Homework 04: Data frames and data wrangling

Please read the entire chapter on data transformation from
R for Data Science before starting this homework.

This homework relies on the `nycflights13` package, which contains several data frames, including:

- `airlines`
- `airports`
- `flights`
- `planes`
- `weather`

Loading `nycflights13` puts these data frames on the search path.

Setup

Load packages (do this every time)

```
library(tidyverse)
```

```
── Attaching core tidyverse packages ──────────────────────── tidyverse 2.0.0 ──
✓ dplyr     1.1.4     ✓ readr     2.1.6
✓ forcats   1.0.1     ✓ stringr   1.6.0
✓ ggplot2   4.0.1     ✓ tibble    3.3.1
✓ lubridate 1.9.4     ✓ tidyr     1.3.2
✓ purrr     1.2.1
── 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 become
errors
```

```
library(nycflights13)
```

Note - if you don't have these installed you will need to first install.packages('nycflights13')

Question 1: Filtering

Make a plot of **air time vs distance** (air time on the y-axis, distance on the x-axis) for all flights that meet the following criteria:

- originate from LaGuardia airport (`"LGA"`)

- departed on the 16th of the month
- have a flight distance of less than 2000 miles

Your code

```
flights
```

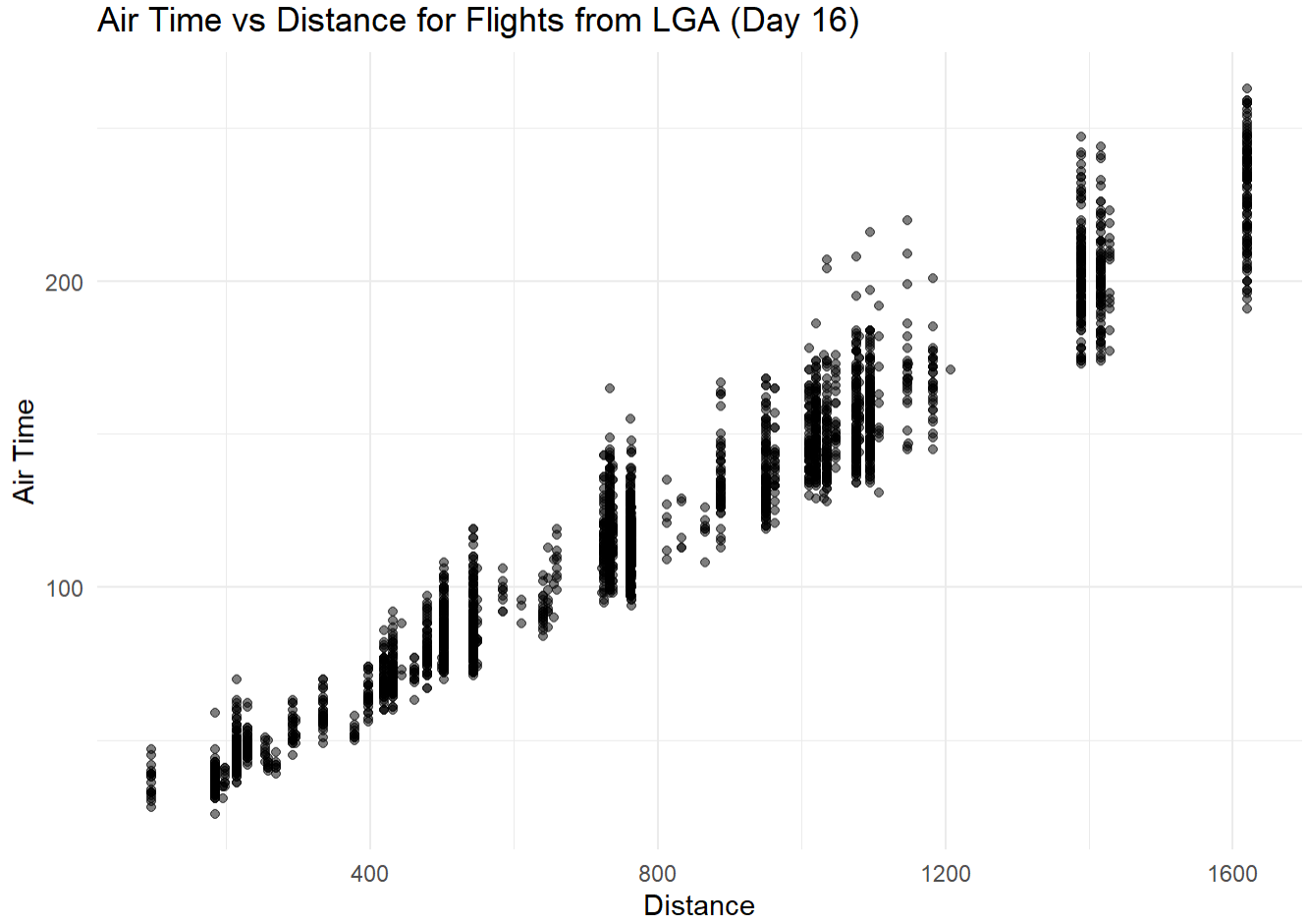
```
# A tibble: 336,776 × 19
    year month   day dep_time sched_dep_time dep_delay arr_time sched_arr_time
   <int> <int> <int>    <int>          <int>     <dbl>    <int>          <int>
 1  2013     1     1      517            515         2      830            819
 2  2013     1     1      533            529         4      850            830
 3  2013     1     1      542            540         2      923            850
 4  2013     1     1      544            545        -1     1004           1022
 5  2013     1     1      554            600        -6      812            837
 6  2013     1     1      554            558        -4      740            728
 7  2013     1     1      555            600        -5      913            854
 8  2013     1     1      557            600        -3      709            723
 9  2013     1     1      557            600        -3      838            846
10  2013     1     1      558            600        -2      753            745
