00 37666.67 38083.33 38500.00 38333.33
## [49] 38333.33 38250.00 38083.33 37833.33 37583.33 37333.33       NA       NA
## [57]       NA       NA       NA       NA
## [1] "The centralized moving average is:"
##  [1]       NA       NA       NA       NA       NA       NA 26145.83 26375.00
##  [9] 26625.00 26875.00 27125.00 27333.33 27500.00 27666.67 27833.33 28000.00
## [17] 28166.67 28333.33 28520.83 28750.00 29000.00 29250.00 29500.00 29750.00
## [25] 30000.00 30250.00 30500.00 30750.00 31000.00 31250.00 31604.17 32041.67
## [33] 32458.33 32875.00 33291.67 33708.33 34125.00 34541.67 34958.33 35375.00
## [41] 35791.67 36208.33 36625.00 37041.67 37458.33 37875.00 38291.67 38416.67
## [49] 38333.33 38291.67 38166.67 37958.33 37708.33 37458.33       NA       NA
## [57]       NA       NA       NA       NA
## [1] "The seasonal relative is:"
##  [1]        NA        NA        NA        NA        NA        NA 122.39044
##  [8] 117.53555 108.92019 100.46512  92.16590  84.14634  80.00000  86.74699
## [15]  93.41317 100.00000 106.50888 112.94118 119.21110 114.78261 106.89655
## [22]  99.14530  91.52542  84.03361  81.66667  89.25620  95.08197 100.81301
## [29] 106.45161 112.00000 117.07317 112.35371 104.74968  97.33840  90.11264
## [36]  83.06551  87.91209  92.64174  97.25864 101.76678 106.16997 110.47181
## [43] 114.67577 110.68616 104.11568  97.68977  91.40370  85.90022  91.30435
## [50]  96.62677 102.18341 108.01317 114.03315 101.44605        NA        NA
## [57]        NA        NA        NA        NA
## [1] "The data in wide format is:"
## # A tibble: 12 × 6
##    Month     `2019` `2020` `2021` `2022` `2023`
##    <chr>      <dbl>  <dbl>  <dbl>  <dbl>  <dbl>
##  1 January     NA     80     81.7   87.9   91.3
##  2 February    NA     86.7   89.3   92.6   96.6
##  3 March       NA     93.4   95.1   97.3  102.
##  4 April       NA    100    101.   102.   108.
##  5 May         NA    107.   106.   106.   114.
##  6 June        NA    113.   112    110.   101.
##  7 July       122.   119.   117.   115.    NA
##  8 August     118.   115.   112.   111.    NA
##  9 September  109.   107.   105.   104.    NA
## 10 October    100.    99.1   97.3   97.7   NA
## 11 November    92.2   91.5   90.1   91.4   NA
## 12 December    84.1   84.0   83.1   85.9   NA
## [1] "With seasonal index"
```

```
month_wide
```

```
## # A tibble: 12 × 8
##    Month     `2019` `2020` `2021` `2022` `2023` median seas_ind
##    <chr>      <dbl>  <dbl>  <dbl>  <dbl>  <dbl>  <dbl>    <dbl>
##  1 January     NA     80     81.7   87.9   91.3   84.8     84.6
##  2 February    NA     86.7   89.3   92.6   96.6   90.9     90.8
##  3 March       NA     93.4   95.1   97.3  102.    96.2     96.0
##  4 April       NA    100    101.   102.   108.   101.     101.
```

```
##  5 May         NA    107.   106.   106.   114.   106.     106.
##  6 June        NA    113.   112    110.   101.   111.     111.
##  7 July       122.   119.   117.   115.    NA    118.     118.
##  8 August     118.   115.   112.   111.    NA    114.     113.
##  9 September  109.   107.   105.   104.    NA    106.     106.
## 10 October    100.    99.1   97.3   97.7   NA     98.4     98.2
## 11 November    92.2   91.5   90.1   91.4   NA     91.5     91.3
## 12 December    84.1   84.0   83.1   85.9   NA     84.1     83.9
```

2. Obtain a clearer view of the underlying trend by eliminating seasonal fluctuations.

Monthly Data

```
monthly_smoothed_data = data.frame(month = rep(month_wide$Month, 5),
                                   year = rep(seq(2019, 2023), 12),
                                   tourists = month_data$No_of_tourists,
                                   adj_seas_ind = rep(month_wide$seas_ind, 5))

monthly_smoothed_data$deseason = monthly_smoothed_data$tourists/ monthly_smoothed_data$adj_seas
_ind * 100
monthly_smoothed_data
```

```
##        month year tourists adj_seas_ind deseason
## 1    January 2019    19500     84.61872 23044.55
## 2   February 2020    21000     90.76591 23136.44
## 3      March 2021    23000     95.97674 23964.14
## 4      April 2022    25000    101.08602 24731.41
## 5        May 2023    27000    106.26593 25407.96
## 6       June 2019    30000    111.01201 27024.10
## 7       July 2020    32000    117.90435 27140.65
## 8     August 2021    31000    113.33957 27351.44
## 9  September 2022    29000    105.61012 27459.49
## 10   October 2023    27000     98.21944 27489.47
## 11  November 2019    25000     91.28047 27388.12
## 12  December 2020    23000     83.92073 27406.82
## 13   January 2021    22000     84.61872 25998.98
## 14  February 2022    24000     90.76591 26441.65
## 15     March 2023    26000     95.97674 27089.90
## 16     April 2019    28000    101.08602 27699.18
## 17       May 2020    30000    106.26593 28231.06
## 18      June 2021    32000    111.01201 28825.71
## 19      July 2022    34000    117.90435 28836.94
## 20    August 2023    33000    113.33957 29116.04
## 21 September 2019    31000    105.61012 29353.25
## 22   October 2020    29000     98.21944 29525.72
## 23  November 2021    27000     91.28047 29579.17
## 24  December 2022    25000     83.92073 29790.02
## 25   January 2023    24500     84.61872 28953.41
## 26  February 2019    27000     90.76591 29746.85
## 27     March 2020    29000     95.97674 30215.66
## 28     April 2021    31000    101.08602 30666.95
## 29       May 2022    33000    106.26593 31054.17
## 30      June 2023    35000    111.01201 31528.12
## 31      July 2019    37000    117.90435 31381.37
## 32    August 2020    36000    113.33957 31762.96
## 33 September 2021    34000    105.61012 32193.88
```

```
## 34   October 2022    32000     98.21944 32580.11
## 35  November 2023    30000     91.28047 32865.74
## 36  December 2019    28000     83.92073 33364.82
## 37   January 2020    30000     84.61872 35453.15
## 38  February 2021    32000     90.76591 35255.53
## 39     March 2022    34000     95.97674 35425.25
## 40     April 2023    36000    101.08602 35613.23
## 41       May 2019    38000    106.26593 35759.35
## 42      June 2020    40000    111.01201 36032.14
## 43      July 2021    42000    117.90435 35622.10
## 44    August 2022    41000    113.33957 36174.48
## 45 September 2023    39000    105.61012 36928.28
## 46   October 2019    37000     98.21944 37670.75
## 47  November 2020    35000     91.28047 38343.36
## 48  December 2021    33000     83.92073 39322.83
## 49   January 2022    35000     84.61872 41362.01
## 50  February 2023    37000     90.76591 40764.20
## 51     March 2019    39000     95.97674 40634.85
## 52     April 2020    41000    101.08602 40559.51
## 53       May 2021    43000    106.26593 40464.52
## 54      June 2022    38000    111.01201 34230.53
## 55      July 2023    42000    117.90435 35622.10
## 56    August 2019    40000    113.33957 35292.17
## 57 September 2020    37000    105.61012 35034.52
## 58   October 2021    34000     98.21944 34616.36
## 59  November 2022    32000     91.28047 35056.79
## 60  December 2023    30000     83.92073 35748.02
```

Quarterly data

