Question 12.1
Describe a situation or problem from your job, everyday life, current events, etc., for which a design of experiments approach would be appropriate.

Working in financial advising, a DOE approach would be appropriate if my firm wanted to do something such as evaluating the effectiveness of different advisory strategies or product recommendations. A focus could be finding the best combination of factors that produces the highest client satisfaction and portfolio returns over a year. Due to my firm having several advisors and investment strategies, factors could be:

1. Investment strategy type – aggressive, moderate, conservative
2. Communication frequency with client – weekly, monthly, quarterly, yearly
3. Recommendations – ETFs, mutual funds, stocks, bonds
4. Advisor experience – junior, senior

A DOE approach could find the best mix of these factors that would result in the best client satisfaction and portfolio performance.

Question 12.2
To determine the value of 10 different yes/no features to the market value of a house (large yard, solar roof, etc.), a real estate agent plans to survey 50 potential buyers, showing a fictitious house with different combinations of features. To reduce the survey size, the agent wants to show just 16 fictitious houses. Use R's FrF2 function (in the FrF2 package) to find a fractional factorial design for this experiment: what set of features should each of the 16 fictitious houses have? Note: the output of FrF2 is "1" (include) or "-1" (don't include) for each feature.

Using the FrF2 function in the FrF2 package, I found the following set of features each of the 16 houses should have:

```
set.seed(123)
library(FrF2)

design <- FrF2(nruns = 16, nfactors = 10)
print(design)
```

```
##     A  B  C  D  E  F  G  H  J  K
## 1  -1  1  1  1 -1 -1  1 -1  1 -1
## 2   1  1  1  1  1  1  1  1  1  1
## 3  -1  1 -1 -1 -1  1 -1  1  1 -1
```

```
## 4   1 -1  1  1 -1  1 -1  1 -1 -1
## 5   1 -1 -1  1 -1 -1  1  1  1  1
## 6   1 -1 -1 -1 -1 -1  1 -1 -1 -1
## 7   1 -1  1 -1 -1  1 -1 -1  1  1
## 8  -1 -1  1 -1  1 -1 -1  1  1 -1
## 9   1  1 -1 -1  1 -1 -1 -1  1  1
## 10  1  1 -1  1  1 -1 -1  1 -1 -1
## 11 -1  1  1 -1 -1 -1  1  1 -1  1
## 12 -1 -1 -1 -1  1  1  1  1 -1  1
## 13 -1  1 -1  1 -1  1 -1 -1 -1  1
## 14 -1 -1  1  1  1 -1 -1 -1 -1  1
## 15  1  1  1 -1  1  1  1 -1 -1 -1
## 16 -1 -1 -1  1  1  1  1 -1  1 -1
## class=design, type= FrF2
```

On the left-hand side of the matrix are the houses numbered from 1 through 16. On the top of the matrix are the features labeled A through K. To see which house should have which features, first look at the house number and then across its row. If the feature is a -1 (for example for house 1, A has a value of -1), then that feature should not be used for that house. If the feature is a 1 (for example for house 1, B has a value of 1), then that feature should be used for that house. Going down the rows, you can see which features should and should not be used for each individual house.

Question 13.1

For each of the following distributions, give an example of data that you would expect to follow this distribution (besides the examples already discussed in class).

a. Binomial

Given x amount of people contacted for a survey, finding the number of people who responded “yes” to it (successes)

b. Geometric

Shuffling a deck of cards and then pulling a card at random – if the ace of spades is pulled then it is a success. If it is a failure (not the ace of spades), the card is put back and the deck is reshuffled.

