Professor Effects on Student Trajectories

A Statistical Data Analysis in R

Ethan Tucker

Advisor Dr. Eric Nordmoe

Kalamazoo College Math Department

A paper submitted in partial fulfillment of the requirements for the degree of Bachelor of Arts at Kalamazoo College.

April 2021

Contents

List of Tables

List of Figures

Preface and Attributions

This project would not have been possible without many of the wonderful people in my life. I would like to thank my parents for encouraging me to be curious and supporting me both physically and emotionally throughout these last five years. My mother, who is currently sitting across from me playing a game on her phone at 5:45 AM while I finish this paper. My father, who read through seventy pages of improvised statistical jargon to look for grammatical errors. I would like to give a massive thanks to my SIP advisor Dr. Nordmoe for both inspiring a love of statistics and for being incredibly patient and understanding during this process. Dr. Nordmoe stands out as a prime example of an educator that changes trajectories. Thank-yous to the Math and Physics departments of K College for helping me discover my love of numbers. I would like to thank my employer Thompson Tutoring for allowing me to discover a love of teaching and pedagogy. I give a special thank you to my advisor Robin Rank who called me every Monday to check in on me through thick and thin, and stuck with me for five entire years.

Abstract

Parameters quantifying the quality of instruction can be difficult to come across. One commonly used measure comes from course evaluations, which are largely subjective reviews that pertain to personality in conjunction with quality of teaching. Being optional surveys, course evaluations may not tell a full story. According to institutional research at Marquette University 83% of students offered in-class time to complete course evaluations did so, and only 59% not offered class time completed surveys. (1) Another common metric comes from online faculty review boards like https://www.ratemyprofessors.com/. According to a 2008 and 2009 study, these reviews are highly correlated with official institutional reviews. (2,3) As such, they may be unreliable for the same reasons. Student outcome data can serve as a good indicator of faculty effects. In this paper we construct two statistics from a large such dataset from the Math and Chemistry departments of Oakland Community College. We will observe the effects that faculty have on their students' final cumulative GPAs, and analyse the distribution of withdraw rates between professors.

Introduction

I started working as a math tutor for a local company called Thompson Tutoring back in the fall of 2019. At first it was just a way to pay rent and keep myself fed, but I came to enjoy it and care about the success of my students. I have about five students that have stuck with me since the beginning, each of whom I see once a week for an hour. Oftentimes I find myself wondering how much of an effect I really have on their mathematics education. As a tutor I am generally employed by a family in two settings:

1. A student is struggling in their current math course. In this scenario my role is to be the remedial instructor, trying to fill in the gaps in my student's knowledge while simultaneously preparing them for the next test.

2. A wealthy family is looking to get their students ahead in mathematics.

It is very difficult to measure the effect I have had on my students for a variety of reasons, primary among them small sample size and no access to quantitative data. For my SIP I still wanted to combine my newfound addiction to data analysis with education, so Dr. Nordmoe sent me the link to a datset that Professor Andrew Eckstrom of Oakland Community College (hereafter OCC) had posted to a messageboard. The source data contain 139228 observations in 5 variables - "Student Random ID", "Course Code", "Grade", "Faculty Random ID", and "Semester". Each row corresponds to one course taken by one student in one semester between the fall of 2010 and the winter of 2017 in

the mathematics and chemistry departments of Oakland Community College. "Student Random ID" and "Faculty Random ID" are as expected randomly generated integers than track a given student or faculty member through the file. In total, the file tracks 66164 students and 268 faculty. My SIP is in half a statistical analysis on new variables I create from Dr. Eckstrom's dataset, and in half learning the requisite technology to perform such an analysis.

To explain the goal of my project, we will need to briefly diverge into a personal tangent. I came into Kalamazoo College thinking I would follow the pre-med track and thus need to major in chemistry or biology. Four and a half years later, I am graduating with two majors, among them neither chemistry nor biology. I only took one course each in those department because I did not click with either professor. In the math and physics departments I found faculty that were both more personable and frankly better at teaching their subject. My majors in math and physics developed as I continued to take courses with these faculty. The central question I pose in my SIP is this: how does the first faculty member a student encounters in a department influence a student's outcome?

Professor Eckstrom's data tracks students through time. In conjunction with grading policies and course credits obtainable through OCC's website we can track each student's cumulative GPA through their stay in the Math and Chemistry departments. We will build linear mixed-effects models stratified to avoid some confounding variables to determine an expected final cumulative GPA for each student that passes a few conditions, then determine the

difference between observed and expected final cumulative GPA. I call this statistic ΔC, short for difference in cumulative GPA. Where C_O is a student's observed final cumulative GPA and C_E is the final cumulative GPA predicted by the linear mixed-effects model:

$$\Delta C = C_O - C_E \tag{1}$$

We will then take the mean ΔC among all students that started with a certain faculty, which can be thought of as the average benefit to final cumulative GPA a student that started with the professor had with respect to the mean student. It is possible to assign **any** faculty "points" for only the change in cumulative GPA that comes after, but the weights are unintuitive due to future courses having less effect on change in cumulative GPA as opposed to the extant value (beyond the second course). After running significance tests we will create faculty rankings based on their overall mean ΔC, and their mean ΔC per course they taught. There will be pretty pictures along the way.

One of the conditions required to create the ΔC statistic is that students take multiple terms in the math and/or chemistry departments. This happens to be a pretty large requirement, especially for a community college such as OCC. I have pulled public data from DataUSA regarding OCC's enrollment statistics for all years available, which ended up being the years 2012 through 2019. (4) The following tables and graph summarise this enrollment data.