# ℹ 336,766 more rows
# ℹ 11 more variables: arr_delay <dbl>, carrier <chr>, flight <int>,
#   tailnum <chr>, origin <chr>, dest <chr>, air_time <dbl>, distance <dbl>,
#   hour <dbl>, minute <dbl>, time_hour <dttm>
```

```
View(flights)
glimpse(flights)
```

```
Rows: 336,776
Columns: 19
$ year           <int> 2013, 2013, 2013, 2013, 2013, 2013, 2013, 2013, 2013, 2…
$ month          <int> 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1…
$ day            <int> 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1…
$ dep_time       <int> 517, 533, 542, 544, 554, 554, 555, 557, 557, 558, 558, …
$ sched_dep_time <int> 515, 529, 540, 545, 600, 558, 600, 600, 600, 600, 600, …
$ dep_delay      <dbl> 2, 4, 2, -1, -6, -4, -5, -3, -3, -2, -2, -2, -2, -2, -1…
$ arr_time       <int> 830, 850, 923, 1004, 812, 740, 913, 709, 838, 753, 849,…
$ sched_arr_time <int> 819, 830, 850, 1022, 837, 728, 854, 723, 846, 745, 851,…
$ arr_delay      <dbl> 11, 20, 33, -18, -25, 12, 19, -14, -8, 8, -2, -3, 7, -1…
$ carrier        <chr> "UA", "UA", "AA", "B6", "DL", "UA", "B6", "EV", "B6", "…
$ flight         <int> 1545, 1714, 1141, 725, 461, 1696, 507, 5708, 79, 301, 4…
$ tailnum        <chr> "N14228", "N24211", "N619AA", "N804JB", "N668DN", "N394…
$ origin         <chr> "EWR", "LGA", "JFK", "JFK", "LGA", "EWR", "EWR", "LGA",…
$ dest           <chr> "IAH", "IAH", "MIA", "BQN", "ATL", "ORD", "FLL", "IAD",…
$ air_time       <dbl> 227, 227, 160, 183, 116, 150, 158, 53, 140, 138, 149, 1…
$ distance       <dbl> 1400, 1416, 1089, 1576, 762, 719, 1065, 229, 944, 733, …
$ hour           <dbl> 5, 5, 5, 5, 6, 5, 6, 6, 6, 6, 6, 6, 6, 6, 6, 5, 6, 6, 6…
$ minute         <dbl> 15, 29, 40, 45, 0, 58, 0, 0, 0, 0, 0, 0, 0, 0, 0, 59, 0…
```

```
$ time_hour      <dttm> 2013-01-01 05:00:00, 2013-01-01 05:00:00, 2013-01-01 0…
```

```r
library(dplyr)
library(ggplot2)
lga_flights <- flights %>%
  filter(
    origin == "LGA",
    day == 16,
    distance < 2000,
    !is.na(air_time),
    !is.na(distance)
  )
ggplot(lga_flights, aes(x = distance, y = air_time)) +
  geom_point(alpha = 0.5) +
  labs(
    title = "Air Time vs Distance for Flights from LGA (Day 16)",
    x = "Distance",
    y = "Air Time"
  ) +
  theme_minimal()
```