c. Poisson

The number of car accidents occurring at a particular intersection per month

d. Exponential

The time between two consecutive emails coming into an inbox

e. Weibull

The time until the cables or beams of a bridge need a repair

Question 13.2

In this problem you, can simulate a simplified airport security system at a busy airport. Passengers arrive according to a Poisson distribution with λ1 = 5 per minute (i.e., mean interarrival rate μ1 = 0.2 minutes) to the ID/boarding-pass check queue, where there are several servers who each have exponential service time with mean rate μ2 = 0.75 minutes. [Hint: model them as one block that has more than one resource.] After that, the passengers are assigned to the shortest of the several personal-check queues, where they go through the personal scanner (time is uniformly distributed between 0.5 minutes and 1 minute). Use the Arena software (PC users) or Python with SimPy (PC or Mac users) to build a simulation of the system, and then vary the number of ID/boarding-pass checkers and personal-check queues to determine how many are needed to keep average wait times below 15 minutes. [If you're using SimPy, or if you have access to a non-student version of Arena, you can use λ1 = 50 to simulate a busier airport.]

Below is my code using SimPy:

```
import simpy

import random

import numpy as np

import matplotlib.pyplot as plt


# Simulation parameters

lambda1 = 5  # Arrival rate (mean passengers per minute)

mu2 = 1 / 0.75  # Service rate for ID checkers (mean services per minute)

num_id_checkers = 3  # Number of ID checkers

num_personal_checkers = 3  # Number of personal checkers
```

```
sim_time = 120  # Total simulation time in minutes
wait_time_limit = 15  # Average wait time limit in minutes

# Data storage for wait times
wait_times = []

def id_boarding_pass_check(env, id_checkers, passenger_id):
    """Process for ID/boarding-pass check."""
    arrival_time = env.now
    with id_checkers.request() as request:
        yield request  # Wait for ID checker to become available
        service_time = np.random.exponential(1 / mu2)  # Exponential service time
        yield env.timeout(service_time)  # Simulate service time
        wait_time = env.now - arrival_time
        wait_times.append(wait_time)

def personal_check(env, personal_checkers, passenger_id):
    """Process for personal check."""
    arrival_time = env.now
    with personal_checkers.request() as request:
        yield request  # Wait for personal checker to become available
        service_time = random.uniform(0.5, 1.0)  # Uniform service time between 0.5 and 1 minute
        yield env.timeout(service_time)  # Simulate service time

def passenger(env, passenger_id, id_checkers, personal_checkers):
```

```
    """Passenger process for both ID check and personal check."""
    yield env.process(id_boarding_pass_check(env, id_checkers, passenger_id))  # ID check
    yield env.process(personal_check(env, personal_checkers, passenger_id))  # Personal check

def passenger_arrivals(env, id_checkers, personal_checkers):
    """Generate passengers arriving at the airport."""
    passenger_id = 0
    while True:
        yield env.timeout(np.random.exponential(1 / lambda1))  # Poisson arrival
        env.process(passenger(env, passenger_id, id_checkers, personal_checkers))
        passenger_id += 1

def run_simulation(num_id_checkers, num_personal_checkers):
    """Run the simulation and calculate average wait time."""
    env = simpy.Environment()
    id_checkers = simpy.Resource(env, capacity=num_id_checkers)
    personal_checkers = simpy.Resource(env, capacity=num_personal_checkers)

    env.process(passenger_arrivals(env, id_checkers, personal_checkers))  # Start passenger arrivals
    env.run(until=sim_time)  # Run the simulation

    return np.mean(wait_times)  # Return the average wait time

# Experiment with different numbers of checkers
id_checkers_range = range(1, 10)
```

```
personal_checkers_range = range(1, 10)

results = []


for id_checkers in id_checkers_range:

    for personal_checkers in personal_checkers_range:

        wait_time = run_simulation(id_checkers, personal_checkers)

        results.append((id_checkers, personal_checkers, wait_time))

        print(f'ID Checkers: {id_checkers}, Personal Checkers: {personal_checkers}, Avg Wait Time: {wait_time:.2f}')


# Filter results to find the configurations that keep wait times below the limit

valid_results = [(id_checkers, personal_checkers, wait_time) for id_checkers, personal_checkers, wait_time in results if wait_time < wait_time_limit]


# Plotting the results

plt.figure(figsize=(10, 6))

for id_checkers in id_checkers_range:

    avg_wait_times = [result[2] for result in results if result[0] == id_checkers]

    plt.plot(personal_checkers_range, avg_wait_times, marker='o', label=f'ID Checkers: {id_checkers}')


plt.axhline(y=wait_time_limit, color='r', linestyle='--', label='Wait Time Limit (15 min)')

plt.title('Average Wait Time vs. Number of Personal Checkers')

plt.xlabel('Number of Personal Checkers')

plt.ylabel('Average Wait Time (minutes)')

plt.xticks(personal_checkers_range)

plt.legend()
```

```
plt.grid()
plt.show()
```