```
quarter_smoothed_data = data.frame(quarter = rep(seas_wide$Season, 5),
                                   year = rep(seq(2019, 2023), 4),
                                   adj_seas_ind = rep(seas_wide$seas_ind, 5))
quarter_smoothed_data = quarter_smoothed_data[order(quarter_smoothed_data$year),]
quarter_smoothed_data$tourist = seasonal_data$No_of_tourists


quarter_smoothed_data$deseasoned = quarter_smoothed_data$tourist/quarter_smoothed_data$adj_seas
_ind * 100

rownames(quarter_smoothed_data) = NULL
quarter_smoothed_data
```

```
##    quarter year adj_seas_ind tourist deseasoned
## 1   Summer 2019     88.26648   63500   71941.24
## 2   Autumn 2019    100.14883   75000   74888.54
## 3   Winter 2019    113.40212   93000   82009.05
## 4   Spring 2019     98.18256   81000   82499.37
## 5   Autumn 2020    100.14883   71000   70894.49
## 6   Winter 2020    113.40212   84000   74072.69
## 7   Spring 2020     98.18256   99000  100832.57
## 8   Summer 2020     88.26648   87000   98565.16
## 9   Winter 2021    113.40212   79500   70104.51
## 10  Spring 2021     98.18256   93000   94721.50
## 11  Summer 2021     88.26648  108000  122356.75
## 12  Autumn 2021    100.14883   96000   95857.34
## 13  Spring 2022     98.18256   95000   96758.52
## 14  Summer 2022     88.26648  108000  122356.75
```

```
## 15  Autumn 2022    100.14883  123000  122817.21
## 16  Winter 2022    113.40212  111000   97881.76
## 17  Summer 2023     88.26648  102000  115559.15
## 18  Autumn 2023    100.14883  123000  122817.21
## 19  Winter 2023    113.40212  120000  105818.12
## 20  Spring 2023     98.18256  103000  104906.61
```

3. Compute the long-term trend in tourist numbers over the specified period by applying the suitable trend analysis to the deseasonalized data.

if a linear model is to be used

$$trend\ line = \beta_0 + \beta_1 t$$

otherwise for quadratic

$$trend\ line = \beta_0 + \beta_1 t + \beta_2 t^2$$

and for exponential

$$trend\ line = \beta_0 + e^{\beta_1 t}$$

Generate t

```
nrow(monthly_smoothed_data)
```

```
## [1] 60
```

60 is even so

$$x_t = 2(t - median(t))$$

Month data

```
monthly_smoothed_data$t = seq(1:nrow(monthly_smoothed_data))
med_month_t = median(monthly_smoothed_data$t)
monthly_smoothed_data$x_t = 2*(monthly_smoothed_data$t - med_month_t)
monthly_smoothed_data$t_sqrd = monthly_smoothed_data$t^2
monthly_smoothed_data
```

```
##        month year tourists adj_seas_ind deseason  t x_t t_sqrd
## 1    January 2019    19500     84.61872 23044.55  1 -59      1
## 2   February 2020    21000     90.76591 23136.44  2 -57      4
## 3      March 2021    23000     95.97674 23964.14  3 -55      9
## 4      April 2022    25000    101.08602 24731.41  4 -53     16
## 5        May 2023    27000    106.26593 25407.96  5 -51     25
## 6       June 2019    30000    111.01201 27024.10  6 -49     36
## 7       July 2020    32000    117.90435 27140.65  7 -47     49
## 8     August 2021    31000    113.33957 27351.44  8 -45     64
## 9  September 2022    29000    105.61012 27459.49  9 -43     81
## 10   October 2023    27000     98.21944 27489.47 10 -41    100
## 11  November 2019    25000     91.28047 27388.12 11 -39    121
## 12  December 2020    23000     83.92073 27406.82 12 -37    144
## 13   January 2021    22000     84.61872 25998.98 13 -35    169
## 14  February 2022    24000     90.76591 26441.65 14 -33    196
```

```
## 15     March 2023    26000     95.97674 27089.90 15 -31    225
## 16     April 2019    28000    101.08602 27699.18 16 -29    256
## 17       May 2020    30000    106.26593 28231.06 17 -27    289
## 18      June 2021    32000    111.01201 28825.71 18 -25    324
## 19      July 2022    34000    117.90435 28836.94 19 -23    361
## 20    August 2023    33000    113.33957 29116.04 20 -21    400
## 21 September 2019    31000    105.61012 29353.25 21 -19    441
## 22   October 2020    29000     98.21944 29525.72 22 -17    484
## 23  November 2021    27000     91.28047 29579.17 23 -15    529
## 24  December 2022    25000     83.92073 29790.02 24 -13    576
## 25   January 2023    24500     84.61872 28953.41 25 -11    625
## 26  February 2019    27000     90.76591 29746.85 26  -9    676
## 27     March 2020    29000     95.97674 30215.66 27  -7    729
## 28     April 2021    31000    101.08602 30666.95 28  -5    784
## 29       May 2022    33000    106.26593 31054.17 29  -3    841
## 30      June 2023    35000    111.01201 31528.12 30  -1    900
## 31      July 2019    37000    117.90435 31381.37 31   1    961
## 32    August 2020    36000    113.33957 31762.96 32   3   1024
## 33 September 2021    34000    105.61012 32193.88 33   5   1089
## 34   October 2022    32000     98.21944 32580.11 34   7   1156
## 35  November 2023    30000     91.28047 32865.74 35   9   1225
## 36  December 2019    28000     83.92073 33364.82 36  11   1296
## 37   January 2020    30000     84.61872 35453.15 37  13   1369
## 38  February 2021    32000     90.76591 35255.53 38  15   1444
## 39     March 2022    34000     95.97674 35425.25 39  17   1521
## 40     April 2023    36000    101.08602 35613.23 40  19   1600
## 41       May 2019    38000    106.26593 35759.35 41  21   1681
## 42      June 2020    40000    111.01201 36032.14 42  23   1764
## 43      July 2021    42000    117.90435 35622.10 43  25   1849
## 44    August 2022    41000    113.33957 36174.48 44  27   1936
## 45 September 2023    39000    105.61012 36928.28 45  29   2025
## 46   October 2019    37000     98.21944 37670.75 46  31   2116
## 47  November 2020    35000     91.28047 38343.36 47  33   2209
## 48  December 2021    33000     83.92073 39322.83 48  35   2304
## 49   January 2022    35000     84.61872 41362.01 49  37   2401
## 50  February 2023    37000     90.76591 40764.20 50  39   2500
## 51     March 2019    39000     95.97674 40634.85 51  41   2601
## 52     April 2020    41000    101.08602 40559.51 52  43   2704
## 53       May 2021    43000    106.26593 40464.52 53  45   2809
## 54      June 2022    38000    111.01201 34230.53 54  47   2916
## 55      July 2023    42000    117.90435 35622.10 55  49   3025
## 56    August 2019    40000    113.33957 35292.17 56  51   3136
## 57 September 2020    37000    105.61012 35034.52 57  53   3249
## 58   October 2021    34000     98.21944 34616.36 58  55   3364
## 59  November 2022    32000     91.28047 35056.79 59  57   3481
## 60  December 2023    30000     83.92073 35748.02 60  59   3600
```

After the transformation, the normal equations for linear trend line are:

$$\sum Y_t = n\beta_0 + \beta_1 \sum X_t$$

