Project Report of Discrete and Continuous Random Samples

Aman Hogan-Bailey, Chuan Ngoc Ha

The University of Texas at Arlington

Contents

Project Report of Discrete and Continuous Random Samples

Introduction

Nearly all engineers require a basic understanding of risk assessment, data analysis, or measurement analysis, all of which use statistics and probability. The overall goal of this project is to allow students to experience gathering and analyzing real-world data. This project aims to have students accomplish the following:

1. Gather two sets of data from the real world.
2. Summarize each set of data statistically.
3. Perform statistical chi-square tests on each set of data.
4. Create a written final report describing the steps above.

Data Collection

Set #1 (Length of Major North American Rivers)

Both data sets in this project used data publicly available from online repositories. The first data set (Set #1) is the length of 'major' rivers in North America, in miles, from the 1975 World Almanac and Book of Facts (Table 3). The river lengths were likely determined by connecting a long continuous line in each river using aerial photos. These lines could then be measured using satellite imagery to produce accurate distances and later stored in a data repository. Additionally, Set #1 consists of 141 rivers, rounds the length of the rivers to the nearest mile, is a continuous random sample, and was initially assumed to be normally distributed.

Set #2 (Interarrival times of Southwest Airlines flights to Dallas on 01/14/2021)

The second raw data set (Set #2) is the arrival times of Southwest Airlines flights to Dallas on January 14th, 2021, from 6:00 AM to 2:00 PM, publicly available from the Bureau of

Transportation Statistics (Table 4). The arrival times may have been collected by the Air Traffic Control Centers or the Airplane's navigation system when it lands and later stored inside a data repository. Additionally, Set #2 consists of 106 events, has a 'NULL' value for the first event, and arrival times rounded to the nearest minute.

Descriptive Statistics

Tools Used

The data was processed and analyzed using: Microsoft Excel, Python 3, and the Pycharm IDE.

Procedure/Explanation of Descriptive Statistics Analysis

1. The data sets were both loaded into individual Excel spreadsheet files. Set #1 was stripped of any extraneous text such that the file only contained two columns of raw numerical data.
2. Set #2 also was stripped of additional text such that the file only had one column of arrival times in the format HH: MM: SS.
3. The files were saved as a .csv.
4. The files were loaded into a developed source code in a python project in the Pycharm IDE.
5. The program was run, and Set #1 was parsed and analyzed using statistics-based python modules: pandas, numpy, and matplotlib. After the program analyzes Set #1, the program outputs one .png of a histogram of the data set, a .png file of a boxplot of the data set, and a .txt file containing the descriptive statistics and a frequency table.

6. Once Set #1 finishes, Set #2 is also parsed. Set #2 then has the inter-arrival times calculated in the program, and afterward, undergoes the same analysis process as Set #1. The program finishes after Set #1 and Set #2 are processed.
7. The corresponding output files can be found in the same directory as the main python file.

Interpretation of Descriptive Statistics Analysis for Managers

Set #1.

For Set #1, the average length of major North American Rivers is 591 miles (Table 5). On average, the rivers would be either 494 miles above or below the average. 25% of the rivers are shorter than 310 miles, 50% are shorter than 425 miles, and 75% are less than 680 miles. Most of the rivers were between 131 and 850 miles in length. There were 11 outliers, all of which were over 1200 miles long, one of these being a river that was longer than 3500 miles long (Figure 3). The histogram of the rivers is right-skewed, meaning most of the rivers were on the 'shorter' side in length (Figure 4).

Set #1.

For Set #2, the average time for another plane to land after the previous one was 258.68 seconds (Table 6). On average, the inter-arrival time would be either 293.51 seconds above or below the average. 25% of the inter-arrival times are shorter than 120 seconds, 50% are less than 180 seconds, and 75% are less than 300 seconds. Most of the inter-arrival times are between 0 and 432 seconds. There were seven outliers, all of which were over 500 seconds long, one of these being an inter-arrival time that was longer than 2000 seconds (Figure 1). The histogram of the inter-arrival times is right-skewed, meaning most of the inter-arrival times were on the 'shorter' side in length (Figure 2).

Distribution of Set #1.

Set #1 does not appear to be a normal distribution. One of the main components of a normal distribution is that the data points are symmetric with no skew. Set #1's Histogram shows a right skew (Figure 4).

Distribution of Set #2.

Set #2 is a more complicated story: when the data is collected correctly, then Set #2 would be an exponential distribution. An exponential distribution follows the Poisson process, which entails these key points:

1. The average time between events is constant.
2. The events are independent.
3. No two events occur at the same time.

So, our distribution follows all the points except for point number 3; two planes can land simultaneously. So, Set #2 is not an exponential distribution (Figure 2). An exponential distribution data set could have been found, had there been more concise wording/understanding of the project requirements.