Table 1: OCC Fall Enrollment by Year, 2012-2019

Year	Full Time	Part Time	Total
2012	8662	18634	27296
2013	8058	18346	26404
2014	7012	17019	24031
2015	5247	15162	20409
2016	5104	13819	18923
2017	4373	12743	17116
2018	3876	11535	15411
2019	3776	11435	15211

Table 2: OCC Mean Enrollment Statistics

Mean Full Time	Mean Part Time	Mean Total
5764	14837	20600

Both full and part time enrollment monotonically declines over the reported years. Unfortunately, professor Eckstrom's dataset begins in the year 2010, so there are two relevant years we do not have data regarding enrollment. Regardless, between 2012 and 2019 on average only 27.978% of students were enrolled full time. (4) According to the National Student Clearinghouse, community college dropout rate is a function of time enrolled - therefore part time

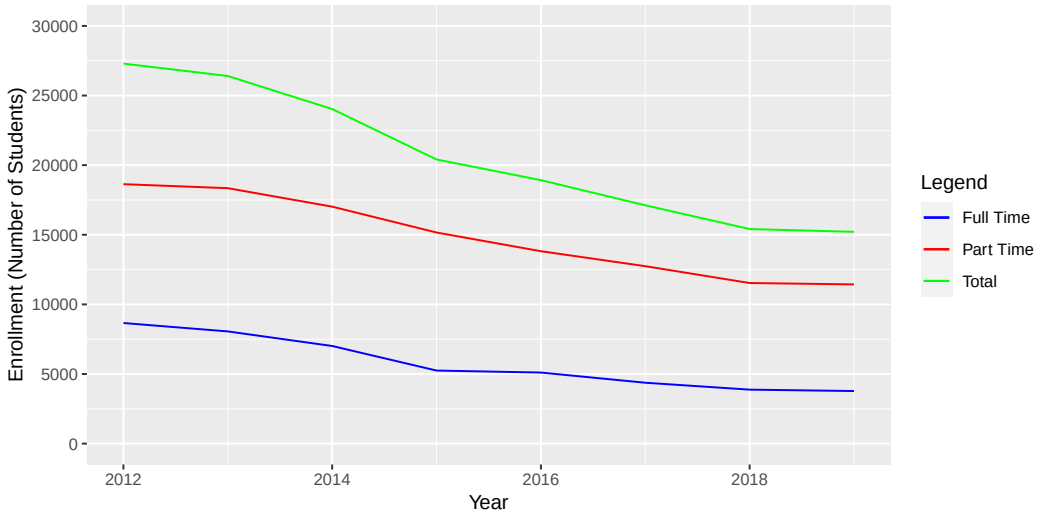

Figure 1: OCC Enrollments By Year for Full Time, Part Time, and Total Student Body

students are less likely to take multiple courses in the same department due to a lower course load per term. (5) As we will see, requiring students to have two *graded* terms drastically reduces the sample size, but still allows for many faculty to be evaluated by the ΔC metric.

Another difficulty arises in light of a second filtration condition - students must not withdraw from the first course they take in the math or chemistry departments. It would be possible to start the models at the first graded term, but if a student withdraws from a course then takes it again, their grade may be artificially inflated and thus not serve as an honest intercept. As a result, the expectation on ΔC becomes negative rather than zero. Requiring students to receive a standard letter grade ensures that the linear mixed effects models have a consistent starting point, and that individual teachers are evaluated

fairly. What if a certain faculty member had a high withdraw rate due to their course being hard, but prepared their students well for future courses and thus also had a high mean ΔC? Simply put, is there a correlation between a faculty's mean ΔC and their withdraw rate? ΔC is blind to any effect outside the realm of cumulative GPA, so we must introduce a new statistic to capture variance in withdraw rates. Fortunately, the calculation of withdraw rate for individual faculty does not require filtration so the sample sizes will be much larger and thus the results much more accurate. As with the ΔC rankings, we will construct rankings using the difference between the overall withdraw rate and a faculty's observed rate for 1 the overall case, and by course taught. To be clear, where W_O is a faculty's mean withdraw rate and W_E is the appropriate expected rate, this statistic will be calculated as:

$$\Delta W = W_E - W_O \tag{2}$$

Note that we have reversed the direction of difference with respect to ΔC. We have chosen orientations such that positive Δ values correspond to "good" effects for ease of human interpretation. Additionally, note that while an observation on ΔC corresponds to one student, an observation on ΔW is specific to a faculty member. To reiterate, we will average over the ΔC values among those students that started with a professor to obtain a value unique to that faculty.

When I began this project, I had no idea what I was doing. For the first month or so I was using some atrocious C++ inspired hard-coding methods,

one remnant of which I have included commented out in the code that follows this paper. When I showed Dr. Nordmoe one copious script I had created to determine the total number of students/faculty in the data, he told me to call the unique() function. In approximately twenty seconds the process I had spent several hours on had been recreated. Moreover, the runtime was spectacularly improved. I was then told to read chapter five of *R For Data Science* by Hadley Wickham and Garrett Grolemund. Chapter five is a guide to the R package dplyr, one of the many packages from the tidyverse I used in the construction of this project. (6) As it happens, I didn't need to reinvent the wheel. A lot of smart people have built a lot of nice functions to make R a phenomenal work environment for data analysis. *R For Data Science* was immensely helpful to my project, and was my primary guide to working with tidyverse functions beyone Dr. Nordmoe's data science course. The full book can be found online for free at https://r4ds.had.co.nz/. I would like to relay the author's wishes for support of the Kakapo parrot, a critically endangered New Zealand native. If a reader so desires, the New Zealand government has a fundraiser at https://www.doc.govt.nz/kakapo-donate.

In order to help learn skills for this project, I enrolled in MAT-295 (Introduction to Data Science) at K. The course used the same free textbook that I read for this project, and so is an excellent offering for all students. MAT-295 covered tidying and presenting data, text manipulation by means of regular expressions, manipulation of categorical variables, basic date/time conversion methods, creation of simple linear models, and a brief introduction to programming in R among other subjects. Course material does not include any

mathematical theory such as probability or mathematical statistics, though the math department offers courses in those subjects in the Winter/Spring each year. Introduction to Data Science does not require probability or math stats for enrollment, but statistics in particular is a powerful weapon in data analysis that we will be making heavy use of in this paper. MAT-295 was enormously helpful in giving me practice manipulating and presenting data using the tidyverse approach, which comprises at least half of the following project.