```
sum(monthly_smoothed_data$x_t)
```

```
## [1] 0
```

therefore

$$\frac{\sum Y_t}{n} = \beta_0$$

additionally

$$\sum X_t Y_t = \beta_0 \sum X_t + \beta_1 \sum X_t^2$$
$$= \beta_1 \sum X_t^2$$

add the column for $X_t Y_t$ and X_t^2

```
monthly_smoothed_data$XY = monthly_smoothed_data$deseason * monthly_smoothed_data$x_t
monthly_smoothed_data$x_sqrd = monthly_smoothed_data$x_t^2
monthly_smoothed_data
```

```
##        month year tourists adj_seas_ind deseason  t x_t t_sqrd          XY
## 1    January 2019    19500     84.61872 23044.55  1 -59      1 -1359628.27
## 2   February 2020    21000     90.76591 23136.44  2 -57      4 -1318777.06
## 3      March 2021    23000     95.97674 23964.14  3 -55      9 -1318027.72
## 4      April 2022    25000    101.08602 24731.41  4 -53     16 -1310764.77
## 5        May 2023    27000    106.26593 25407.96  5 -51     25 -1295805.77
## 6       June 2019    30000    111.01201 27024.10  6 -49     36 -1324181.02
## 7       July 2020    32000    117.90435 27140.65  7 -47     49 -1275610.32
## 8     August 2021    31000    113.33957 27351.44  8 -45     64 -1230814.59
## 9  September 2022    29000    105.61012 27459.49  9 -43     81 -1180758.06
## 10   October 2023    27000     98.21944 27489.47 10 -41    100 -1127068.08
## 11  November 2019    25000     91.28047 27388.12 11 -39    121 -1068136.53
## 12  December 2020    23000     83.92073 27406.82 12 -37    144 -1014052.25
## 13   January 2021    22000     84.61872 25998.98 13 -35    169  -909964.16
## 14  February 2022    24000     90.76591 26441.65 14 -33    196  -872574.30
## 15     March 2023    26000     95.97674 27089.90 15 -31    225  -839786.83
## 16     April 2019    28000    101.08602 27699.18 16 -29    256  -803276.22
## 17       May 2020    30000    106.26593 28231.06 17 -27    289  -762238.69
## 18      June 2021    32000    111.01201 28825.71 18 -25    324  -720642.73
## 19      July 2022    34000    117.90435 28836.94 19 -23    361  -663249.51
## 20    August 2023    33000    113.33957 29116.04 20 -21    400  -611436.93
## 21 September 2019    31000    105.61012 29353.25 21 -19    441  -557711.71
## 22   October 2020    29000     98.21944 29525.72 22 -17    484  -501937.28
## 23  November 2021    27000     91.28047 29579.17 23 -15    529  -443687.48
## 24  December 2022    25000     83.92073 29790.02 24 -13    576  -387270.25
## 25   January 2023    24500     84.61872 28953.41 25 -11    625  -318487.46
## 26  February 2019    27000     90.76591 29746.85 26  -9    676  -267721.66
## 27     March 2020    29000     95.97674 30215.66 27  -7    729  -211509.59
## 28     April 2021    31000    101.08602 30666.95 28  -5    784  -153334.75
## 29       May 2022    33000    106.26593 31054.17 29  -3    841   -93162.51
## 30      June 2023    35000    111.01201 31528.12 30  -1    900   -31528.12
## 31      July 2019    37000    117.90435 31381.37 31   1    961    31381.37
## 32    August 2020    36000    113.33957 31762.96 32   3   1024    95288.87
## 33 September 2021    34000    105.61012 32193.88 33   5   1089   160969.42
## 34   October 2022    32000     98.21944 32580.11 34   7   1156   228060.75
## 35  November 2023    30000     91.28047 32865.74 35   9   1225   295791.65
## 36  December 2019    28000     83.92073 33364.82 36  11   1296   367013.04
## 37   January 2020    30000     84.61872 35453.15 37  13   1369   460890.94
## 38  February 2021    32000     90.76591 35255.53 38  15   1444   528832.91
## 39     March 2022    34000     95.97674 35425.25 39  17   1521   602229.27
## 40     April 2023    36000    101.08602 35613.23 40  19   1600   676651.40
## 41       May 2019    38000    106.26593 35759.35 41  21   1681   750946.26
## 42      June 2020    40000    111.01201 36032.14 42  23   1764   828739.14
## 43      July 2021    42000    117.90435 35622.10 43  25   1849   890552.42
## 44    August 2022    41000    113.33957 36174.48 44  27   1936   976710.94
## 45 September 2023    39000    105.61012 36928.28 45  29   2025  1070920.10
## 46   October 2019    37000     98.21944 37670.75 46  31   2116  1167793.21
```

```
## 47  November 2020    35000     91.28047 38343.36 47  33   2209  1265330.96
## 48  December 2021    33000     83.92073 39322.83 48  35   2304  1376298.88
## 49   January 2022    35000     84.61872 41362.01 49  37   2401  1530394.27
## 50  February 2023    37000     90.76591 40764.20 50  39   2500  1589803.93
## 51     March 2019    39000     95.97674 40634.85 51  41   2601  1666028.72
## 52     April 2020    41000    101.08602 40559.51 52  43   2704  1744059.09
## 53       May 2021    43000    106.26593 40464.52 53  45   2809  1820903.54
## 54      June 2022    38000    111.01201 34230.53 54  47   2916  1608834.90
## 55      July 2023    42000    117.90435 35622.10 55  49   3025  1745482.74
## 56    August 2019    40000    113.33957 35292.17 56  51   3136  1799900.91
## 57 September 2020    37000    105.61012 35034.52 57  53   3249  1856829.64
## 58   October 2021    34000     98.21944 34616.36 58  55   3364  1903900.01
## 59  November 2022    32000     91.28047 35056.79 59  57   3481  1998236.95
## 60  December 2023    30000     83.92073 35748.02 60  59   3600  2109133.35
##    x_sqrd
## 1    3481
## 2    3249
## 3    3025
## 4    2809
## 5    2601
## 6    2401
## 7    2209
## 8    2025
## 9    1849
## 10   1681
## 11   1521
## 12   1369
## 13   1225
## 14   1089
## 15    961
## 16    841
## 17    729
## 18    625
## 19    529
## 20    441
## 21    361
## 22    289
## 23    225
## 24    169
## 25    121
## 26     81
## 27     49
## 28     25
## 29      9
## 30      1
## 31      1
## 32      9
## 33     25
## 34     49
## 35     81
## 36    121
## 37    169
## 38    225
## 39    289
## 40    361
## 41    441
## 42    529
## 43    625
## 44    729
## 45    841
```

```
## 46    961
## 47   1089
## 48   1225
## 49   1369
## 50   1521
## 51   1681
## 52   1849
## 53   2025
## 54   2209
## 55   2401
## 56   2601
## 57   2809
## 58   3025
## 59   3249
## 60   3481
```

Trend Values

Test the best trend

1. simple linear

```
summary(lm(deseason~x_t, data = monthly_smoothed_data))
```

```
##
## Call:
## lm(formula = deseason ~ x_t, data = monthly_smoothed_data)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -4316.3  -831.0  -178.6  1186.0  4723.7
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 31922.204    247.892  128.78   <2e-16 ***
## x_t           127.463      7.157   17.81   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1920 on 58 degrees of freedom
## Multiple R-squared:  0.8454, Adjusted R-squared:  0.8427
## F-statistic: 317.2 on 1 and 58 DF,  p-value: < 2.2e-16
```

2. quadratic trend

```
summary(lm(deseason~poly(x_t, 2, raw = T), data = monthly_smoothed_data))
```