Conclusion

In conclusion, the overall goal of this project was to expose students to real-world applications by gathering data sets, summarizing data, performing analysis, and writing academic reports. From collecting the data, developing the program to analyze the data, and drafting the report of the data, we feel we achieved that experience. This project allowed us to gain insight into the underlying calculations behind much of the visualized data we consumed. This project was very beneficial and relevant to our professional degrees.

References

(2021, January 14). Retrieved October 25, 2022, from https://www.bts.gov/.

The World Almanac And Book of Facts, 1975-. (1975). New York: World Almanac Books.

Appendix I

Tables

Table 1

Class Frequency Table for Length of Major North American Rivers

Length (L) [Miles]	**Frequency (f) [#]**
(131, 850]	118
(850, 1565]	17
(2280, 2995]	3
(1565, 2280]	2
(2995, 3710]	1

Table 1 - shows the frequency table using five bins. Most of the data lies between (131, 850].

Table 2

Class Frequency Table for Inter-arrival times of Southwest Airlines

flights to Dallas on January 14th, 2021, from 6:00 AM to 2:00 PM

Time (T) [seconds]	**Frequency (f) [#]**
(0, 432]	94
(432, 864]	8
(864, 1296]	3
(1728, 2160]	1
(1296, 1728]	0

Table 2 - shows the frequency table using five bins. Most of the data lies between (0, 432].

Table 3

Data Table of Length of Major Rivers in North America

River (R)[#]	Length (l)[miles]
1	735
2	320
3	325
4	392
5	524
6	450
7	1459
8	135
9	465
10	600
11	330
12	336
13	280
14	315
15	870
16	906
17	202
18	329
19	290
20	1000
21	600
22	505
23	1450
24	840
25	1243
26	890
27	350
28	407
29	286
30	280
31	525
32	720
33	390
34	250
35	327

River (R)[#]	Length (l)[miles]
36	230
37	265
38	850
39	210
40	630
41	260
42	230
43	360
44	730
45	600
46	306
47	390
48	420
49	291
50	710
51	340
52	217
53	281
54	352
55	259
56	250
57	470
58	680
59	570
60	350
61	300
62	560
63	900
64	625
65	332
66	2348
67	1171
68	3710
69	2315
70	2533
71	780
72	280
73	410
74	460
75	260
76	255

River (R)[#]	Length (l)[miles]
77	431
78	350
79	760
80	618
81	338
82	981
83	1306
84	500
85	696
86	605
87	250
88	411
89	1054
90	735
91	233
92	435
93	490
94	310
95	460
96	383
97	375
98	1270
99	545
100	445
101	1885
102	380
103	300
104	380
105	377
106	425
107	276
108	210
109	800
110	420
111	350
112	360
113	538
114	1100
115	1205
116	314
117	237

River (R)[#]	Length (l)[miles]
118	610
119	360
120	540
121	1038
122	424
123	310
124	300
125	444
126	301
127	268
128	620
129	215
130	652
131	900
132	525
133	246
134	360
135	529
136	500
137	720
138	270
139	430
140	671
141	1770

Table 3 - shows the length of 141 rivers in North America. The table is not sorted.

Table 4

Data Table of Arrival times of Southwest Airlines

flights to Dallas on January 14th, 2021, from 6:00 AM to 2:00 PM

Arrival time (A) [HH:MM]	**Inter-Arrival Time (t) [HH:MM]**
6:19	NULL
6:55	0:36
7:02	0:07
7:04	0:02
7:25	0:21
7:30	0:05
7:30	0:00
7:35	0:05
7:44	0:09
7:48	0:04
7:54	0:06
7:58	0:04
8:05	0:07
8:07	0:02
8:11	0:04
8:16	0:05
8:17	0:01
8:24	0:07
8:28	0:04
8:29	0:01
8:33	0:04
8:36	0:03
8:38	0:02
8:40	0:02
8:52	0:12
8:54	0:02
8:56	0:02
8:56	0:00
9:01	0:05
9:13	0:12
9:13	0:00
9:16	0:03