Statistical Background

Probability Theory

Probability theory is the backbone of any statistical method. In particular, the central limit theorem is an amazing piece of technology that guarantees when sufficiently many nicely behaved i.i.d. random variables are added, their sum will have a Gaussian distribution. (7) Where the i.i.d. random variables X_i with population mean μ and standard deviation σ:

$$\lim_{k\to\infty} \sum_{i=1}^{k} X_i \; N(k\mu, \frac{\sigma^2}{k})(8) \tag{3}$$

Clearly a perfectly normal distribution would require the sum of infinitely many random variables. In reality it is quite tedious to attempt to collect an infinite number of values to add together, so it is common practice to set a lower bound of observations before data is considered approximately normal. (8) This bound is often considered to be between 30 and 40 depending on the author, but this value can decrease arbitrarily if the random variables themselves have normal distribution. (8, 9, 10) Unfortunately, final cumulative GPA is inherently discrete, and withdraw rate is binomial. As such we must use an approximation to the central limit theorem. Where the random variables X_1 through X_n are i.i.d. each with sample mean $\bar{x}$ and standard error $s = \frac{\sigma_x}{\sqrt{n}}$, (8)

$$X = \sum_{i=1}^{n} X_i \sim N(n\bar{x}, s^2) \tag{4}$$

Like the Gaussian distribution, Student's t distribution is heavily used in significance testing. (10) For a sample of N i.i.d. random variables with sample mean $\bar{x}$, a-priori population mean μ, and sample standard deviation σ_x, the t statistic is defined to be: (10)

$$t = \frac{\bar{x} - \mu}{\sigma_x / \sqrt{N}} \tag{5}$$

The t distribution is designed to enable the analysis of small sample size, in accordance to a few conditions. These are: (10)

1. Data must be collected from random variables that come from a continuous scale. This assumption is relaxed as the number of possible values increases. As I will show later, both ΔC and ΔW have sufficient possible values per case to pass this condition.

2. Data must be collected by way of a simple random sample. From what I can tell, Professor Eckstrom's file contains a census, and so is immune to this requirement.

3. Random variables must resemble the normal random variable under combination. We will see in the case study that while the empirical distribution of the mean ΔC statistic is very close to normal, the empirical distribution of the ΔW statistic is not. As such, the ΔW statistic is not

well modeled by Student's t distribution.

4. The sample must have a reasonable sample size. The central t-statistic is undefined for $N = 1$, and gradually converges to the standard normal z-statistic as N increases. The smaller the original sample size, the higher the requisite t-value for significance. The only requirement on sample size is that there are enough data points to calculate the sample standard deviation. For safety, in my analysis I will require a minimum sample size of 8 students for calculation of both ΔC and ΔW. The reason for the number eight will be discussed after introducing the bootstrap. (13)

Significance Testing

Any empirically derived model will have error. Statistical methods such as hypothesis testing employ significance tests to determine whether a measured effect is "real". Such tests remove the human element by determining the likelihood a measured result could be generated by random chance under the assumption that the effect is "fake". (10). Even if the data collected are accurate and complete, confounding variables may still be present in any models created based on choice of groups and variables. According to Glenn Hymel, "If an extraneous variable is not appropriately controlled, it may be unequally present in the comparison groups. As a result, the variable becomes a confounding variable." (11)

We will use the t distribution to perform significance tests at the $\alpha = 0.05$ level, where α is the probability of rejecting the null hypothesis when it is

true. (10) α is also known as the type I error. (10) For any significance test, we first define the null and alternate hypotheses. Faculty effects on future cumulative GPA can be either positive, zero, or negative, as can their effect on withdraw rate. There is no information outside the data that lead us to believe an individual faculty member will have a positive or negative ΔW or mean ΔC, so we will use a two sided hypothesis. Where θ is the true value of a statistic (i.e. the ΔW or mean ΔC which would be calculated after an infinite number of students for one faculty): (10)

$$H_0 : \theta = 0$$

$$H_A : \theta \neq 0$$

Just as it would be impossible to collect infinite variables to ensure perfect normality, we cannot mathematically prove from a sample which provides an estimate $\hat{\theta}$ that the null hypothesis is false. However, we can create a range of values where θ is expected to lie based on the observation of $\hat{\theta}$. This concept is called a confidence interval. (10, 12) In my analysis, I will construct such confidence intervals using two techniques from mathematical statistics.

As the name suggests, the one sample t-test utilizes the t distribution. Where α is the probability of type I error and $t_{\alpha/2,N-1}$ is the t statistic's $\frac{\alpha}{2}$ quantile for N degrees of freedom, the derivation of the $100(1-\alpha)\%$ confidence interval follows: (10, 12)

$$P(-t_{\alpha/2,N-1} \leq t \leq t_{\alpha/2,N-1}) = 1-\alpha$$
$$P(-t_{\alpha/2,N-1} \leq \frac{\bar{x}-\mu}{\sigma_x/\sqrt{N}} \leq\leq t_{\alpha/2,N-1}) = 1-\alpha$$
$$P(\bar{x} - t_{\alpha/2,N-1}\frac{\sigma_x}{\sqrt{N}} \leq \mu \leq \bar{x} + t_{\alpha/2,N-1}\frac{\sigma_x}{\sqrt{N}}) = 1-\alpha$$

$$100(1-\alpha)\% \quad CI = [\bar{x} - t_{\alpha/2,N-1}\frac{\sigma_x}{\sqrt{N}}, \bar{x} + t_{\alpha/2,N-1}\frac{\sigma_x}{\sqrt{N}}] \tag{6}$$

Using this 95% confidence interval, our hypothesis tests on θ using the t-statistic are: (10, 12)

$$H_0 : \theta \in [\bar{x} - t_{\alpha/2,N-1}\frac{\sigma_x}{\sqrt{N}}, \bar{x} + t_{\alpha/2,N-1}\frac{\sigma_x}{\sqrt{N}}] \tag{7}$$
$$H_A : \theta \notin [\bar{x} - t_{\alpha/2,N-1}\frac{\sigma_x}{\sqrt{N}}, \bar{x} + t_{\alpha/2,N-1}\frac{\sigma_x}{\sqrt{N}}] \tag{8}$$

The bootstrap principle is a powerful tool that can allow for significance testing with fewer assumptions than the one sample t-test. The general process for the bootstrapping of a population parameter θ using a given sample containing an estimate $\hat{\theta}$ follows: (10)

1. Create an empty vector V_{boot} with length equal to a large number n_{sim}.
2. Create a sample with replacement S_{boot} from the original sample whose length is equal to the length of the original sample. In this example this

length is N.