```
##
## Call:
## lm(formula = deseason ~ poly(x_t, 2, raw = T), data = monthly_smoothed_data)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -3347.7 -1131.4  -466.2   940.3  4813.5
##
## Coefficients:
##                          Estimate Std. Error t value Pr(>|t|)
## (Intercept)            32558.7662   357.7195  91.018   <2e-16 ***
```

```
## poly(x_t, 2, raw = T)1   127.4627     6.8837  18.517   <2e-16 ***
## poly(x_t, 2, raw = T)2    -0.5306     0.2223  -2.387   0.0203 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1847 on 57 degrees of freedom
## Multiple R-squared:  0.8595, Adjusted R-squared:  0.8545
## F-statistic: 174.3 on 2 and 57 DF,  p-value: < 2.2e-16
```

```
summary(lm(deseason~x_t + x_sqrd, data = monthly_smoothed_data))
```

```
##
## Call:
## lm(formula = deseason ~ x_t + x_sqrd, data = monthly_smoothed_data)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -3347.7 -1131.4  -466.2   940.3  4813.5
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept) 32558.7662   357.7195  91.018   <2e-16 ***
## x_t           127.4627     6.8837  18.517   <2e-16 ***
## x_sqrd         -0.5306     0.2223  -2.387   0.0203 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1847 on 57 degrees of freedom
## Multiple R-squared:  0.8595, Adjusted R-squared:  0.8545
## F-statistic: 174.3 on 2 and 57 DF,  p-value: < 2.2e-16
```

3. exponential

```
summary(nls(deseason~a + exp(b * x_t), data = monthly_smoothed_data, start = list(a = 1, b = 0.
1)))
```

```
##
## Formula: deseason ~ a + exp(b * x_t)
##
## Parameters:
##    Estimate Std. Error t value Pr(>|t|)
## a 3.148e+04  6.368e+02   49.44   <2e-16 ***
## b 1.497e-01  8.509e-03   17.60   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 4681 on 58 degrees of freedom
##
## Number of iterations to convergence: 8
## Achieved convergence tolerance: 3.996e-06
```

compare the models

Lower **Residual R** values indicate a better fit.

```
quadratic_model = lm(deseason~poly(x_t, 2, raw = T), data = monthly_smoothed_data)
linear_model = lm(deseason~x_t, data = monthly_smoothed_data)
```

```
linear_residual_R = summary(linear_model)$sigma
quadrat_residual_R = summary(quadratic_model)$sigma
if (linear_residual_R < quadrat_residual_R){
  print("Linear Model is the best model")
}else if(linear_residual_R == quadrat_residual_R){
  print("any model can be used")
}else{
  print("Quadratic Model is the best model")
}
```

```
## [1] "Quadratic Model is the best model"
```

quad vs exp

```
exp_model = nls(deseason~a + exp(b * x_t), data = monthly_smoothed_data, start = list(a = 1, b
= 0.1))

exp_residual_R = summary(exp_model)$sigma
if (exp_residual_R < quadrat_residual_R){
  print("Exponential Model is the best model")
}else if(linear_residual_R == quadrat_residual_R){
  print("any model can be used; Exponential or Quadratic")
}else{
  print("Quadratic Model is the best model")
}
```

```
## [1] "Quadratic Model is the best model"
```

Choosing quadratic model

```
coeff_monthly = coef(quadratic_model)
coeff_monthly
```

```
##           (Intercept) poly(x_t, 2, raw = T)1 poly(x_t, 2, raw = T)2
##          32558.7662374           127.4626974            -0.5306161
```

```
monthly_b0 = coeff_monthly[1]
monthly_b1 = coeff_monthly[2]
monthly_b2 = coeff_monthly[3]
```

```
monthly_smoothed_data$trend = monthly_b0 + (monthly_b1 * monthly_smoothed_data$t) + (monthly_b2
* monthly_smoothed_data$t_sqrd)
monthly_smoothed_data
```

```
##        month year tourists adj_seas_ind deseason  t x_t t_sqrd          XY
## 1    January 2019    19500     84.61872 23044.55  1 -59      1 -1359628.27
## 2   February 2020    21000     90.76591 23136.44  2 -57      4 -1318777.06
## 3      March 2021    23000     95.97674 23964.14  3 -55      9 -1318027.72
## 4      April 2022    25000    101.08602 24731.41  4 -53     16 -1310764.77
## 5        May 2023    27000    106.26593 25407.96  5 -51     25 -1295805.77
## 6       June 2019    30000    111.01201 27024.10  6 -49     36 -1324181.02
## 7       July 2020    32000    117.90435 27140.65  7 -47     49 -1275610.32
## 8     August 2021    31000    113.33957 27351.44  8 -45     64 -1230814.59
## 9  September 2022    29000    105.61012 27459.49  9 -43     81 -1180758.06
## 10   October 2023    27000     98.21944 27489.47 10 -41    100 -1127068.08
```

```
##  11   November 2019    25000     91.28047 27388.12 11 -39    121 -1068136.53
##  12   December 2020    23000     83.92073 27406.82 12 -37    144 -1014052.25
##  13    January 2021    22000     84.61872 25998.98 13 -35    169  -909964.16
##  14   February 2022    24000     90.76591 26441.65 14 -33    196  -872574.30
##  15      March 2023    26000     95.97674 27089.90 15 -31    225  -839786.83
##  16      April 2019    28000    101.08602 27699.18 16 -29    256  -803276.22
##  17        May 2020    30000    106.26593 28231.06 17 -27    289  -762238.69
##  18       June 2021    32000    111.01201 28825.71 18 -25    324  -720642.73
##  19       July 2022    34000    117.90435 28836.94 19 -23    361  -663249.51
##  20     August 2023    33000    113.33957 29116.04 20 -21    400  -611436.93
##  21  September 2019    31000    105.61012 29353.25 21 -19    441  -557711.71
##  22    October 2020    29000     98.21944 29525.72 22 -17    484  -501937.28
##  23   November 2021    27000     91.28047 29579.17 23 -15    529  -443687.48
##  24   December 2022    25000     83.92073 29790.02 24 -13    576  -387270.25
##  25    January 2023    24500     84.61872 28953.41 25 -11    625  -318487.46
##  26   February 2019    27000     90.76591 29746.85 26  -9    676  -267721.66
##  27      March 2020    29000     95.97674 30215.66 27  -7    729  -211509.59
##  28      April 2021    31000    101.08602 30666.95 28  -5    784  -153334.75
##  29        May 2022    33000    106.26593 31054.17 29  -3    841   -93162.51
##  30       June 2023    35000    111.01201 31528.12 30  -1    900   -31528.12
##  31       July 2019    37000    117.90435 31381.37 31   1    961    31381.37
##  32     August 2020    36000    113.33957 31762.96 32   3   1024    95288.87
##  33  September 2021    34000    105.61012 32193.88 33   5   1089   160969.42
##  34    October 2022    32000     98.21944 32580.11 34   7   1156   228060.75
##  35   November 2023    30000     91.28047 32865.74 35   9   1225   295791.65
##  36   December 2019    28000     83.92073 33364.82 36  11   1296   367013.04
##  37    January 2020    30000     84.61872 35453.15 37  13   1369   460890.94
##  38   February 2021    32000     90.76591 35255.53 38  15   1444   528832.91
##  39      March 2022    34000     95.97674 35425.25 39  17   1521   602229.27
##  40      April 2023    36000    101.08602 35613.23 40  19   1600   676651.40
##  41        May 2019    38000    106.26593 35759.35 41  21   1681   750946.26
##  42       June 2020    40000    111.01201 36032.14 42  23   1764   828739.14
##  43       July 2021    42000    117.90435 35622.10 43  25   1849   890552.42
##  44     August 2022    41000    113.33957 36174.48 44  27   1936   976710.94
##  45  September 2023    39000    105.61012 36928.28 45  29   2025  1070920.10
##  46    October 2019    37000     98.21944 37670.75 46  31   2116  1167793.21
##  47   November 2020    35000     91.28047 38343.36 47  33   2209  1265330.96
##  48   December 2021    33000     83.92073 39322.83 48  35   2304  1376298.88
##  49    January 2022    35000     84.61872 41362.01 49  37   2401  1530394.27
##  50   February 2023    37000     90.76591 40764.20 50  39   2500  1589803.93
##  51      March 2019    39000     95.97674 40634.85 51  41   2601  1666028.72
##  52      April 2020    41000    101.08602 40559.51 52  43   2704  1744059.09
##  53        May 2021    43000    106.26593 40464.52 53  45   2809  1820903.54
##  54       June 2022    38000    111.01201 34230.53 54  47   2916  1608834.90
##  55       July 2023    42000    117.90435 35622.10 55  49   3025  1745482.74
##  56     August 2019    40000    113.33957 35292.17 56  51   3136  1799900.91
##  57  September 2020    37000    105.61012 35034.52 57  53   3249  1856829.64
##  58    October 2021    34000     98.21944 34616.36 58  55   3364  1903900.01
##  59   November 2022    32000     91.28047 35056.79 59  57   3481  1998236.95
##  60   December 2023    30000     83.92073 35748.02 60  59   3600  2109133.35
##     x_sqrd    trend
## 1    3481 32685.70
## 2    3249 32811.57
## 3    3025 32936.38
## 4    2809 33060.13
## 5    2601 33182.81
## 6    2401 33304.44
## 7    2209 33425.00
## 8    2025 33544.51
## 9    1849 33662.95
```