This code resulted in the following results:

ID Checkers: 1, Personal Checkers: 1, Avg Wait Time: 43.97

ID Checkers: 1, Personal Checkers: 2, Avg Wait Time: 44.37

ID Checkers: 1, Personal Checkers: 3, Avg Wait Time: 45.39

ID Checkers: 1, Personal Checkers: 4, Avg Wait Time: 46.47

ID Checkers: 1, Personal Checkers: 5, Avg Wait Time: 45.56

ID Checkers: 1, Personal Checkers: 6, Avg Wait Time: 45.04

ID Checkers: 1, Personal Checkers: 7, Avg Wait Time: 44.17

ID Checkers: 1, Personal Checkers: 8, Avg Wait Time: 44.62

ID Checkers: 1, Personal Checkers: 9, Avg Wait Time: 44.28

ID Checkers: 2, Personal Checkers: 1, Avg Wait Time: 40.52

ID Checkers: 2, Personal Checkers: 2, Avg Wait Time: 38.27

ID Checkers: 2, Personal Checkers: 3, Avg Wait Time: 36.63

ID Checkers: 2, Personal Checkers: 4, Avg Wait Time: 35.06

ID Checkers: 2, Personal Checkers: 5, Avg Wait Time: 34.30

ID Checkers: 2, Personal Checkers: 6, Avg Wait Time: 33.14

ID Checkers: 2, Personal Checkers: 7, Avg Wait Time: 32.72

ID Checkers: 2, Personal Checkers: 8, Avg Wait Time: 32.46

ID Checkers: 2, Personal Checkers: 9, Avg Wait Time: 32.32

ID Checkers: 3, Personal Checkers: 1, Avg Wait Time: 30.24

ID Checkers: 3, Personal Checkers: 2, Avg Wait Time: 28.64

ID Checkers: 3, Personal Checkers: 3, Avg Wait Time: 27.54

ID Checkers: 3, Personal Checkers: 4, Avg Wait Time: 26.32

ID Checkers: 3, Personal Checkers: 5, Avg Wait Time: 25.63

ID Checkers: 3, Personal Checkers: 6, Avg Wait Time: 24.69

ID Checkers: 3, Personal Checkers: 7, Avg Wait Time: 24.09

ID Checkers: 3, Personal Checkers: 8, Avg Wait Time: 23.32

ID Checkers: 3, Personal Checkers: 9, Avg Wait Time: 22.97

ID Checkers: 4, Personal Checkers: 1, Avg Wait Time: 21.65

ID Checkers: 4, Personal Checkers: 2, Avg Wait Time: 20.49

ID Checkers: 4, Personal Checkers: 3, Avg Wait Time: 19.41

ID Checkers: 4, Personal Checkers: 4, Avg Wait Time: 18.47

ID Checkers: 4, Personal Checkers: 5, Avg Wait Time: 17.64

ID Checkers: 4, Personal Checkers: 6, Avg Wait Time: 17.00

ID Checkers: 4, Personal Checkers: 7, Avg Wait Time: 16.35