3. Store the observed mean of $\hat{\theta}$ from (2) into V_{boot}.

4. Repeat steps (2) through (3) $n_{sim} - 1$ times.

The resulting vector V_{boot} contains the bootstrap sampling distribution, which is an estimate for the sampling distribution due to equation (4). (10) As such, we can use the bootstrap distribution to create a confidence interval for θ. R's built-in quantile() function is used to create 95% confidence intervals for $\hat{\theta}$ from this bootstrap sampling distribution. (10) The $100(1-\alpha)\%$ bootstrap CI is given by the set: (10)

$$100(1-\alpha)\%CI = [\text{quantile}(V_{boot}, \frac{\alpha}{2}), \quad \text{quantile}(V_{boot}, 1-\frac{\alpha}{2})]. \tag{9}$$

The corresponding hypothesis tests on θ are given by: (10)

$$H_0 : \theta \in [\text{quantile}(V_{boot}, \frac{\alpha}{2}), \quad \text{quantile}(V_{boot}, 1-\frac{\alpha}{2})] \tag{10}$$

$$H_A : \theta \notin [\text{quantile}(V_{boot}, \frac{\alpha}{2}), \quad \text{quantile}(V_{boot}, 1-\frac{\alpha}{2})] \tag{11}$$

In my analysis, the ΔC statistic is calculable for all students that take only one course in the first term, do not withdraw from that course, and receive at least one more grade. It is possible to group students by the course code of their first course, then immediately create a bootstrap sampling distribution for ΔC using only those students. This is pointless; recall we define each

stratified linear models such that the mean ΔC over all students that take the same first course code is precisely zero. Furthermore, the average ΔC for a course code is not what we wish to test. Our goal in creating the ΔC statistic is to determine if individual faculty have a statistically significant effect on their student's final cumulative GPA. We will seperate this investigation into two experiments. Experiment one will be to test for significance the mean ΔC among all students that passed filtration and started with a given instructor. Experiment two will restrict the students the ΔC's for averaging to only those that started in the same course code with the same professor. In summary, we wish to construct a 95% range for the population (read: "true") mean ΔC among students that started with a professor, both overall and by individual course code. Faculty in OCC's Math and Chemistry departments vary in the number of students taught, which effects the coding methods necessary for establishment of bootstrap confidence intervals. Specifically, we must on a case-by-case basis adjust the number of observations per bootstrap sample to mirror the number observed. To reiterate, in experiment one the number of observations per bootstrap sample will be equal to the total number of students that started with the faculty. In experiment two we set the number of observations per bootstrap sample equal to the number of students that started with the faculty *in a particular course code*. There are two ways to accomplish this: (10)

1. In experiment one, we can create a bootstrap sampling distribution by drawing samples *from the overall list of* ΔC*'s* for all students. In experiment two the ΔC's available for sampling are restricted to those of

students that started in the desired course code. If the number of simulations and the number of values are each sufficiently large, one can use R's quantile() method to determine the 95% CI as outlined above. The observed mean for ΔC is then checked against the confidence interval. If the observed mean is within the confidence interval we fail to reject the null hypothesis that the faculty did not have a significantly different effect on a student's final cumulative GPA than the mean difference. If the observed mean is outside the boundary, we have sufficiently shown at the $\alpha = 0.05$ level that the faculty's mean effect was different than the overall mean effect. Moreover, for such significantly different faculty we have shown at the $\alpha = 0.025$ level that the mean effect was either positive or negative depending on the polarity of the observed mean. This method amounts to determining the probability of randomly generating a bootstrap sample with a mean at least as extreme as the observed mean.

2. To prevent needing to simulate the sampling distribution from the total sample for each sample size possessed by a faculty member, we can simply use the ΔC's attained by those students that passed filtration and started with the faculty member as the population from which bootstrap samples are created (in a particular course in the case of experiment two). The confidence interval generated by sampling from the ΔCs attained by a faculty's students range for the true mean ΔC of that instructor. Therefore, if the stratum's mean ΔC is within the generated CI, the faculty's effect cannot be shown to be different than the mean. We then

fail to reject the null hypothesis. On the other hand if the 95% confidence interval does not include the overall mean, then we reject the null hypothesis in favor of the alternate. Again, for faculty with significantly different effects we show at the $\alpha = 0.025$ level that said effect is either positive or negative, depending on the polarity of the observed mean. This method has the added benefit of permitting individual faculty to originate from different population mean ΔC distributions than other faculty, say, if chemistry instructors are fundamentally different from math instructors. In my analysis I chose to use this method to slightly save on processing time, but to my knowledge both are equally valid.

For a faculty's ΔW the bootstrapping process and experiments follow exactly as for their mean ΔC. As a final note, the bootstrap confidence interval may not accurately represent the sampling distribution if there are not enough unique observations. This is a particular issue with a fundamentally discrete statistic like difference between observed and expected final cumulative GPA. On the topic of a minimum original sample size for using the bootstrap method, the author of two books on the bootstrap Dr. Michael R Chernick states that: (13)

> Now if the sample size is very small—say 4—the bootstrap may not work just because the set of possible bootstrap samples is not rich enough. In my book or Peter Hall's book this issue of too small a sample size is discussed. But this number of distinct bootstrap samples gets large very quickly. So this is not an issue even for

sample sizes as small as 8.

From this guideline I chose to discard all faculty effects that were measured from fewer than eight students. I also chose to use eight students as minimum for the t-tests to create a closer approximation to normality.

Case Study

Initial Manipulations

Without further adieu, let's do some data analysis. I forego including my program itself in this section in favor of linking the GitHub repository in the appendix. The repository contains all necessary data, images, and a copy of the code that creates this report. To familiarize ourselves with the basic dataset posted by Professor Eckstrom we will take a glance at the first six observations:

Student Random ID	Course Code	Grade	Faculty Random ID	Semester
31	MAT-1540	F	643249	2012/SU
31	MAT-1540	WP	643249	2012/FA
31	MAT-1540	C-	287234	2013/SU
31	MAT-1540	WP	650997	2014/SU
48	MAT-1500	W	105920	2012/SU
50	MAT-1100	A	723344	2011/WI

One observation pertains to one course taken in one semester by one student taught by one faculty. The letter grade recieved for this course is recorded in the variable Grade. Our eventual goal is to create a statistic based on cumulative GPA and graded term, so a good place to start the data wrangling process is converting letter grades to the standard four point GPA scale. The function used for creating GPA scores is called convert_grades(), and was

adapted from stackoverflow user (sic) A5C1D2H2I1M1N2O1R2T1. (14) We also create integer representations of the semesters under the variable name Total.Term, where Total.Term takes a minimum at 1 in the fall semester of 2010 and a maximum at 20 in the winter semester of 2017. The function which accomplishes this task is Number.Semesters().