```
## 10   1681 33780.33
## 11   1521 33896.65
## 12   1369 34011.91
## 13   1225 34126.11
## 14   1089 34239.24
## 15    961 34351.32
## 16    841 34462.33
## 17    729 34572.28
## 18    625 34681.18
## 19    529 34789.01
## 20    441 34895.77
## 21    361 35001.48
## 22    289 35106.13
## 23    225 35209.71
## 24    169 35312.24
## 25    121 35413.70
## 26     81 35514.10
## 27     49 35613.44
## 28     25 35711.72
## 29      9 35808.94
## 30      1 35905.09
## 31      1 36000.19
## 32      9 36094.22
## 33     25 36187.19
## 34     49 36279.11
## 35     81 36369.96
## 36    121 36459.74
## 37    169 36548.47
## 38    225 36636.14
## 39    289 36722.74
## 40    361 36808.29
## 41    441 36892.77
## 42    529 36976.19
## 43    625 37058.55
## 44    729 37139.85
## 45    841 37220.09
## 46    961 37299.27
## 47   1089 37377.38
## 48   1225 37454.44
## 49   1369 37530.43
## 50   1521 37605.36
## 51   1681 37679.23
## 52   1849 37752.04
## 53   2025 37823.79
## 54   2209 37894.48
## 55   2401 37964.10
## 56   2601 38032.67
## 57   2809 38100.17
## 58   3025 38166.61
## 59   3249 38231.99
## 60   3481 38296.31
```

quarter

```
nrow(quarter_smoothed_data)
```

```
## [1] 20
```

```
quarter_smoothed_data$t = seq(1, 20)
med_quarter = median(quarter_smoothed_data$t)
quarter_smoothed_data$x = 2 * (quarter_smoothed_data$t- med_quarter)
```

Compare the models

```
quadratic_model_q = lm(deseasoned~poly(x, 2, raw = T), data = quarter_smoothed_data)
linear_model_q = lm(deseasoned~x, data = quarter_smoothed_data)

linear_residual_R = summary(linear_model_q)$sigma
quadrat_residual_R = summary(quadratic_model_q)$sigma
if (linear_residual_R < quadrat_residual_R){
  print("Linear Model is the best model")
}else if(linear_residual_R == quadrat_residual_R){
  print("any model can be used")
}else{
  print("Quadratic Model is the best model")
}
```

```
## [1] "Quadratic Model is the best model"
```

```
quarter_smoothed_data$log_deseason = log(quarter_smoothed_data$deseasoned)
exp_model_q = lm(log_deseason~x, data = quarter_smoothed_data)

exp_residual_R = summary(exp_model_q)$sigma
if (exp_residual_R < quadrat_residual_R){
  print("Exponential Model is the best model")
}else if(linear_residual_R == quadrat_residual_R){
  print("any model can be used; Exponential or Quadratic")
}else{
  print("Quadratic Model is the best model")
}
```

```
## [1] "Exponential Model is the best model"
```

Generate the rest of the columns

```
quarter_smoothed_data$XY = quarter_smoothed_data$deseasoned * quarter_smoothed_data$x
quarter_smoothed_data$x_sqrd = quarter_smoothed_data$x^2
quarter_smoothed_data$t_sqrd = quarter_smoothed_data$t^2
quarter_coeff = coef(exp_model_q)
quarter_coeff
```

```
## (Intercept)           x
## 11.45783071  0.01279659
```

```
quarter_b0 = exp(quarter_coeff[1])
quarter_b0
```

```
## (Intercept)
##    94639.55
```

```
quarter_b1 = quarter_coeff[2]
quarter_b1
```

```
##          x
## 0.01279659
```

```
quarter_smoothed_data$trend = quarter_b0 + exp(quarter_b1 * quarter_smoothed_data$t)
quarter_smoothed_data
```

```
##    quarter year adj_seas_ind tourist deseasoned  t   x log_deseason         XY
## 1   Summer 2019     88.26648   63500   71941.24  1 -19     11.18360 -1366883.5
## 2   Autumn 2019    100.14883   75000   74888.54  2 -17     11.22376 -1273105.2
## 3   Winter 2019    113.40212   93000   82009.05  3 -15     11.31458 -1230135.7
## 4   Spring 2019     98.18256   81000   82499.37  4 -13     11.32055 -1072491.9
## 5   Autumn 2020    100.14883   71000   70894.49  5 -11     11.16895  -779839.4
## 6   Winter 2020    113.40212   84000   74072.69  6  -9     11.21280  -666654.2
## 7   Spring 2020     98.18256   99000  100832.57  7  -7     11.52122  -705828.0
## 8   Summer 2020     88.26648   87000   98565.16  8  -5     11.49847  -492825.8
## 9   Winter 2021    113.40212   79500   70104.51  9  -3     11.15774  -210313.5
## 10  Spring 2021     98.18256   93000   94721.50 10  -1     11.45870   -94721.5
## 11  Summer 2021     88.26648  108000  122356.75 11   1     11.71470   122356.8
## 12  Autumn 2021    100.14883   96000   95857.34 12   3     11.47062   287572.0
## 13  Spring 2022     98.18256   95000   96758.52 13   5     11.47997   483792.6
## 14  Summer 2022     88.26648  108000  122356.75 14   7     11.71470   856497.3
## 15  Autumn 2022    100.14883  123000  122817.21 15   9     11.71845  1105354.9
## 16  Winter 2022    113.40212  111000   97881.76 16  11     11.49152  1076699.4
## 17  Summer 2023     88.26648  102000  115559.15 17  13     11.65754  1502269.0
## 18  Autumn 2023    100.14883  123000  122817.21 18  15     11.71845  1842258.2
## 19  Winter 2023    113.40212  120000  105818.12 19  17     11.56948  1798908.1
## 20  Spring 2023     98.18256  103000  104906.61 20  19     11.56083  1993225.6
##    x_sqrd t_sqrd    trend
## 1     361      1 94640.56
## 2     289      4 94640.57
## 3     225      9 94640.59
## 4     169     16 94640.60
## 5     121     25 94640.61
## 6      81     36 94640.63
## 7      49     49 94640.64
## 8      25     64 94640.65
## 9       9     81 94640.67
## 10      1    100 94640.68
## 11      1    121 94640.70
## 12      9    144 94640.71
## 13     25    169 94640.73
## 14     49    196 94640.74
## 15     81    225 94640.76
## 16    121    256 94640.77
## 17    169    289 94640.79
## 18    225    324 94640.81
## 19    289    361 94640.82
## 20    361    400 94640.84
```

4. Obtain the cyclic index to better understand the cyclical variations in tourist numbers that may be influenced by economic cycles or other factors.

$$cyclic\ index = \frac{deseasonalized\ data}{trend\ value} * 100$$

Monthly