ID Checkers: 4, Personal Checkers: 8, Avg Wait Time: 15.75

ID Checkers: 4, Personal Checkers: 9, Avg Wait Time: 15.22

ID Checkers: 5, Personal Checkers: 1, Avg Wait Time: 14.65

ID Checkers: 5, Personal Checkers: 2, Avg Wait Time: 14.12

ID Checkers: 5, Personal Checkers: 3, Avg Wait Time: 13.65

ID Checkers: 5, Personal Checkers: 4, Avg Wait Time: 13.15

ID Checkers: 5, Personal Checkers: 5, Avg Wait Time: 12.72

ID Checkers: 5, Personal Checkers: 6, Avg Wait Time: 12.31

ID Checkers: 5, Personal Checkers: 7, Avg Wait Time: 11.94

ID Checkers: 5, Personal Checkers: 8, Avg Wait Time: 11.58

ID Checkers: 5, Personal Checkers: 9, Avg Wait Time: 11.27

ID Checkers: 6, Personal Checkers: 1, Avg Wait Time: 10.94

ID Checkers: 6, Personal Checkers: 2, Avg Wait Time: 10.64

ID Checkers: 6, Personal Checkers: 3, Avg Wait Time: 10.35

ID Checkers: 6, Personal Checkers: 4, Avg Wait Time: 10.10

ID Checkers: 6, Personal Checkers: 5, Avg Wait Time: 9.86

ID Checkers: 6, Personal Checkers: 6, Avg Wait Time: 9.63

ID Checkers: 6, Personal Checkers: 7, Avg Wait Time: 9.38

ID Checkers: 6, Personal Checkers: 8, Avg Wait Time: 9.16

ID Checkers: 6, Personal Checkers: 9, Avg Wait Time: 8.96

ID Checkers: 7, Personal Checkers: 1, Avg Wait Time: 8.75

ID Checkers: 7, Personal Checkers: 2, Avg Wait Time: 8.56

ID Checkers: 7, Personal Checkers: 3, Avg Wait Time: 8.37

ID Checkers: 7, Personal Checkers: 4, Avg Wait Time: 8.21

ID Checkers: 7, Personal Checkers: 5, Avg Wait Time: 8.06

ID Checkers: 7, Personal Checkers: 6, Avg Wait Time: 7.91

ID Checkers: 7, Personal Checkers: 7, Avg Wait Time: 7.76

ID Checkers: 7, Personal Checkers: 8, Avg Wait Time: 7.63

ID Checkers: 7, Personal Checkers: 9, Avg Wait Time: 7.50

ID Checkers: 8, Personal Checkers: 1, Avg Wait Time: 7.37

ID Checkers: 8, Personal Checkers: 2, Avg Wait Time: 7.24

ID Checkers: 8, Personal Checkers: 3, Avg Wait Time: 7.12

ID Checkers: 8, Personal Checkers: 4, Avg Wait Time: 7.01

ID Checkers: 8, Personal Checkers: 5, Avg Wait Time: 6.89

ID Checkers: 8, Personal Checkers: 6, Avg Wait Time: 6.77

ID Checkers: 8, Personal Checkers: 7, Avg Wait Time: 6.67

