When a rule of thumb fails: CLT and Departure Delays dataset

Tural Sadigov

9/5/22

In this example, we will examine the rule of thumb, $n \geq 30$, for the Central Limit Theorem. Theorem says that the sampling distribution of the sample mean approaches to the Gaussian (Normal) distribution as the sample size increases (under the condition that mean and variance of the underlying distribution exists), and in practice, more than 30 data points is claimed to be sufficient to assume normality for the sampling distribution.

Consider the data consists of departure delays from Syracuse Airport in 2019 by three major airlines (United, Delta, American). Data is downloaded from Bureau of Transportation Statistics, and cleaned up.

Load libraries

```
library(tidyverse)
```

```
-- Attaching packages --------------------------------------- tidyverse 1.3.2 --
v ggplot2 3.3.6     v purrr   0.3.4
v tibble  3.1.8     v dplyr   1.0.9
v tidyr   1.2.0     v stringr 1.4.0
v readr   2.1.2     v forcats 0.5.1
-- Conflicts ------------------------------------------ tidyverse_conflicts() --
x dplyr::filter() masks stats::filter()
x dplyr::lag()    masks stats::lag()
```

```
library(infer)
library(cowplot)
```

Read data from GitHub

```
delays <- read_csv(file = "https://raw.githubusercontent.com/turalsadigov/MATH_254/main/da
```

```
Rows: 3161 Columns: 6
-- Column specification --------------------------------------------------------
Delimiter: ","
chr (4): Carrier, Date, Tail #, Destination
dbl (2): Flight #, Departure Delay Minutes

i Use `spec()` to retrieve the full column specification for this data.
i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
delays %>%
  head()
```

```
# A tibble: 6 x 6
  Carrier Date      `Flight #` `Tail #` Destination `Departure Delay Minutes`
  <chr>   <chr>          <dbl> <chr>    <chr>                             <dbl>
1 UA      1/8/2019         714 N810UA   ORD                                  32
2 UA      1/9/2019         714 N4888U   ORD                                  -2
3 UA      1/10/2019        714 N816UA   ORD                                   0
4 UA      1/11/2019        714 N4888U   ORD                                  28
5 UA      1/12/2019        714 N897UA   ORD                                  -1
6 UA      1/13/2019        714 N832UA   ORD                                   4
```

Summary of categorical variables

```
# base R counts
table(delays$Carrier)
```

```
  AA   DL   UA
1157 1515  489
```

```
table(delays$Destination)
```

```
 ATL  CLT  DTW  EWR  MSP  ORD
1038  975  259    1  218  670
```

```r
# tidyverse/dplyr counts
delays %>%
  count(Carrier)
```

```
# A tibble: 3 x 2
  Carrier     n
  <chr>   <int>
1 AA       1157
2 DL       1515
3 UA        489
```

```r
delays %>%
  count(Destination)
```

```
# A tibble: 6 x 2
  Destination     n
  <chr>       <int>
1 ATL          1038
2 CLT           975
3 DTW           259
4 EWR             1
5 MSP           218
6 ORD           670
```

Extract delay minutes, look at numerical summary

```r
# base R
delay_mins = delays$`Departure Delay Minutes`
summary(delay_mins)
```

```
    Min.  1st Qu.   Median     Mean  3rd Qu.     Max.
-24.000   -7.000   -4.000    7.035    0.000 1232.000
```

```r
# tidyverse/dplyr
delays %>%
  pull(`Departure Delay Minutes`) %>%
  summary()
```

```
   Min.  1st Qu.   Median     Mean  3rd Qu.     Max.
-24.000   -7.000   -4.000    7.035    0.000 1232.000
```

```r
# or
delays %>%
  select(`Departure Delay Minutes`) %>%
  summary()
```