Brief written response

In 2–4 sentences, describe what you observe in the plot. (For example: Is the relationship roughly linear? Are there any clear outliers?)

Your answer here: Air time increases as distance increases in a mostly linear way. The points cluster by route, with a few flights taking slightly longer than expected.

Question 2: Dealing with NAs

Make a data frame of all rows of `flights` that have values for *both* `arr_time` and `dep_time` (i.e., neither value is `NA`).

Your code

```
# Create a new data frame that removes rows where arr_time or dep_time is NA.
flights_times <- flights %>%
  filter(!is.na(arr_time), !is.na(dep_time))
```

Filtering NAs (conceptual)

`ggplot()` will automatically remove `NA` values from a plot, but it emits a warning message about it. You *could* silence warnings using chunk options, but instead:

Brief written response

Explain (in words) how you could prevent those `NA` values from appearing in the plot in the first place.

Your answer here: You can prevent NA values from appearing in the plot by removing them from the data before plotting. This can be done by filtering out rows where the relevant variables are NA, so ggplot never sees those missing values and does not need to drop them automatically.

Question 3: Adding columns

Create a data frame of **average flight speeds**, based on `air_time` and `distance`.

Then make either:

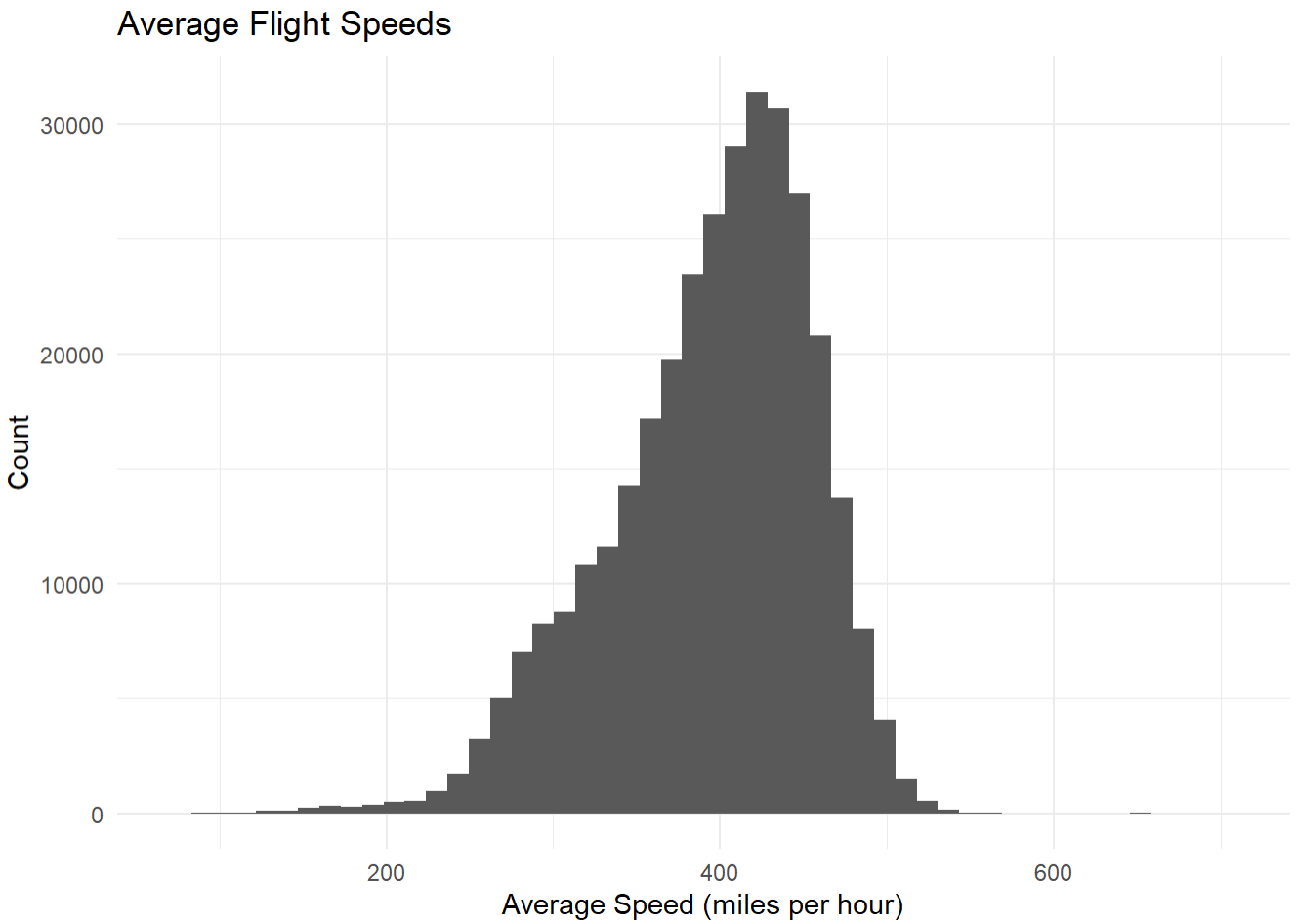
- a histogram, **or**
- a density plot

If you like, you may break the data out (e.g., by airline or another variable) in a way that you think makes