```
monthly_smoothed_data$cyclic = monthly_smoothed_data$deseason/ monthly_smoothed_data$trend * 10
0
monthly_smoothed_data
```

```
##        month year tourists adj_seas_ind deseason  t x_t t_sqrd          XY
## 1    January 2019    19500     84.61872 23044.55  1 -59      1 -1359628.27
## 2   February 2020    21000     90.76591 23136.44  2 -57      4 -1318777.06
## 3      March 2021    23000     95.97674 23964.14  3 -55      9 -1318027.72
## 4      April 2022    25000    101.08602 24731.41  4 -53     16 -1310764.77
## 5        May 2023    27000    106.26593 25407.96  5 -51     25 -1295805.77
## 6       June 2019    30000    111.01201 27024.10  6 -49     36 -1324181.02
## 7       July 2020    32000    117.90435 27140.65  7 -47     49 -1275610.32
## 8     August 2021    31000    113.33957 27351.44  8 -45     64 -1230814.59
## 9  September 2022    29000    105.61012 27459.49  9 -43     81 -1180758.06
## 10   October 2023    27000     98.21944 27489.47 10 -41    100 -1127068.08
## 11  November 2019    25000     91.28047 27388.12 11 -39    121 -1068136.53
## 12  December 2020    23000     83.92073 27406.82 12 -37    144 -1014052.25
## 13   January 2021    22000     84.61872 25998.98 13 -35    169  -909964.16
## 14  February 2022    24000     90.76591 26441.65 14 -33    196  -872574.30
## 15     March 2023    26000     95.97674 27089.90 15 -31    225  -839786.83
## 16     April 2019    28000    101.08602 27699.18 16 -29    256  -803276.22
## 17       May 2020    30000    106.26593 28231.06 17 -27    289  -762238.69
## 18      June 2021    32000    111.01201 28825.71 18 -25    324  -720642.73
## 19      July 2022    34000    117.90435 28836.94 19 -23    361  -663249.51
## 20    August 2023    33000    113.33957 29116.04 20 -21    400  -611436.93
## 21 September 2019    31000    105.61012 29353.25 21 -19    441  -557711.71
## 22   October 2020    29000     98.21944 29525.72 22 -17    484  -501937.28
## 23  November 2021    27000     91.28047 29579.17 23 -15    529  -443687.48
## 24  December 2022    25000     83.92073 29790.02 24 -13    576  -387270.25
## 25   January 2023    24500     84.61872 28953.41 25 -11    625  -318487.46
## 26  February 2019    27000     90.76591 29746.85 26  -9    676  -267721.66
## 27     March 2020    29000     95.97674 30215.66 27  -7    729  -211509.59
## 28     April 2021    31000    101.08602 30666.95 28  -5    784  -153334.75
## 29       May 2022    33000    106.26593 31054.17 29  -3    841   -93162.51
## 30      June 2023    35000    111.01201 31528.12 30  -1    900   -31528.12
## 31      July 2019    37000    117.90435 31381.37 31   1    961    31381.37
## 32    August 2020    36000    113.33957 31762.96 32   3   1024    95288.87
## 33 September 2021    34000    105.61012 32193.88 33   5   1089   160969.42
## 34   October 2022    32000     98.21944 32580.11 34   7   1156   228060.75
## 35  November 2023    30000     91.28047 32865.74 35   9   1225   295791.65
## 36  December 2019    28000     83.92073 33364.82 36  11   1296   367013.04
## 37   January 2020    30000     84.61872 35453.15 37  13   1369   460890.94
## 38  February 2021    32000     90.76591 35255.53 38  15   1444   528832.91
## 39     March 2022    34000     95.97674 35425.25 39  17   1521   602229.27
## 40     April 2023    36000    101.08602 35613.23 40  19   1600   676651.40
## 41       May 2019    38000    106.26593 35759.35 41  21   1681   750946.26
## 42      June 2020    40000    111.01201 36032.14 42  23   1764   828739.14
## 43      July 2021    42000    117.90435 35622.10 43  25   1849   890552.42
## 44    August 2022    41000    113.33957 36174.48 44  27   1936   976710.94
## 45 September 2023    39000    105.61012 36928.28 45  29   2025  1070920.10
## 46   October 2019    37000     98.21944 37670.75 46  31   2116  1167793.21
## 47  November 2020    35000     91.28047 38343.36 47  33   2209  1265330.96
## 48  December 2021    33000     83.92073 39322.83 48  35   2304  1376298.88
## 49   January 2022    35000     84.61872 41362.01 49  37   2401  1530394.27
## 50  February 2023    37000     90.76591 40764.20 50  39   2500  1589803.93
## 51     March 2019    39000     95.97674 40634.85 51  41   2601  1666028.72
## 52     April 2020    41000    101.08602 40559.51 52  43   2704  1744059.09
```

```
## 53       May 2021    43000    106.26593 40464.52 53  45   2809  1820903.54
## 54      June 2022    38000    111.01201 34230.53 54  47   2916  1608834.90
## 55      July 2023    42000    117.90435 35622.10 55  49   3025  1745482.74
## 56    August 2019    40000    113.33957 35292.17 56  51   3136  1799900.91
## 57 September 2020    37000    105.61012 35034.52 57  53   3249  1856829.64
## 58   October 2021    34000     98.21944 34616.36 58  55   3364  1903900.01
## 59  November 2022    32000     91.28047 35056.79 59  57   3481  1998236.95
## 60  December 2023    30000     83.92073 35748.02 60  59   3600  2109133.35
##    x_sqrd    trend    cyclic
## 1    3481 32685.70  70.50346
## 2    3249 32811.57  70.51305
## 3    3025 32936.38  72.75888
## 4    2809 33060.13  74.80737
## 5    2601 33182.81  76.56962
## 6    2401 33304.44  81.14264
## 7    2209 33425.00  81.19863
## 8    2025 33544.51  81.53774
## 9    1849 33662.95  81.57184
## 10   1681 33780.33  81.37713
## 11   1521 33896.65  80.79888
## 12   1369 34011.91  80.58006
## 13   1225 34126.11  76.18500
## 14   1089 34239.24  77.22614
## 15    961 34351.32  78.86131
## 16    841 34462.33  80.37524
## 17    729 34572.28  81.65808
## 18    625 34681.18  83.11630
## 19    529 34789.01  82.89095
## 20    441 34895.77  83.43716
## 21    361 35001.48  83.86287
## 22    289 35106.13  84.10418
## 23    225 35209.71  84.00854
## 24    169 35312.24  84.36175
## 25    121 35413.70  81.75764
## 26     81 35514.10  83.76068
## 27     49 35613.44  84.84341
## 28     25 35711.72  85.87363
## 29      9 35808.94  86.72184
## 30      1 35905.09  87.80960
## 31      1 36000.19  87.17002
## 32      9 36094.22  88.00012
## 33     25 36187.19  88.96485
## 34     49 36279.11  89.80405
## 35     81 36369.96  90.36508
## 36    121 36459.74  91.51140
## 37    169 36548.47  97.00309
## 38    225 36636.14  96.23156
## 39    289 36722.74  96.46679
## 40    361 36808.29  96.75329
## 41    441 36892.77  96.92779
## 42    529 36976.19  97.44685
## 43    625 37058.55  96.12382
## 44    729 37139.85  97.40071
## 45    841 37220.09  99.21599
## 46    961 37299.27 100.99595
## 47   1089 37377.38 102.58440
## 48   1225 37454.44 104.98843
## 49   1369 37530.43 110.20926
## 50   1521 37605.36 108.39998
## 51   1681 37679.23 107.84415
```