Arrival time (A) [HH:MM]	**Inter-Arrival Time (t) [HH:MM]**
9:17	0:01
9:20	0:03
9:23	0:03
9:25	0:02
9:27	0:02
9:28	0:01
9:30	0:02
9:31	0:01
9:32	0:01
9:32	0:00
9:35	0:03
9:36	0:01
9:41	0:05
9:45	0:04
9:49	0:04
9:53	0:04
9:57	0:04
10:10	0:13
10:11	0:01
10:13	0:02
10:15	0:02
10:16	0:01
10:22	0:06
10:27	0:05
10:29	0:02
10:30	0:01
10:34	0:04
10:38	0:04
10:41	0:03
10:44	0:03
10:48	0:04
10:55	0:07
10:57	0:02
11:04	0:07
11:13	0:09
11:16	0:03
11:23	0:07
11:33	0:10
11:37	0:04

Arrival time (A) [HH:MM]	**Inter-Arrival Time (t) [HH:MM]**
11:39	0:02
11:42	0:03
11:43	0:01
11:46	0:03
11:55	0:09
11:58	0:03
11:59	0:01
12:01	0:02
12:02	0:01
12:06	0:04
12:11	0:05
12:18	0:07
12:23	0:05
12:25	0:02
12:29	0:04
12:29	0:00
12:31	0:02
12:34	0:03
12:39	0:05
12:40	0:01
12:46	0:06
12:47	0:01
12:48	0:01
12:56	0:08
13:00	0:04
13:04	0:04
13:04	0:00
13:05	0:01
13:23	0:18
13:24	0:01
13:24	0:00
13:27	0:03
13:46	0:19
13:47	0:01
13:53	0:06
13:56	0:03

Table 4 - shows the raw data. ‘t’ is calculated by subtracting the current time from the previous time. The first time is NULL, because there is no time before it.

Table 5

Descriptive Statistics Table for Length of Major Rivers in North America

Mean (μ) [miles]	**Std. Deviation (σ)[miles]**	**Quartile 1 (Q1) [miles]**	**Quartile 2 (Q2) [miles]**	**Quartile 3 (Q3) [miles]**
591	494	310	425	680

Table 5 data was calculated in python3 using the following equations on the sample:

$\mu = \Sigma\, x_i / N$, $\sigma = (\sqrt{\sum (x_i - \mu)^2 / N})$, $Q1 = (N + 1) * 1 / 4$, $Q2 = (N + 1) * 2 / 4$

$Q3 = (N + 1) * 3 / 4$. *Where N is the sample size*

Table 6

Descriptive Statistics Table for Inter-arrival times of Southwest Airlines

flights to Dallas on January 14th, 2021, from 6:00 AM to 2:00 PM

Mean (μ) [seconds]	**Std. Deviation (σ)[seconds]**	**Quartile 1 (Q1) [seconds]**	**Quartile 2 (Q2) [seconds]**	**Quartile 3 (Q3) [seconds]**
258.68	293.51	120.00	180.00	300.00

Table 6 data was calculated in python3 using the following equations on the sample:

$\mu = \Sigma\, x_i / N$, $\sigma = (\sqrt{\sum (x_i - \mu)^2 / N})$, $Q1 = (N + 1) * 1 / 4$, $Q2 = (N + 1) * 2 / 4$

$Q3 = (N + 1) * 3 / 4$. *Where N is the sample size*

Appendix II

Figures

Boxplot of Inter-arrival times of Southwest Airlines at Dallas Love Field on Jan.14th, 2021