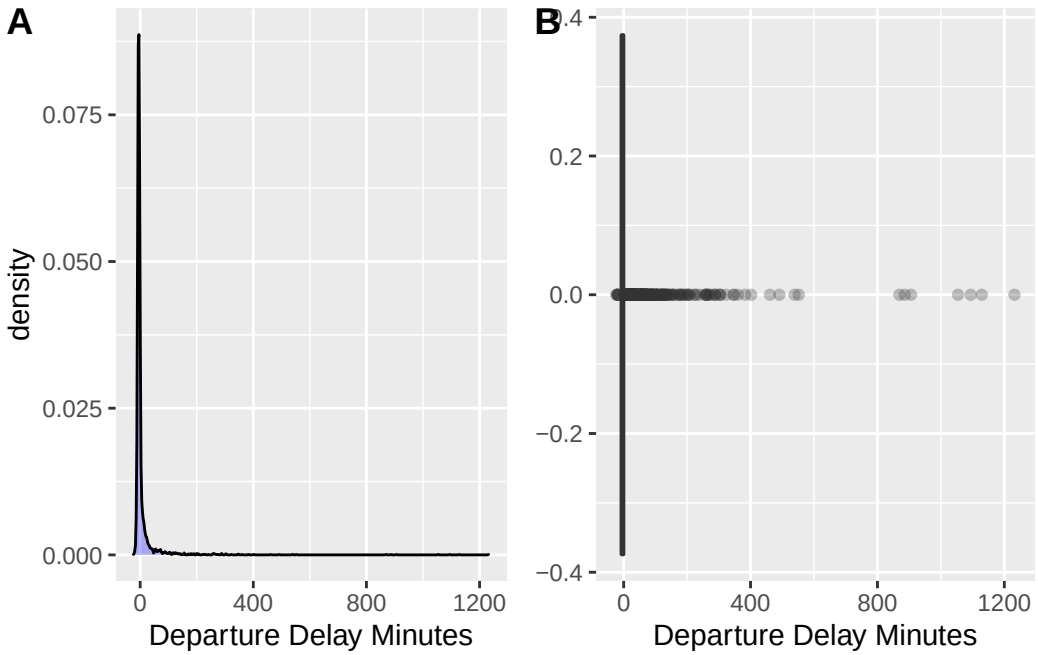
```
Departure Delay Minutes
Min.    : -24.000
1st Qu.:  -7.000
Median :  -4.000
Mean   :   7.035
3rd Qu.:   0.000
Max.   :1232.000
```

Population distribution

```r
plot1 <-
  delays %>%
  ggplot(aes(x = `Departure Delay Minutes`)) +
  geom_density(fill = 'blue', alpha = 0.3)

plot2 <-
  delays %>%
  ggplot(aes(x = `Departure Delay Minutes`)) +
  geom_boxplot(fill = 'blue', alpha = 0.3)

plot_grid(plot1, plot2, labels = "AUTO")
```



Now, we will be sampling from this population 1000 samples with various sizes: $20, 30, 50, 100, 300, 600, 1000, 1500, 2000$. For each size, we will construct the sampling distribution of the sample mean of the sampled delay minutes, and look at the distribution. As sample size increase, the sampling distribution will tend to look more Gaussian, but we will also pay attention to the rule of thumb, $n \geq 30$.

Initialize the number of simulated samples and sample sizes

```
sample_sizes = c(20, 30, 50, 100, 300, 600, 1000, 1500, 2000)
replicate_size = 1000  # number of samples
df <- tibble(replicate = 1:replicate_size)
df
```

```
# A tibble: 1,000 x 1
   replicate
       <int>
 1         1
 2         2
 3         3
 4         4
```

```
 5         5
 6         6
 7         7
 8         8
 9         9
10        10
# ... with 990 more rows
# i Use `print(n = ...)` to see more rows
```