There are some missing values in the variable "GPA Assigned" which correspond to withdraw grades. Students may withdraw from courses for any number of reasons, and so no inference can be properly made about the GPA they would have received. Withdraws will be discarded in the analysis of faculty's mean ΔC's, but will be integral to the investigation into a faculty's individual ΔW. To analyze trends in cumulative GPA, there must be a cumulative GPA variable. Cumulative GPA is a recursive calculation that depends on the number of credits granted by a course and the previous value of cumulative GPA. A table of course credits can be found in the appendix. Before we can create cumulative GPA we must first create another variable that will be used in its calculation.

Although the Total.Term variable provides a numerical representation of the semester variable, Total.Term does not give information as to which semesters an individual student was enrolled in OCC's Math and Chemistry departments. In order to create a linear model for cumulative GPA progression as a function of term enrolled, it is easiest to standardize the time scale for each student. The function I created called student.terms() maps any new unique term to the first available natural number given a set of indices in Gen_Data

that define a student. A helper function student.index() identifies the rows in Gen_Data possessed by a student. For example: the student with random id number 50 took four courses in the Math and Chemistry departments in the total term set $\{2, 4, 8, 10\}$. Student number 50 is the third student from the top when arranging by students, so student.index(3) returns the indices in Gen_Data possessed by the desired student. Passing the output of student.index(3) to student.terms() returns the vector $\{1, 2, 3, 4\}$. The Student.Term variable is initialized in Gen_Data by looping this process over all students. The max.terms variable simply records the largest number in the set of student terms, and will be useful later.

Cumulative GPA is calculated by taking the ratio of the sum of the products of course credits and GPA received with the sum of the course credits. Where C is cumulative GPA, R is course credits, A is the GPA assigned for the course, and the indices $i \in [1, N]$ refer to all the graded terms that contribute to cumulative GPA:

$$C = \frac{\sum_{i=1}^{N} R_i * A_i}{\sum_{i=1}^{N} R_i} \tag{12}$$

Although unnecessary, to observations that resulted in withdraws I assign the previous recorded cumulative GPA if extant. This is just for bookkeeping, and does not affect any statistical analysis or graphic produced. If a student has withdraws from a course without having a previous graded course in the Math and Chemistry departments, the observation of cumulative GPA retains its missing value.

Building Delta C

Ideally our measure of faculty effect on final cumulative GPA would not require the faculty to be the first encountered in a department, because some professors may only teach upper level courses and therefore have many fewer students start with them. This decrease in sample size increases standard error and thus makes significance difficult to prove. Unfortunately, not all faculty can contribute equally to final cumulative GPA. For a student that takes four courses in the Math and Chemistry departments the first encountered faculty might be said to have a four term effect whereas the last faculty only contributes to the final term. It is unfair to give a "good" last faculty a negative score because of the contributions of previous "bad" faculty. We therefore restrict our inquiry to the first faculty encountered. Tables 3 and 4 describe how the overall sample size is affected by each filtration condition.

Table 4: Proportion of students in OCC's Math and Chemistry departments that withdrew from their first course.

First.Term.Withdraw	prop
No	0.781
Yes	0.219

Table 5: Proportion of students in OCC's Math and Chemistry departments that recieved grades for at least two courses

At.Least.Two.Grades	prop
No	0.607
Yes	0.393

Before we proceed, let's examine some potential source of error. The data we have are only for the courses that students took in the Math and Chemistry departments during their stay at OCC. We cannot with any degree of certainty claim that those first recorded terms in the dataset are indeed the first terms at Oakland Community College. Student X could have taken an English course in a term before appearing in Professor Eckstrom's data. We have implicitly filtered for: (1) students who did not for some reason attend a standard four year college, and (2) students that wished to take math and/or chemistry courses in community college. We have moreover restricted our departmental information to only chemistry and mathematics courses, which removes our ability to discern the possibility that the math or chemistry departments are outliers in terms of cumulative GPA progression and withdraw rate. I will now indulge in conjecture, so take this next section with a grain of salt. Students pursuing degrees in other departments may be less likely to complete numerous courses in the Math and Chemistry departments. Students only filling prerequisites such as quantitative reasoning or calculus one would be less likely to pass the

filtration condition of taking at least two graded terms. The error is therefore mitigated because problematic observations are more likely to get picked off. Such an effect is not falsifiable without a larger dataset containing more departments for comparison, and is therefore an important follow-up question.

After filtering for students that (1) took only one course in their first term which (2) they received a grade for and (3) proceeded to receive a grade in a different term, we are left with 21605 unique students from the original 66164, distributed over 267 unique faculty members. Only one faculty member is removed entirely by these conditions, though many more will have insufficient data to be significantly differentiated from the mean professor. The mean faculty member contributes to 80.918 measurable outcomes. The new variable "First.Prof" is created for all students that passed filtration. As the name suggests First.Prof simply takes value equal to a student's first observed faculty, which is a unique value due to filtration.

Linear Mixed-Effects Modeling

As of 4/2021, the package lme4 was an up-to-date mixed modeling package available through CRAN. (15) The lme4 function lmer() allows the user to generate a linear model with a combination of fixed and random effects. A linear mixed-effects model represents a dependent variable - in our case the cumulative GPA of a student - as a linear function of some inputs. The benefit of using such a model is that instead of just using a fixed effect as predictor, such as the student's term number, we can manually specify certain variables to lme4 that either effect the intercept or the slope of our linear model. (15,

16) Each student enters the dataset with various levels of previous exposure to mathematics and chemistry, and encounter different challenges during their first term; there are myriad factors which influence a student's first observation of GPA. Mixed-effects models allow an analyst to adjust for these factors. (15, 16) We will build two linear mixed-effects models - one using lmer() and one from scratch. After comparing the models, we will choose the better for usage in the data analysis.

Confounding Variables

One influence on future cumulative GPA achievement is the course in which students begin their tenure at OCC's math/chemistry departments. For instance - there may be a difference in both the final cumulative GPA and the slope for cumulative GPA in the mean student that begins their community college math courses in Preparation for Algebra (MAT - 1050) and a student that began in Calculus II (Mat - 1740). Figure 1 is a series of boxplots that show the distribution of final cumulative GPA for students that begin in each allowed first course. It is clear from this plot that the best linear model for a student that starts in a particular course would be specific to their peers. If we were to lump the students that started in CHE-1510 with the students that started in MAT-1050, the chemistry students would tend to finish with cumulative GPAs above the mean just as the math students would be erroneously assigned negative scores.