ID Checkers: 8, Personal Checkers: 8, Avg Wait Time: 6.57

ID Checkers: 8, Personal Checkers: 9, Avg Wait Time: 6.47

ID Checkers: 9, Personal Checkers: 1, Avg Wait Time: 6.38

ID Checkers: 9, Personal Checkers: 2, Avg Wait Time: 6.28

ID Checkers: 9, Personal Checkers: 3, Avg Wait Time: 6.19

ID Checkers: 9, Personal Checkers: 4, Avg Wait Time: 6.12

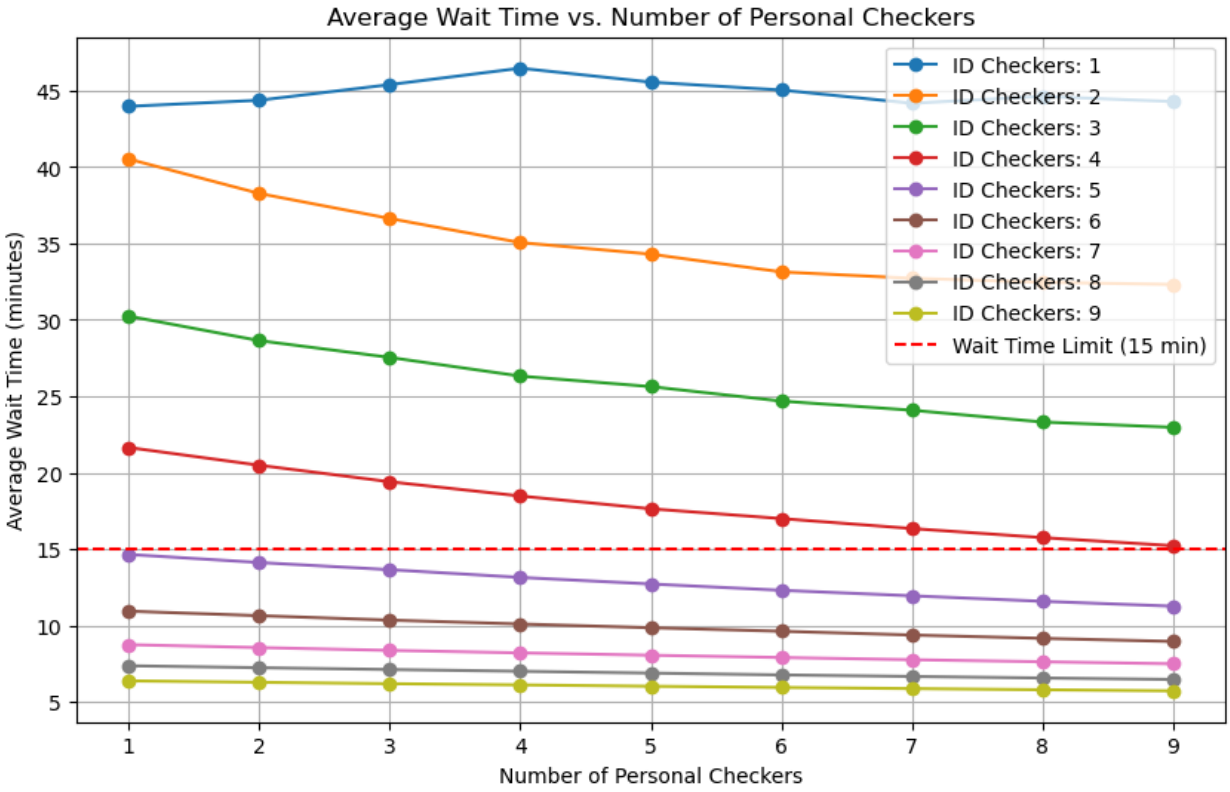
ID Checkers: 9, Personal Checkers: 5, Avg Wait Time: 6.03

ID Checkers: 9, Personal Checkers: 6, Avg Wait Time: 5.95

ID Checkers: 9, Personal Checkers: 7, Avg Wait Time: 5.87

ID Checkers: 9, Personal Checkers: 8, Avg Wait Time: 5.80

ID Checkers: 9, Personal Checkers: 9, Avg Wait Time: 5.72



After reading the question and mocking up the initial simulation, my main goal was to do what the graph shows above. I wanted to test different numbers of personal checkers with a set ID checker, then keep doing that but increasing the number of ID checkers. I do not want to walk through the code, but rather want to talk about the findings. As shown by the graph, for each amount of ID checkers, by increasing the number of personal checkers it made the average wait time decrease by each addition of a personal checker (other than

when there was just 1 ID checker which I found interesting). It also shows, as I assumed would be the result, that with the addition of one more ID checker each time, the average wait time decreased compared to the previous amount. To keep the wait times below 15 minutes, a minimum of 5 ID checkers is going to be needed. 4 ID checkers with 9 personal checkers is almost right at 15 minutes (15:22). 5 ID checkers to 9 ID checkers are all under 15 minutes even with 1 personal checker. Comparing 7 ID checkers to 9 ID checkers, there really is not much of a difference in average wait times per number of personal checkers. There really is not even that much of a difference in wait times for if there is 1 personal checker vs if there is 9 of them for those 7 to 9 ID checker values. This means that after 7 ID checkers, the marginal benefit of adding another one is pretty low.