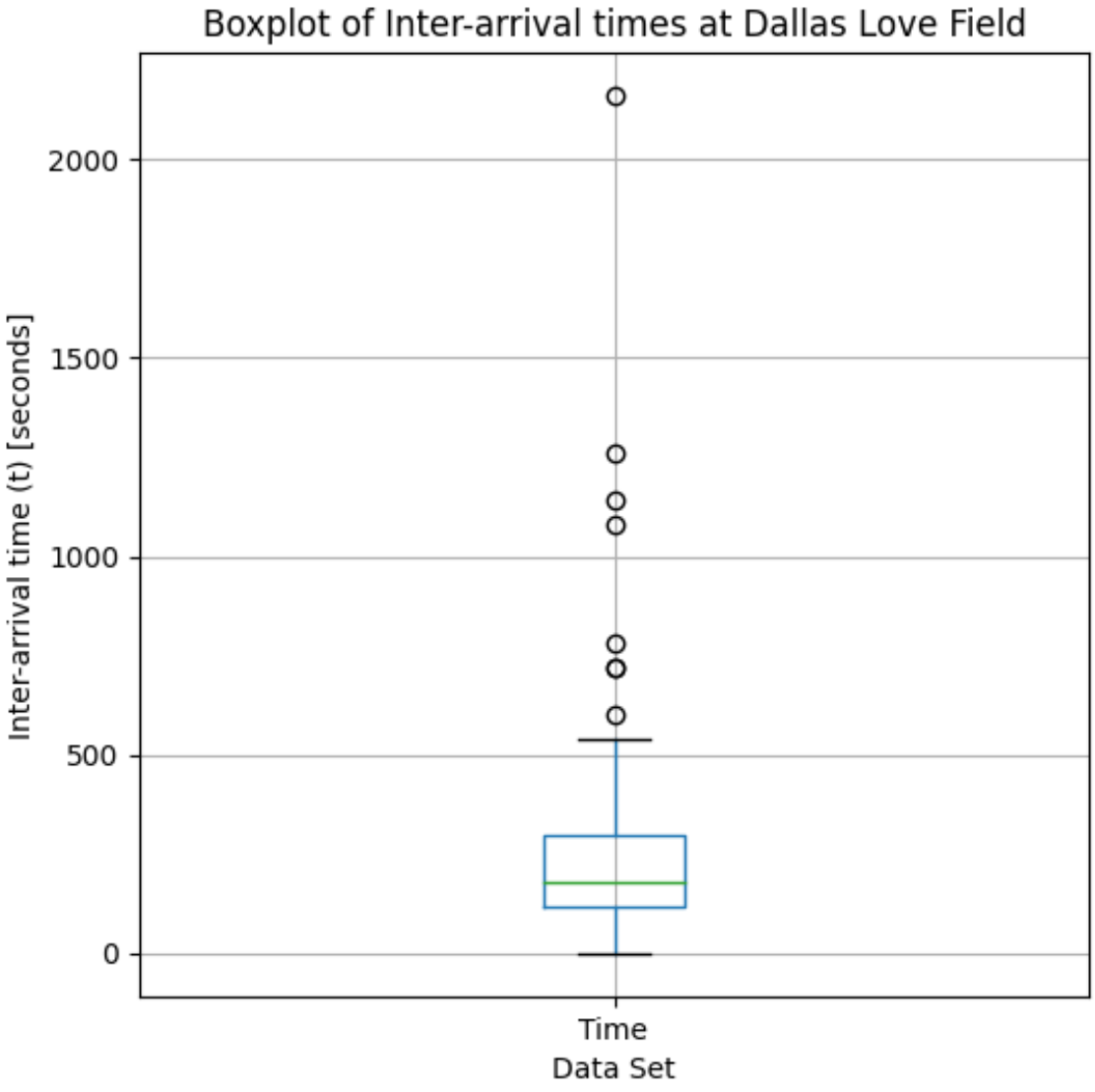

Figure 1 - There were seven outliers, all of which were over 500 seconds long, one of these being an inter-arrival time that was longer than 2000 seconds.

Histogram of Inter-arrival times of Southwest Airlines at Dallas Love Field on Jan.14th, 2021