Create the sampling distributions of sample means for various sizes

```
set.seed(2022)
for(sample_size in sample_sizes){

  new_df <-
    delays %>%
    rep_sample_n(size = sample_size,
                 reps = 1000,
                 replace = FALSE) %>%
    group_by(replicate) %>%
    summarise(mean = mean(`Departure Delay Minutes`))

  df <-
    df %>%
    inner_join(new_df, by = 'replicate')
}
colnames(df) <- c('replicate', sample_sizes)
df
```

```
# A tibble: 1,000 x 10
   replicate  `20`  `30`  `50` `100` `300` `600` `1000` `1500` `2000`
       <int> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl>  <dbl>  <dbl>  <dbl>
 1         1  10.9 12.5   0.92  1.45  9.05 14.0    8.74   6.64   7.15
 2         2   6   11.6  24.7   7.99  9.29  4.98   8.80   5.21   6.69
 3         3  -0.4 -1.2  -1.24  8.39  2.55  3.13   6.48   6.97   6.79
 4         4  -3.9 -4.77 13.3   6.06  9.19  4.51   6.18   7.06   6.18
 5         5  -6.1  4.7   3.22  1.39  7.65  6.35   7.54   7.70   6.91
 6         6  -2    2.63  3.36  1.79  5.75  8.98  11.2    7.82   7.45
 7         7  41.8  4.27  3.84 12.9  12.7   3.42   5.35   7.58   6.82
 8         8  -5.3  2.93 -0.3   4.57  2.43  6.89   6.96   5.10   7.19
```

```
 9         9  28.0 35.3   9.9   1.17 11.9   8.10   6.28   5.61   7.05
10        10  59    4.13 -2.84 11.9   4.92  6.60   5.18   7.20   7.26
# ... with 990 more rows
# i Use `print(n = ...)` to see more rows
```

Pivot longer

```
df %>%
  pivot_longer(cols = !replicate,
               names_to = 'sample_sizes',
               values_to = 'X_bar') %>%
  head(20)
```

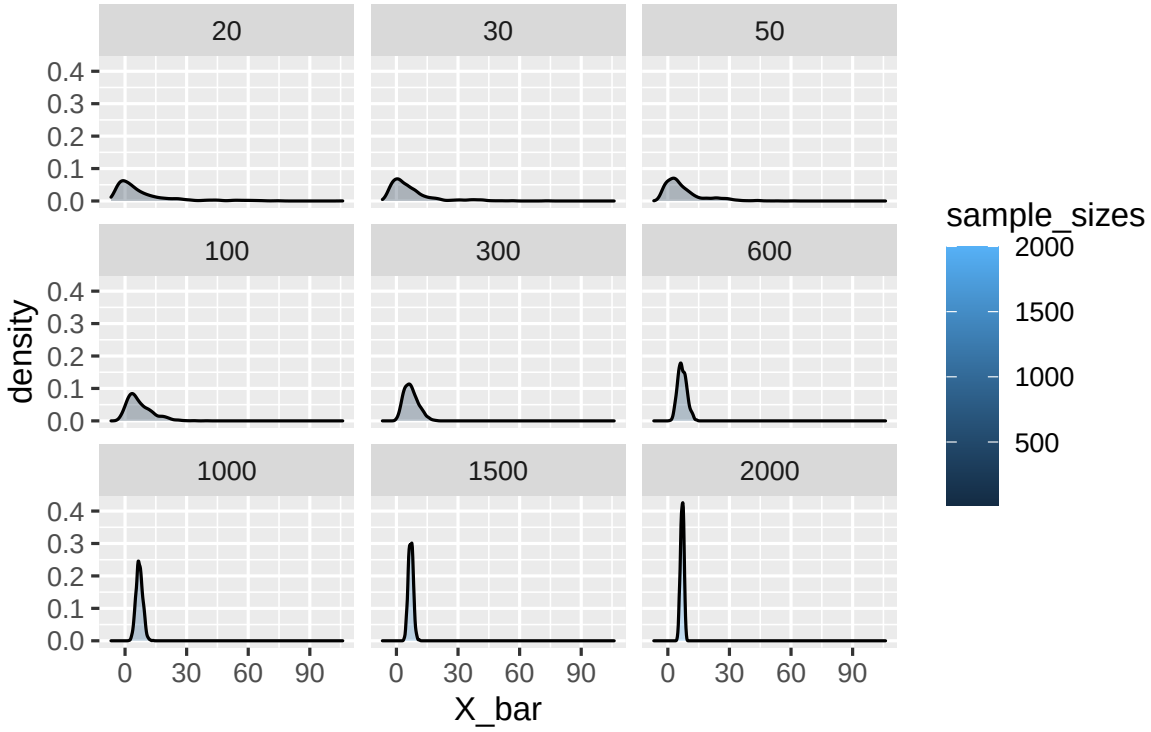
```
# A tibble: 20 x 3
   replicate sample_sizes X_bar
       <int> <chr>        <dbl>
 1         1 20           10.9
 2         1 30           12.5
 3         1 50            0.92
 4         1 100           1.45
 5         1 300           9.05
 6         1 600          14.0
 7         1 1000          8.74
 8         1 1500          6.64
 9         1 2000          7.15
10         2 20            6
11         2 30           11.6
12         2 50           24.7
13         2 100           7.99
14         2 300           9.29
15         2 600           4.98
16         2 1000          8.80
17         2 1500          5.21
18         2 2000          6.69
19         3 20           -0.4
20         3 30           -1.2
```

```
df %>%
  pivot_longer(cols = !replicate,
               names_to = 'sample_sizes',
               values_to = 'X_bar') %>%
```

```
    mutate(sample_sizes = as.integer(sample_sizes))
```