Students take courses in the Math and Chemistry departments for different number of terms. Oakland Community College only grants Associate's degrees

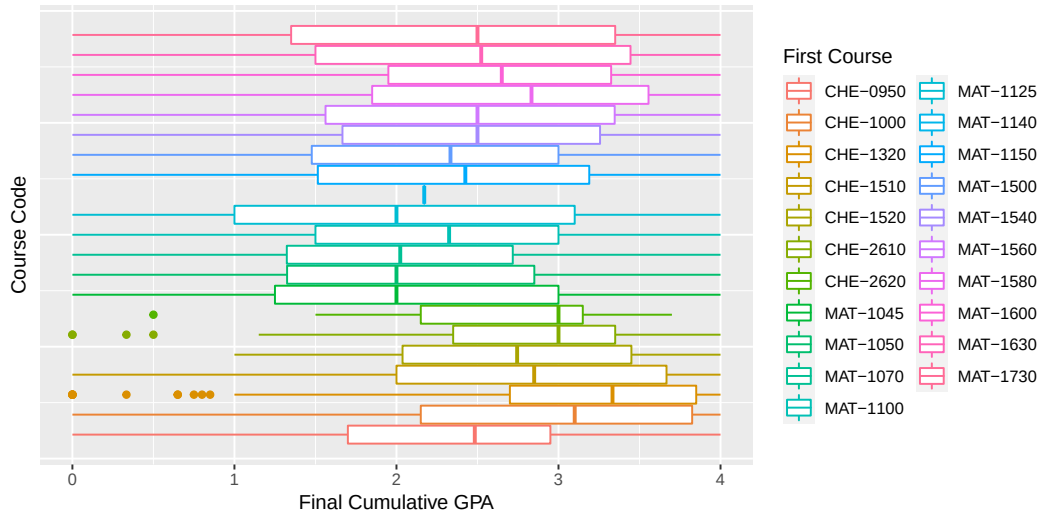

Figure 2: Boxplots of Final Cumulative GPA for all Students that Passed Filtration Conditions, Colored by First Course

in Math and Chemistry - the highest math course is Linear Algebra, and the highest chemistry course is Organic Chemistry II. While it may be a false equivalency to compare students that enroll in community college to those that enroll in standard four year schools, the highest courses offered at OCC should be completed at the latest by the end of a four-year student's second year assuming the student is majoring in that field. By this rough heuristic I estimate that a student beginning in Calculus 1 should complete OCC's math regime in about four terms at one course per term. A student might not take a math course every term, and community college students that are seeking mathematics degrees may require additional training before calculus.

As can be seen in figure 2, the mean final cumulative GPA for those students that take between three and seven courses are essentially identical. Students

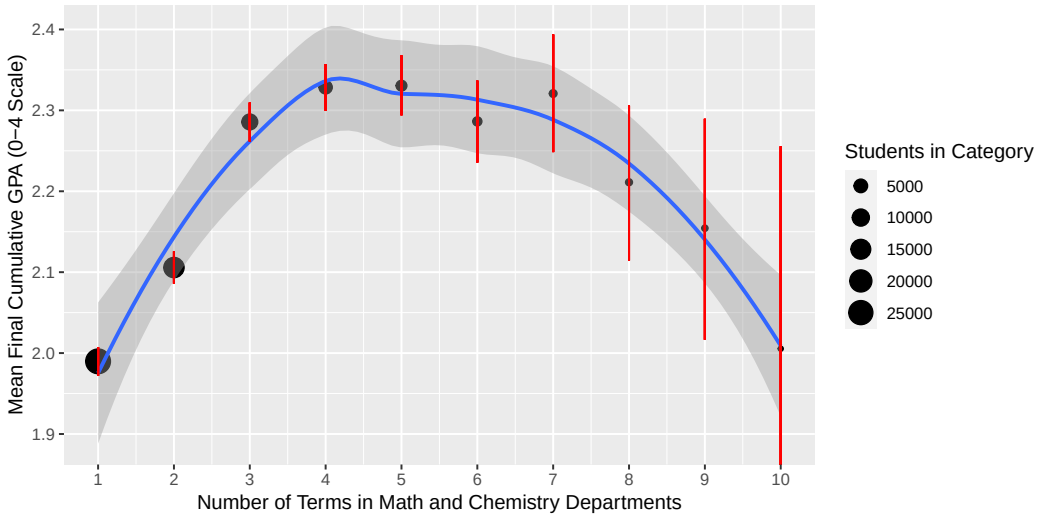

Figure 3: Mean Final Cumulative GPA Vs. Number of Terms in the Math and Chemistry Departments, with 95% Confidence Interval

that complete their stay in OCC's Math and Chemistry departments within this five course spread have the empirically highest mean cumulative GPA's by the time they finish. Past each extreme of this range mean final cumulative GPA begins to fall. There are two orientations, to each of which I will assign a speculative reason for the lower cumulative observed GPA. Please note the following hypotheses are merely my guesses at the mechanisms, and there are likely additional factors that I have not thought of that affect mean final cumulative GPA by max term.

1. **Less than three terms:** A student that takes less than three terms in the Math and Chemistry departments may be testing out community college itself, or perhaps trialing the courses offered in the respective departments. Such students may not be as committed to their learning as

those pursuing an associate's degree in math or chemistry. By choosing the students that only take one or two terms of Math and Chemistry, we have retroactively largely filtered out those students that ARE pursuing such a degree, thus lowering the mean final cumulative GPA. An alternate explanation is that if a student does poorly in their first one or two courses in a department, they become less likely to continue study in that field. There would then be a higher concentration of lower grades in the first few terms, as we observe.

2. **Greater than seven terms:** Students that are hellbent on obtaining a degree in math or chemistry will take courses in those departments until they either finish the degree or change their conviction. If a student fails multiple courses within the department, they will necessarily need to take additional terms to retake the classes. They will then have both a higher number of terms taken, and a lower cumulative GPA.

The mean final cumulative GPA for all terms lie within the set (1.99, 2.33) GPA points. The minimum occurs in those students that only take one term in the Math and Chemistry departments, and the maximum occurs at five terms. The range is only about 0.3 GPA points, so the effect that the total terms taken is rather small, approximately the difference between a C average and a C+ average.