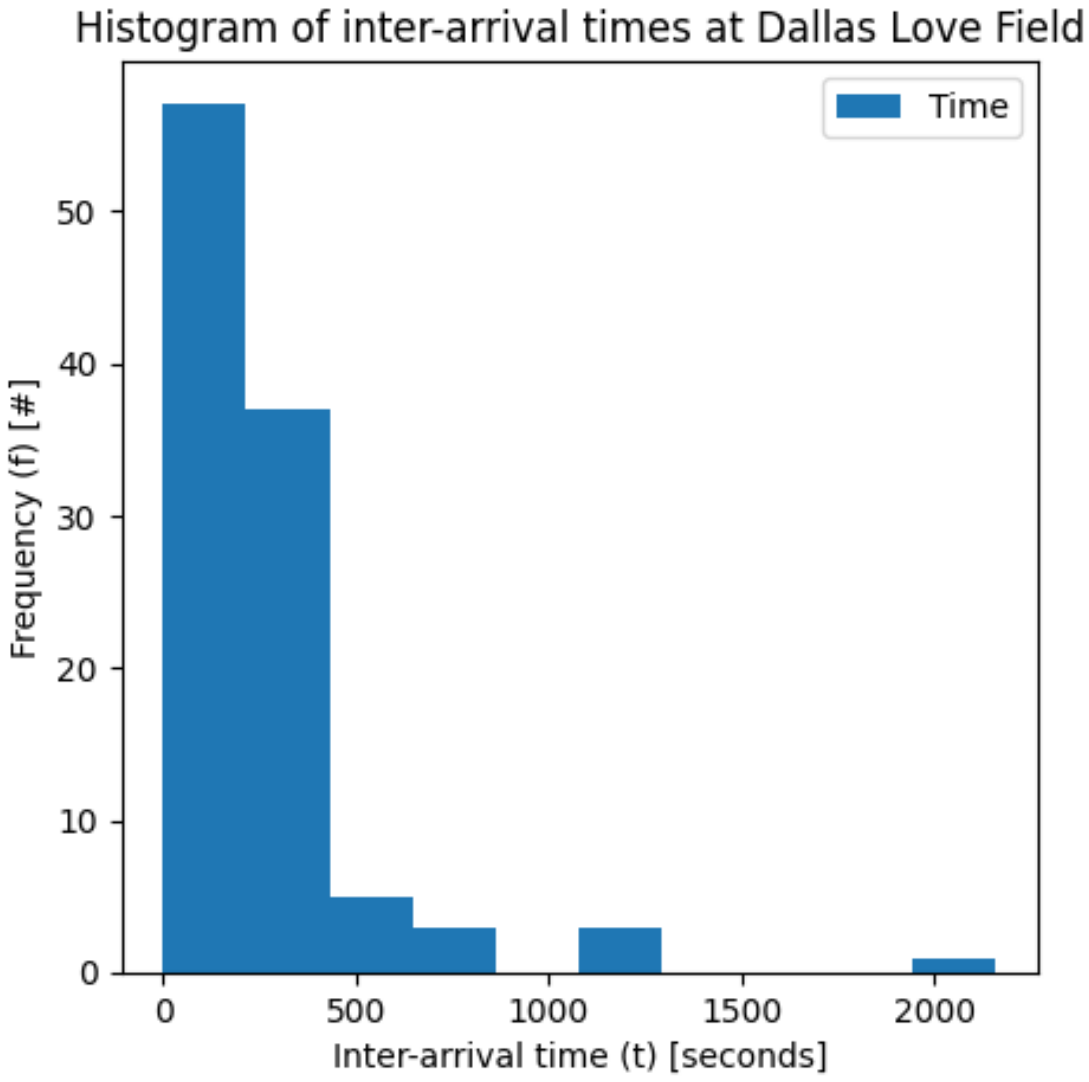

Figure 2 - The histogram of the inter-arrival times is right-skewed, meaning most of the inter-arrival times were on the 'shorter' side in length.