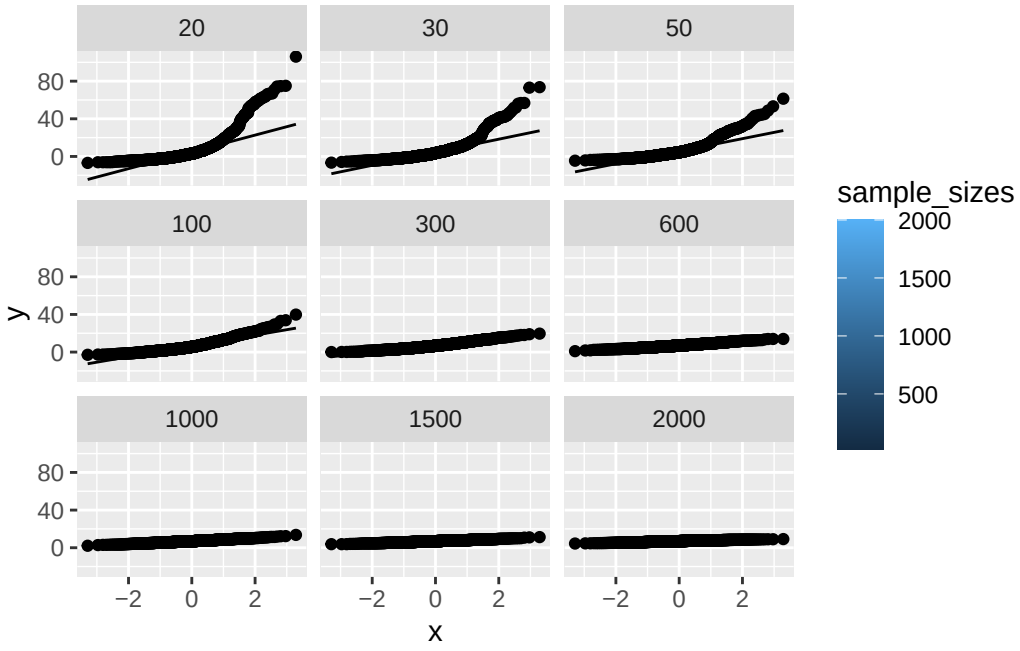
```
# A tibble: 9,000 x 3
   replicate sample_sizes X_bar
       <int>        <int> <dbl>
 1         1           20 10.9
 2         1           30 12.5
 3         1           50  0.92
 4         1          100  1.45
 5         1          300  9.05
 6         1          600 14.0
 7         1         1000  8.74
 8         1         1500  6.64
 9         1         2000  7.15
10         2           20  6
# ... with 8,990 more rows
# i Use `print(n = ...)` to see more rows
```

```
  df %>%
    pivot_longer(cols = !replicate,
                 names_to = 'sample_sizes',
                 values_to = 'X_bar') %>%
    mutate(sample_sizes = as.integer(sample_sizes)) %>%
    ggplot(aes(x = X_bar, fill = sample_sizes)) +
    geom_density(alpha = 0.3) +
    facet_wrap(~sample_sizes)
```



Check normality via QQ-plots

```
df %>%
  pivot_longer(cols = !replicate,
               names_to = 'sample_sizes',
               values_to = 'X_bar') %>%
  mutate(sample_sizes = as.integer(sample_sizes)) %>%
  ggplot(aes(sample = X_bar, fill = sample_sizes)) +
  stat_qq() +
  stat_qq_line() +
  facet_wrap(~sample_sizes)
```


20
30
50
100
300
600
1000
1500
2000
80
40
0
y
−2
0
2
x
sample_sizes
2000
1500
1000
500