As the number of terms increases, the number of students that were enrolled in the Math and Chemistry department for that number of terms decreases exponentially and moreover monotonically. I was forced to filter for terms that

had at least thirty students finish within them, thus losing a small amount of data (on the order of 50 students). The confidence intervals for the later terms become increasingly wide, to the point where the negative trend observed in terms 8 through 10 is barely significant at the $\alpha = 0.05$ level. Another interesting statistic is the mean change in cumulative GPA per term, which we will look at presently.

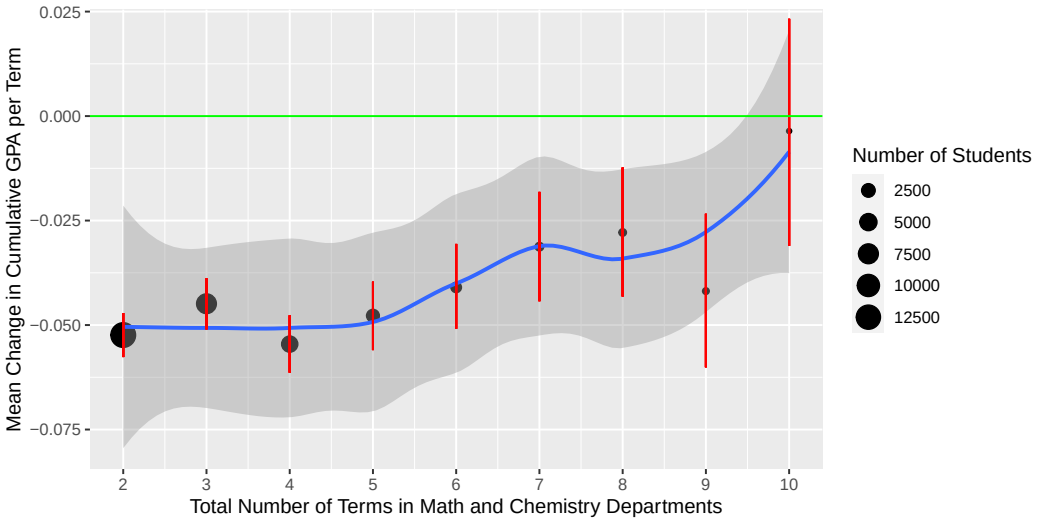

Figure 4: Mean change in cumulative GPA per term Vs. maximum number of terms, with 95% confidence intervals.

Figure 3 compares the mean change in a student's GPA against the number of terms that they took courses in the Math and Chemistry departments. This statistic is measured as the difference between final cumulative GPA and the first recorded observation of GPA, divided by the number of graded terms. As in figure 3, ten is the highest term number for which there is sufficient data for inference. I did not have any guesses regarding whether certain terms would yield positive or negative effects, so we have used a two-sided confidence

interval. These will be compared about the expected effect of zero. If we let the variable measured on the vertical axis be s for slope, the hypotheses we will test for significance for each term are: (10)

$$H_0 : s = 0$$

$$H_A : s \neq 0$$

For all terms except 10, at the $\alpha = 0.05$ level we reject the null hypothesis in favor of the alternative. Because we were using a two sided confidence interval, we have shown that for terms one through nine that $s < 0$. The 95% confidence interval for term 10 intersects with the green line demarking the null hypothesis, so for term 10 we fail to reject H_0. We are running ten tests at the $\alpha = 0.05$ level, and so we would expect roughly one in twenty experiments to have a population mean outside of the confidence interval. For most of these CI's, the boundary is rather tight and so even in the case of such errors, the population mean is unlikely to usurp the validity of our results. Note that these changes in slope are relatively small. The average student that took five terms in OCC's Math and Chemistry department experienced a -0.04 decrease in cumulative GPA per term, for a total of -0.24 GPA points. The end effect is rather large, so it is worth adjusting for in the upcoming linear mixed-effects models.

Using these plots we have shown that students with the highest mean final

cumulative GPA tend to take between three and seven terms in the Math and Chemistry departments, and that those students also have a downwards trend in their cumulative GPA from their entry into the departments to the last course they take.

Deciding on a Model

To provide a better prediction on final cumulative GPA we create a stratified model. Each course is assumed to have its own mean final cumulative GPA slope against which to measure student outcomes. A student's expected final cumulative GPA is determined by adding the GPA assigned by the first professor (a random effect) to the product of the number of graded terms and the first course's mean cumulative GPA slope (a fixed effect). Where C_E is expected cumulative GPA, G_0 is the first observation of GPA, T is the current graded term number, S is the first course's change in mean cumulative GPA per term, and k is an adjustment on slope determined by the current term, the linear mixed-effects model is defined as follows: (15, 16)

$$C_E = G_0 + T(S + k) \tag{13}$$

Each student is given their own affine predictor. By evaluating a student's model at their final graded term we determine the final cumulative GPA their average peer would have received given G_0. The derivation of ΔC follows as in equation (1). Note that there was only a single student to start in MAT-1140 and pass filtration: student number 460449. It will be impossible

to create a valid model for that student because their ΔC will always get mapped to precisely zero. We will create models in two contexts: using the lmer() function and manually. We will use whichever ends up being the better of the two models.

The lmer() method from the R package lme4 provides a straightforward method of creating hierarchical models. First a user must specify how the variables are related, then define the data from which to obtain said variables. In lmer() syntax, a fixed effect is denoted by the name of the variable with no additional formatting. For example, the model "Cum.GPA ~ Graded.Term" would simply create a standard fixed-effects linear regression model with dependent variable cumulative GPA and independent variable graded term number. To define a random intercept with no correlated variables, we use the (1 | X) notation: "Cum.GPA ~ Graded.Term + (1 | `Student Random ID`)" creates a model where all students have the same change in cumulative GPA per term, but may start with different GPAs. Allowing a correlated variable to act as both a random slope modifier and as a random intercept modifier requires the (1 + Y | X) clause. lmer() has the notable benefit of allowing grouping to be defined within the model to prevent unnecessary duplication of effort. Unfortunately, when I used this feature the models failed to converge. It is only a little more work to manually stratify the students by first course before running mixed effects models on each. After adjusting for first course, each of the 19 linear mixed-effects models have the same formula: "Cum.GPA ~ Graded.Term + (1 + Graded.Term | `Student Random ID`)". This allows each student's random deviation from the mean to affect both their starting

point and their cumulative GPA slope. ΔCs are calculated in accordance with equation (1) using the lmer() derived mean slopes per stratum.