Boxplot of Length of Major Rivers in North America

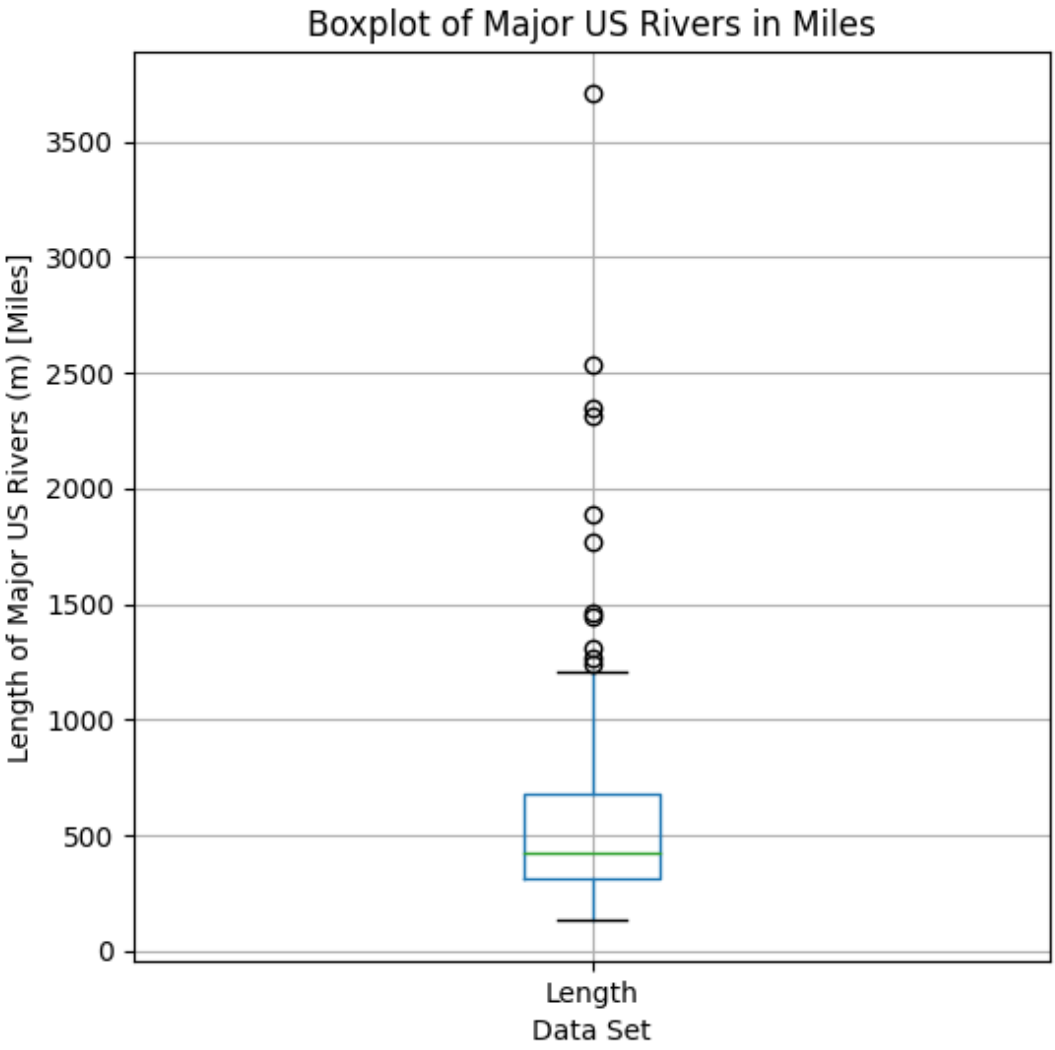

Figure 3 - There were 11 outliers, all of which were over 1200 miles long, one of these being a river that was longer than 3500 miles long (Figure 3)

Histogram of Length of Major Rivers in North America

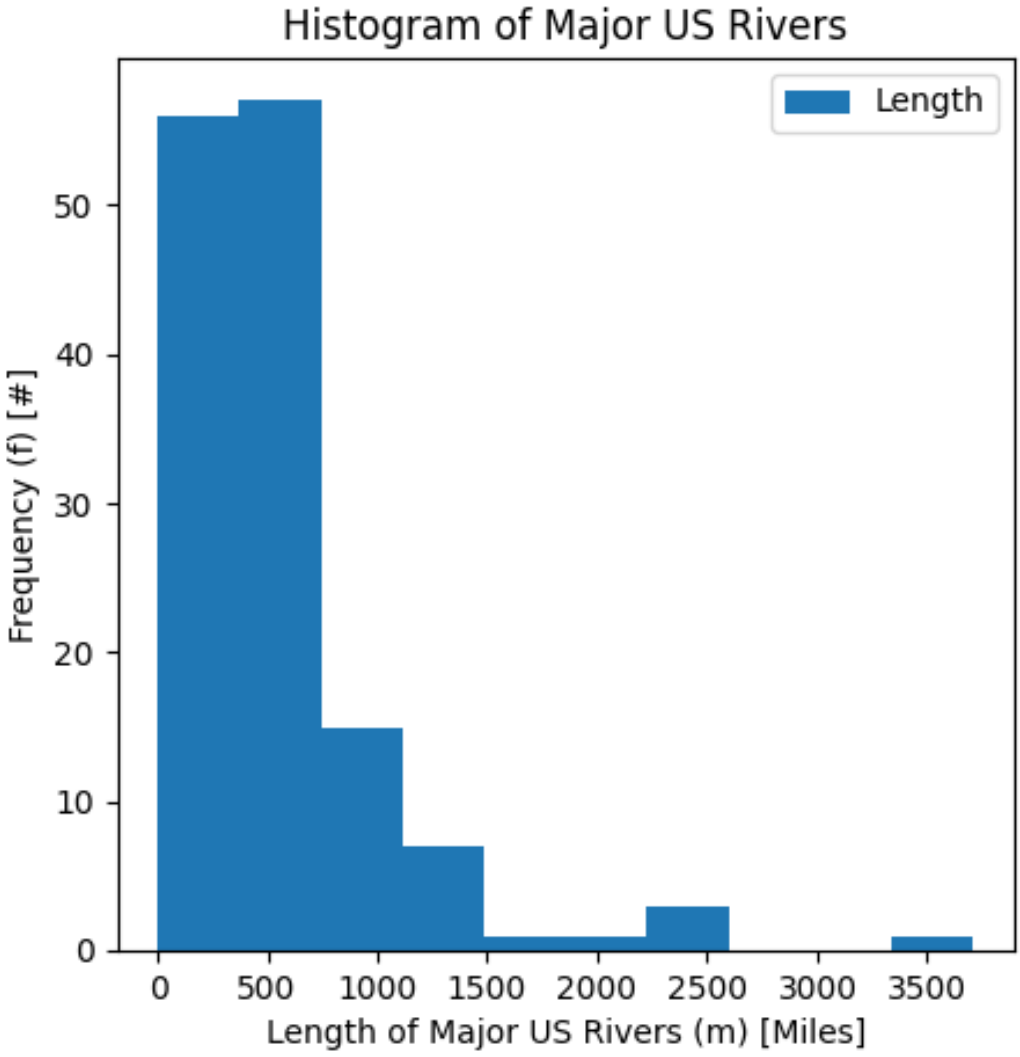

Figure 4 - The histogram of the rivers is right-skewed, meaning most of the rivers were on the 'shorter' side in length