sense.

Your code

```
# Create a new column for average speed.
# (Hint: think carefully about units — air_time is in minutes.)
# Then make a histogram or density plot.
flight_speeds <- flights %>%
  filter(!is.na(air_time), !is.na(distance), air_time > 0) %>%
  mutate(avg_speed = distance / (air_time / 60))
ggplot(flight_speeds, aes(x = avg_speed)) +
  geom_histogram(bins = 50) +
  labs(
    title = "Average Flight Speeds",
    x = "Average Speed (miles per hour)",
    y = "Count"
  ) +
  theme_minimal()
```



Brief written response

Describe the main features of your speed distribution.

Your answer here: The distribution has a single clear peak and most flights are clustered around 400–450 miles per hour. There are fewer very slow or very fast flights, hence the thin tails on both ends.

Rendering and submission (GitHub)

Canvas contains the course GitHub link and instructions for **forking** the course repository.

In this file, your job is to:

1. Complete the homework by adding code + written responses above.
2. Render this Quarto file to **HTML or PDF**.
3. Move the rendered file(s) to the /doc folder (note that, if you render to html you will also have to move your hmk_04_data_frames_files folder)
4. Add/commit both:
    - this edited `.qmd` file
    - the rendered output file(s) (HTML or PDF)
5. Push your changes to your fork on GitHub.
6. Open a Pull Request back to the course repository.

What to submit on Canvas

On Canvas, submit a link to your GitHub fork repository (not files).