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.

The design of experiments approach is highly relevant whenever you're dealing with unknowns that require a non-hacky solution. Currently, I'm working on developing a data pipeline that uses an LLM to aggregate emails from Google scholar papers. The design of experiments approach can be utilized to verify whether the produced emails are quality, in a scientific method. To do so, it would be important to select controls in my email aggregation process, and email distribution process. Emails should be consistently delivered throughout the same timeframes, to evenly distributed locations. Otherwise, it may be unclear whether response patterns are strictly based upon email contents. Our sample sizes for this problem will have to be high to mitigate all of the randomness that could possibly occur as that will likely prevent us from making any statistically significant claims.

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.

My solution to this problem assumes that the first four factors (A, B, C, and D) are free, while the other six factors are generated in the schema: E = A*B, F = A*C, G = A*D, H = B*C, I = B*D, and J = C*D. Using the FrF2 package, we can do 16 runs and 10 factors within a single line of code. I displayed the associated output using a webapp built using the R package 'shiny'. Each row of data represents a hypothetical house, with each arbitrary feature being given a unique letter for identification. Below is a matrix wherein 1 indicates that the feature should be included within the fictional house, and -1 indicating that a feature should not be included.

Georgia Tech

Fractional Factorial Design for House Features

Each row represents a fictitious house. The columns (A to J) represent 10 yes/no features. '1' indicates the feature is included; '-1' indicates it is not included.

A	B	C	D	E	F	G	H	J	K
-1	1	-1	-1	-1	1	1	-1	-1	1
1	-1	-1	1	-1	-1	1	1	-1	-1
-1	-1	1	1	1	-1	-1	-1	-1	1
-1	1	1	1	-1	-1	-1	1	1	1
-1	1	1	-1	-1	-1	1	1	-1	-1
-1	-1	1	-1	1	-1	1	-1	1	-1
-1	-1	-1	-1	1	1	1	1	1	1
1	-1	-1	-1	-1	-1	-1	1	1	1
1	-1	1	-1	-1	1	-1	-1	1	-1
1	1	-1	1	1	-1	1	-1	1	-1
1	-1	1	1	-1	1	1	-1	-1	1
1	1	-1	-1	1	-1	-1	-1	-1	1
-1	-1	-1	1	1	1	-1	1	-1	-1
-1	1	-1	1	-1	1	-1	-1	1	-1
1	1	1	-1	1	1	-1	1	-1	-1
1	1	1	1	1	1	1	1	1	1

I.

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**
   Coin flips where the Y axis represents events, and the X axis represents % of heads flipped
b. **Geometric**
   Modeling average retrials to complete an assignment with a passing score
c. **Poisson**
   Audience retention
d. **Exponential**
   Positive feedback loops such as modeling average time between contractions during birth
e. **Weibull**
   Estimating when cars parked streetside will deteriorate to natural elements

Question 13.2

II.

In this problem you, can simulate a simplified airport security system at a busy airport. Passengers arrive according to a Poisson distribution with $\lambda_1 = 5$ per minute (i.e., mean interarrival rate $\mu_1 = 0.2$ minutes) to the ID/boarding-pass check queue, where there are several servers who each have exponential service time with mean rate $\mu_2 = 0.75$ 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 $\lambda_1 = 50$ to simulate a busier airport.]

My solution to this problem was a shiny webapp that used data from the frontend to run a Simply simulation of these events iteratively to find the ideal number of ID/boarding-pass checkers and personal check queues. At the top of the program, users are able to run manual simulations, with the default values being the values provided in this question, along with the ideal values for checkers. The bottom of the program uses matplotlib to display charts that summarize the findings of iteratively run simulations. It was found that 3 checkers in both categories nears the 15 minute average wait time, but that 4 checkers successfully crosses this threshold. After this point, there are diminishing returns. Therefore, it would be best for the airport security system to employ 4 checkers with 4 personal-check queues.

Airport Security System Simulation

Number of ID/Boarding-Pass Checkers:

4

Number of Personal-Check Queues:

4

Simulation Time (minutes):

100

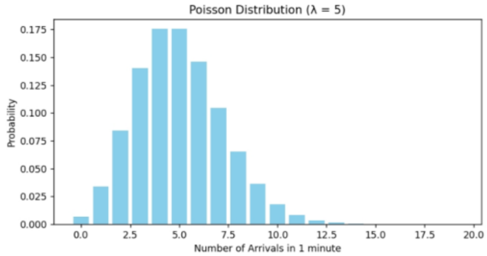
Passenger Arrival Rate (per minute):

5

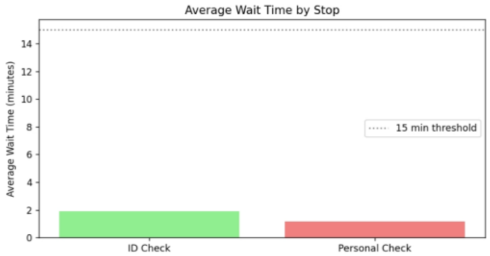
Run Simulation

Simulation complete over 487 passengers. Average total time in system: 4.57 minutes. Average wait time for ID check: 1.89 minutes. Average wait time for Personal check: 1.16 minutes. Goal: Average wait time should be below 15 minutes.

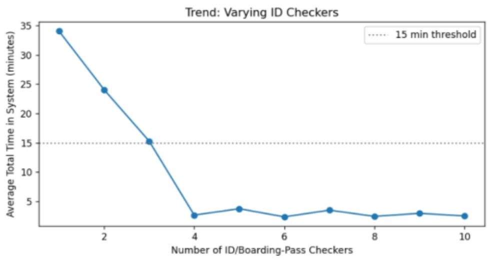
Poisson Arrival Distribution



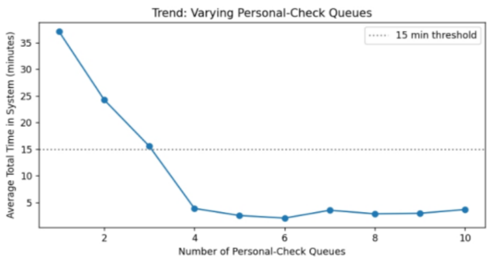
Average Wait Times at Each Stop



Trend Analysis: Varying ID/Boarding-Pass Checkers



Trend Analysis: Varying Personal-Check Queues



Code for this homework utilized ChatGPT as a resource.