Appendix III

Source Code

Main.py

```
import sys
import csv_to_list as ctl
import stats_calc as stats

original_stdout = sys.stdout

# 0 indicates csv file contains events
# 1 indicates csv file contains object-value
events_var = 0
objects_var = 1

# retrieve filename of csv
events_filename = input("Enter the name of the csv file
containing Event times: ")
event_graph_name = input(f"Enter the desired graph title for
{events_filename}: ")

objects_filename = input("Enter the name of the csv file
containing an Object | Value: ")
object_graph_name = input(f"Enter the desired graph titles for
{objects_filename}: ")

# get list of events or list of objects
events_list = ctl.parse_csv_file(0, events_filename)
objects_list = ctl.parse_csv_file(1, objects_filename)

# create object to store descriptive statistics
events = stats.DescriptiveStats(events_list, events_var)
objects = stats.DescriptiveStats(objects_list, objects_var)

# create graphs from the data
events.create_graphs(events.data, events_var, event_graph_name)
objects.create_graphs(objects.data, objects_var,
object_graph_name)

# save a file containing the descriptive statistics
with open('events_freq_table.txt', 'w') as f:
    sys.stdout = f  # Change the standard output to the file we
created.
    events.print_data()
    sys.stdout = original_stdout
```

```
with open('objects_freq_table.txt', 'w') as f:
    sys.stdout = f  # Change the standard output to the file we
created.
    objects.print_data()
    sys.stdout = original_stdout
```

csv_to_list.py

```
import csv


#################################################################
##
# FUNCTION: find_inter_arrival_time(time1, time2)
# Finds the difference between time2 and time1 (inter-arrival
time)
#   PARAMS:
#       time1 - the previous time listed in csv file
#       time2 - the current time listed in the csv file
#   RETURNS:
#      delta_t - difference in seconds of time2 and time1
#
#################################################################
##
def find_inter_arrival_time(time1, time2):
    # get the string time of event 1 and event 2
    # remove the colon from the time
    time1 = str(time1)
    time2 = str(time2)
    time1_str = time1.split(":")
    time2_str = time2.split(":")

    # turn the HH : MM : SS into integers
    h1 = int(time1_str[0])
    h2 = int(time2_str[0])
    m1 = int(time1_str[1])
    m2 = int(time2_str[1])
    s1 = int(time1_str[2])
    s2 = int(time2_str[2])

    # convert HH and MM into seconds
    h1 = h1 * 60 * 60
    h2 = h2 * 60 * 60
    m1 = m1 * 60
    m2 = m2 * 60
```

```
    # return the difference between time1 and time2
    t1_total = h1 + m1 + s1
    t2_total = h2 + m2 + s2
    delta_t = t2_total - t1_total

    return delta_t


#################################################################
#########
# FUNCTION: parse_csv_file(event_or_object, filename)
# Converts csv data into a list and returns this list
#   PARAMS:
#       event_or_object - 0 or 1 whether csv has events or
object | value
#       filename - name of file to be parsed
#   RETURNS:
#      list_of_events | list_of_objects - list of the data in
csv
#
#################################################################
#########
def parse_csv_file(event_or_object, filename):
    # the file to be parsed is an event
    if event_or_object == 0:

        # Parse the csv file
        with open(filename, "r") as f:
            csv_reader = csv.DictReader(f, fieldnames=["Event",
"Time"])
            list_of_events = list(csv_reader)

            # for each row, calculate the inter-arrival time and
            # input it into a new column called Time
            for i in range(len(list_of_events)):
                if i == 0:
                    continue

                diff = find_inter_arrival_time(list_of_events[i
- 1].get("Event"), list_of_events[i].get("Event"))
                list_of_events[i]["Time"] = diff

        return list_of_events

    # the file to be parsed is an object | value
    elif event_or_object == 1:
```

```
        # Parse the csv file
        with open(filename, "r") as f:
            csv_reader = csv.DictReader(f, fieldnames=["Object",
"Length"])
            list_of_objects = list(csv_reader)

            # convert each key value into an integer
            for obj in list_of_objects:
                for keys in obj:
                    obj[keys] = int(obj[keys])

        return list_of_objects
```