```
## 52   1849 37752.04 107.43661
## 53   2025 37823.79 106.98168
## 54   2209 37894.48  90.33119
## 55   2401 37964.10  93.83100
## 56   2601 38032.67  92.79438
## 57   2809 38100.17  91.95372
## 58   3025 38166.61  90.69803
## 59   3249 38231.99  91.69491
## 60   3481 38296.31  93.34587
```

Quarters

```
quarter_smoothed_data$cyclic = quarter_smoothed_data$deseason/ quarter_smoothed_data$trend * 10
0
quarter_smoothed_data
```

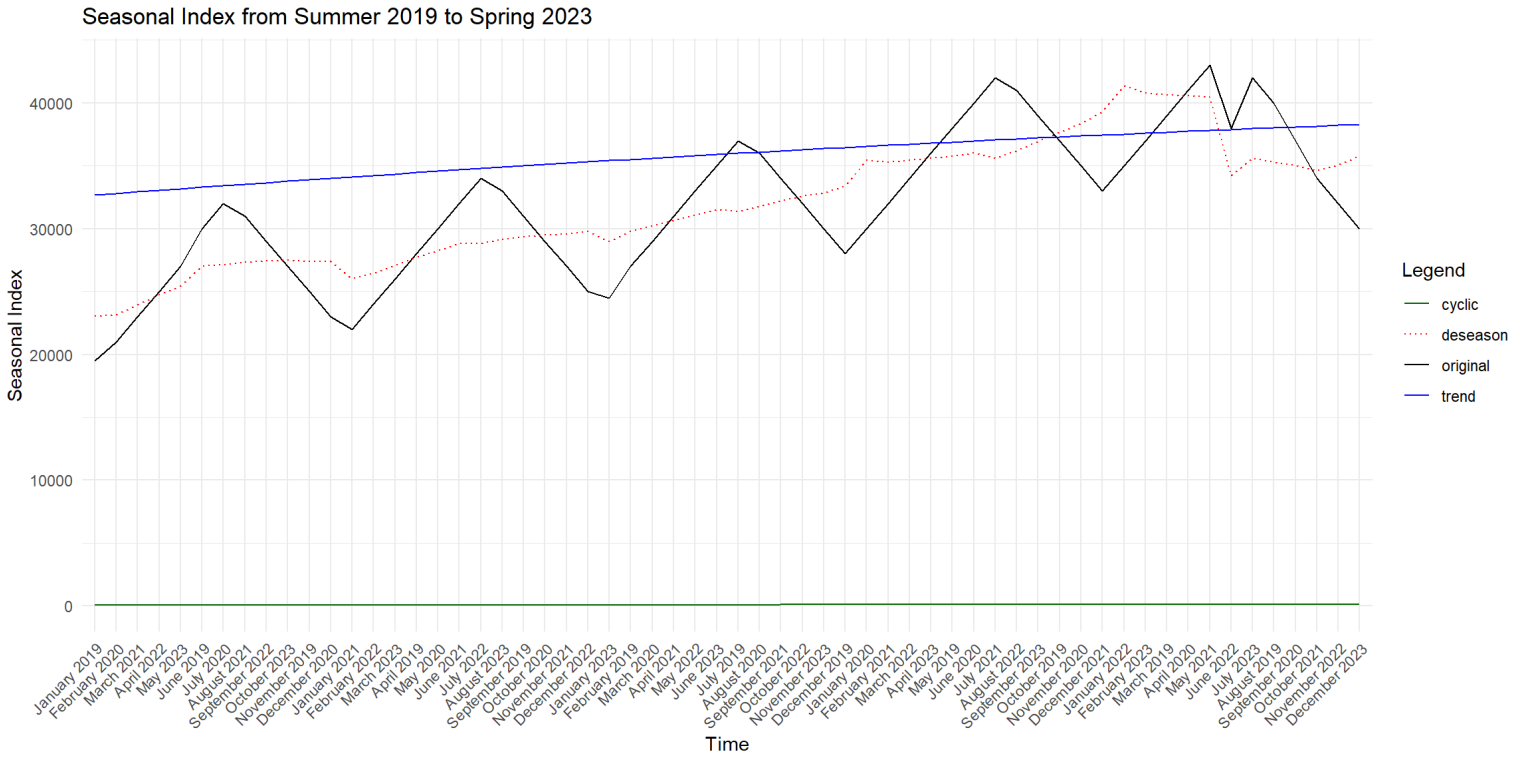
```
##    quarter year adj_seas_ind tourist deseasoned  t   x log_deseason         XY
## 1   Summer 2019     88.26648   63500   71941.24  1 -19     11.18360 -1366883.5
## 2   Autumn 2019    100.14883   75000   74888.54  2 -17     11.22376 -1273105.2
## 3   Winter 2019    113.40212   93000   82009.05  3 -15     11.31458 -1230135.7
## 4   Spring 2019     98.18256   81000   82499.37  4 -13     11.32055 -1072491.9
## 5   Autumn 2020    100.14883   71000   70894.49  5 -11     11.16895  -779839.4
## 6   Winter 2020    113.40212   84000   74072.69  6  -9     11.21280  -666654.2
## 7   Spring 2020     98.18256   99000  100832.57  7  -7     11.52122  -705828.0
## 8   Summer 2020     88.26648   87000   98565.16  8  -5     11.49847  -492825.8
## 9   Winter 2021    113.40212   79500   70104.51  9  -3     11.15774  -210313.5
## 10  Spring 2021     98.18256   93000   94721.50 10  -1     11.45870   -94721.5
## 11  Summer 2021     88.26648  108000  122356.75 11   1     11.71470   122356.8
## 12  Autumn 2021    100.14883   96000   95857.34 12   3     11.47062   287572.0
## 13  Spring 2022     98.18256   95000   96758.52 13   5     11.47997   483792.6
## 14  Summer 2022     88.26648  108000  122356.75 14   7     11.71470   856497.3
## 15  Autumn 2022    100.14883  123000  122817.21 15   9     11.71845  1105354.9
## 16  Winter 2022    113.40212  111000   97881.76 16  11     11.49152  1076699.4
## 17  Summer 2023     88.26648  102000  115559.15 17  13     11.65754  1502269.0
## 18  Autumn 2023    100.14883  123000  122817.21 18  15     11.71845  1842258.2
## 19  Winter 2023    113.40212  120000  105818.12 19  17     11.56948  1798908.1
## 20  Spring 2023     98.18256  103000  104906.61 20  19     11.56083  1993225.6
##    x_sqrd t_sqrd    trend    cyclic
## 1     361      1 94640.56  76.01523
## 2     289      4 94640.57  79.12943
## 3     225      9 94640.59  86.65315
## 4     169     16 94640.60  87.17123
## 5     121     25 94640.61  74.90916
## 6      81     36 94640.63  78.26732
## 7      49     49 94640.64 106.54257
## 8      25     64 94640.65 104.14674
## 9       9     81 94640.67  74.07440
## 10      1    100 94640.68 100.08540
## 11      1    121 94640.70 129.28555
## 12      9    144 94640.71 101.28552
## 13     25    169 94640.73 102.23772
## 14     49    196 94640.74 129.28549
## 15     81    225 94640.76 129.77201
## 16    121    256 94640.77 103.42452
## 17    169    289 94640.79 122.10290
## 18    225    324 94640.81 129.77194
## 19    289    361 94640.82 111.81023
## 20    361    400 94640.84 110.84709
```

5. Generate graphs of the data to illustrate the seasonal, trend, and cyclic components in the number of tourists visiting Cape Town.

Monthly

```
months = c(
  "January", "February", "March", "April", "May", "June",
  "July", "August", "September", "October", "November", "December"
)

monthly_smoothed_data$month = factor(monthly_smoothed_data$month, levels = months)
monthly_smoothed_data$season_year = paste(monthly_smoothed_data$month, monthly_smoothed_data$ye
ar)
monthly_smoothed_data$season_year = factor(monthly_smoothed_data$season_year, levels = unique(m
onthly_smoothed_data$season_year))
#
#
ggplot(monthly_smoothed_data, aes(x = season_year)) +
  geom_line(aes(y = tourists, color = "original"), group = 1) +
  # geom_line(aes(y = adj_seas_ind, color = "seasonal"), group = 1) +
  geom_line(aes(y = deseason, color = "deseason"), group = 1, linetype = "dotted") +
  geom_line(aes(y = trend, color = "trend"), group = 1) +
  geom_line(aes(y = cyclic, color = "cyclic"), group = 1) +
  labs(title = "Seasonal Index from Summer 2019 to Spring 2023",
       x = "Time",
       y = "Seasonal Index") +
  theme_minimal() +
  scale_color_manual(name = "Legend", values = c("original" = "black", "trend"= "blue", "cycli
c" = "darkgreen", "deseason" = "red"))+
  theme(axis.text.x = element_text(angle = 45, hjust = 1))
```



Using the monthly data,

1. There is an upward trend in the visitors to Capetown

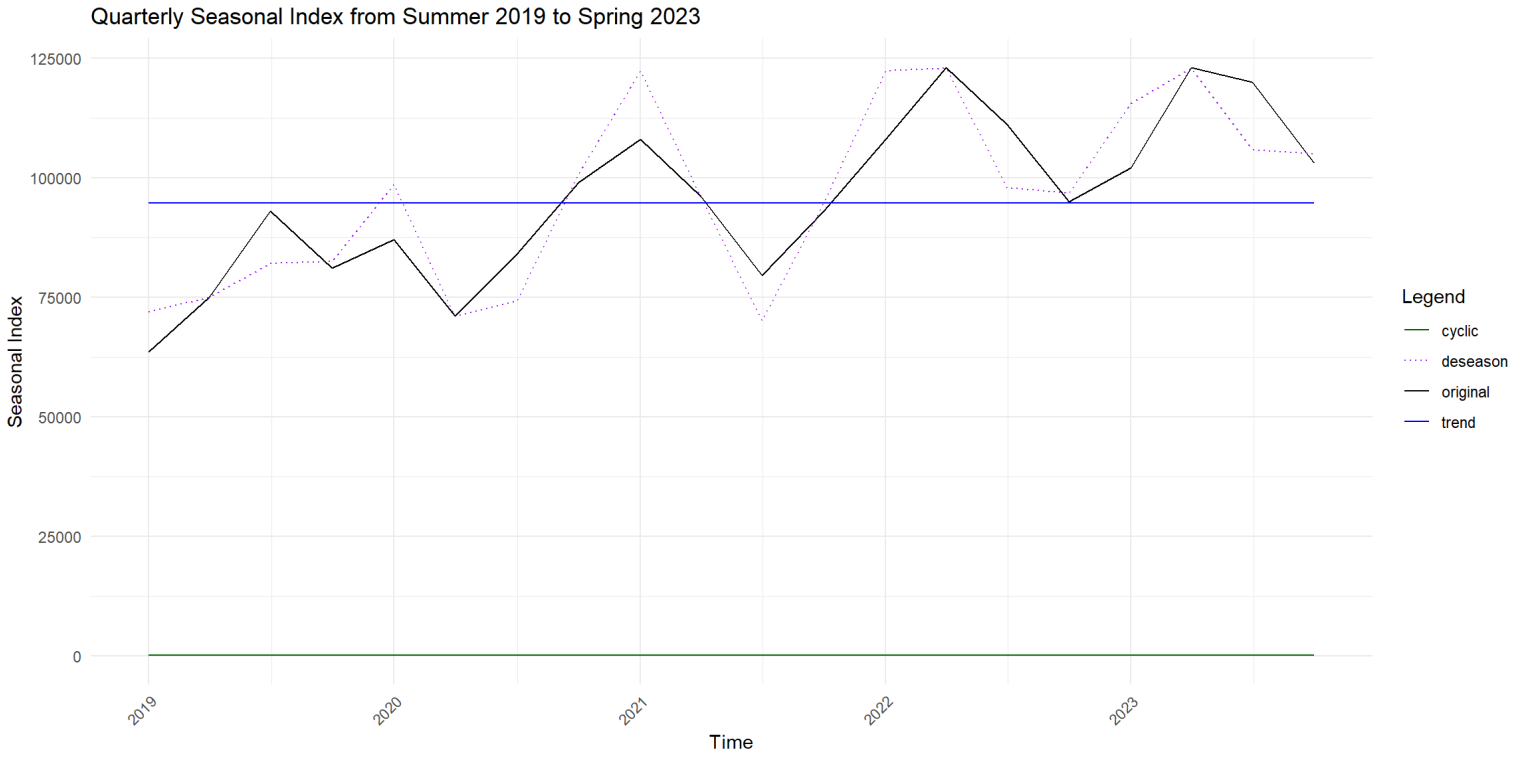
2. There does not seem to be a cycle where visitors stop or increase when visiting cape town
3. There is a very low rise in visitors to Capetown up to 2021 where there seems to be a spike and then a drop after may 2021

Quarter

```
quarter_smoothed_data$date = as.Date(paste(quarter_smoothed_data$year, as.numeric(quarter_smoothed_data$quarter) * 3 - 2, "1", sep = "-"), format = "%Y-%m-%d")

quarter_smoothed_data$season_year = paste(quarter_smoothed_data$quarter, quarter_smoothed_data$year)

ggplot(quarter_smoothed_data, aes(x = date)) +
  geom_line(aes(y = tourist, color = "original"), group = 1) +
  geom_line(aes(y = deseasoned, color = "deseason"), group = 1, linetype = "dotted") +
  geom_line(aes(y = trend, color = "trend"), group = 1) +
  geom_line(aes(y = cyclic, color = "cyclic"), group = 1) +
  labs(title = "Quarterly Seasonal Index from Summer 2019 to Spring 2023",
       x = "Time",
       y = "Seasonal Index") +
  theme_minimal() +
  scale_color_manual(name = "Legend", values = c("original" = "black", "trend"= "blue", "cyclic" = "darkgreen", "deseason" = "purple"))+
  theme(axis.text.x = element_text(angle = 45, hjust = 1))
```



The quarter trend

1. There seems to be no trend when viewed across quarters
2. There are peaks only during summer and then the visitation drops.