stats_calc.py

```
import matplotlib.pyplot as plt
import pandas as pd


#################################################################
#################
#
# Class - DescriptiveStats
# FUNCTION: Calculates mean, frequency table, std deviation, and
quartiles
# print_data - prints descriptive statistics and frequency table
to a .txt file
# create_graphs - creates a histogram and boxplot from data and
saves file
#
#################################################################
#################
class DescriptiveStats:
    ind = 0

    def __init__(self, data_list, event_or_object):

        self.event_or_object = event_or_object
        self.data = pd.DataFrame(data_list)

        # get the names of the columns of data
        self.column_names = list(data_list[0].keys())
        self.column_1 = self.column_names[0]
        self.column_2 = self.column_names[1]

        # calculate descriptive statistics
```

```
        self.frequency_table =
self.data[self.column_2].value_counts(bins=5)
        self.mean = self.data.mean(numeric_only=True)
        self.median = self.data.median(numeric_only=True)
        self.deviation = self.data.std(numeric_only=True)
        self.q1 = self.data[str(self.column_2)].quantile(0.25)
        self.q2 = self.data[str(self.column_2)].quantile(0.50)
        self.q3 = self.data[str(self.column_2)].quantile(0.75)

    ################################################
    #
    # FUNCTION : create_graphs
    #   Displays histogram and boxplot
    #   Saves the histogram and boxplot
    #
    # PARAMS:
    #   df - Dataframe object
    #   num - 0 for events, 1 for continuous samples
    #
    # RETURNS
    #   None
    #
    ################################################
    def create_graphs(self, df, num, graph_name):

        # if event times are being graphed
        if num == 0:

            boxplot_name = "Boxplot of Inter-arrival times of "
+ graph_name
            event_boxplot = df.boxplot(column=[self.column_2],
return_type="axes", figsize=(6, 6))
            event_boxplot.set_ylabel("Inter-arrival time (t)
[seconds]")
            event_boxplot.set_xlabel("Data Set")
            event_boxplot.set_title(boxplot_name)
            plt.gcf().savefig("events" + "_boxplot" + ".png",
format="png")

            histogram_name = "Histogram of inter-arrival times
of" + graph_name
            event_histogram =
df.plot.hist(column=[self.column_2], figsize=(5, 5),
title=histogram_name, bins=10)
            event_histogram.set_xlabel("Inter-arrival time (t)
[seconds]")
            event_histogram.set_ylabel("Frequency (f) [#]")
```

```
            plt.gcf().savefig("events" + "_histogram" + ".png",
format="png")

            plt.show()
            plt.close()

        # if a continuous random sample is being graphed
        else:

            boxplot_name = "Boxplot of " + graph_name
            event_boxplot = df.boxplot(column=[self.column_2],
return_type="axes", figsize=(6, 6))
            event_boxplot.set_ylabel("Length of Major US Rivers
(m) [Miles]")
            event_boxplot.set_xlabel("Data Set")
            event_boxplot.set_title(boxplot_name)
            plt.gcf().savefig("objects" + "_boxplot" + ".png",
format="png")

            histogram_name = "Histogram of " + graph_name
            event_histogram =
df.plot.hist(column=[self.column_2], figsize=(5, 5),
title=histogram_name, bins=10)
            event_histogram.set_xlabel("Length of Major US
Rivers (m) [Miles]")
            event_histogram.set_ylabel("Frequency (f) [#]")
            plt.gcf().savefig("objects" + "_histogram" + ".png",
format="png")

            plt.show()
            plt.close()

    ################################################
    #
    # FUNCTION : print_data
    #   Prints descriptive statistics to a file
    #
    # PARAMS:
    #   None
    # RETURNS
    #   None
    #
    ################################################
    def print_data(self):

        print("____________________FREQUENCY
TABLE_________________")
```

```
        print("Frequency Table")
        print(str(self.frequency_table))
        print("\n", end="")

print("____________________MEAN_____________________________")
        print(str(self.mean))
        print("\n", end="")
        print("____________________STD
DEVIATION_________________")
        print(str(self.deviation))
        print("\n", end="")

print("____________________QUARTILES_____________________")
        print(f"Q1: {self.q1}")
        print("\n", end="")
        print(f"Q1: {self.q2}")
        print("\n", end="")
        print(f"Q1: {self.q3}")
        print("\n", end="")

print("_____________________________________________________")
```