As we will see later, lme4 has some issues with my data when supplied - likely from user error. I did quite a lot of reading on linear mixed-effects models, and recreated the concept from scratch using dplyr functions. First I stratify by each combination of total terms in the Math and Chemistry department and first course taken that had at least eight students. Now that the confounding variables and outliers are accounted for, we calculate the mean change in cumulative GPA per term for each stratum. These are the slopes of the linear mixed-effects models. Finally, we run the slope from the first observation of cumulative GPA, then calculate ΔC in accordance with equation (1).

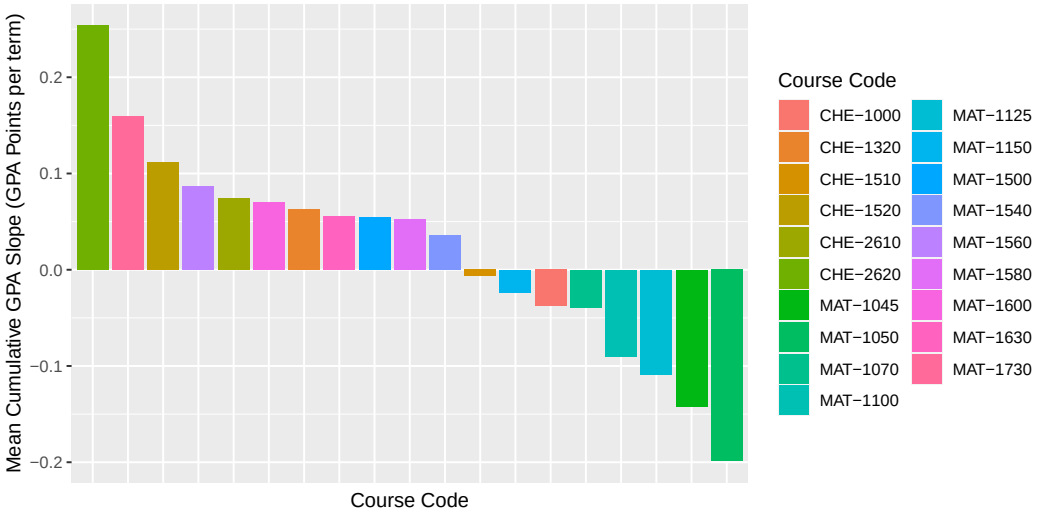

Figure 5: Barcharts for mean slope of final cumulative GPA per term enrolled by first course code for each student that passed filtration. Mean course slopes calculated manually.

Figure 6: Barcharts for mean slope of final cumulative GPA per term enrolled by first course code for each student that passed filtration. Mean course slopes calculated by lme4.

Table 6: Absolute differences in mean course slopes between the manual and lmer models

Course.Code	Slope.Diff	Num.Students
CHE-2620	0.149	8
MAT-1730	0.108	548
MAT-1125	0.085	42
MAT-1050	0.061	4158
CHE-1520	0.054	41
CHE-2610	0.053	57
MAT-1045	0.043	1463
MAT-1070	0.040	117

Course.Code	Slope.Diff	Num.Students
MAT-1560	0.034	376
CHE-1320	0.034	140
MAT-1600	0.033	60
MAT-1100	0.027	7321
MAT-1500	0.025	287
MAT-1540	0.025	1222
MAT-1580	0.023	513
CHE-1000	0.022	920
MAT-1630	0.015	240
CHE-1510	0.004	363
MAT-1150	0.002	3579

Figures 4 and 5 show the mean cumulative GPA slopes for each first course that had at least eight students, calculated manually and by lmer() respectively. Table 5 shows the absolute value of the difference in the mean slopes between figures 4 and 5. CHE-2650, MAT-1140, and MAT-1525 were removed from both models because less than eight students began in those courses. (13) If the two models were calculated identically, each entry of the Slope.Diff column in table 5 would be zero. One model will end up being better than the other, though we cannot determine which based only on the mean course slopes. Let's see how they do at evaluating the ΔC statistic, keeping in mind that the mean ΔC for each course code *should* be zero barring some minor deviations based

on the discreteness of cumulative GPA and non-linearity of cumulative GPA. The better method will be the one whose individual course mean ΔC's deviate the least from zero.

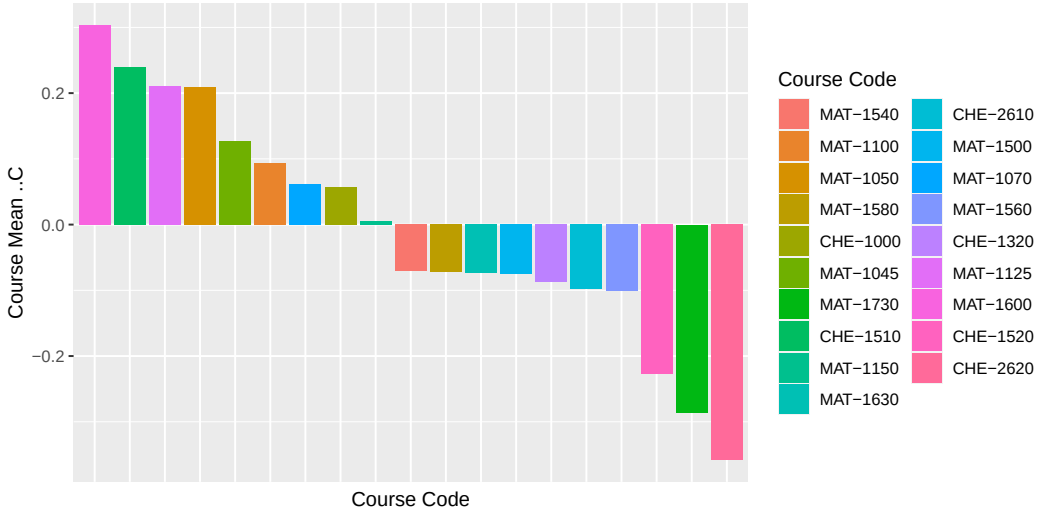

Figure 7: Barchart of course mean Delta C's as generated by lmer models

Figures 6 and 7 show the mean ΔC for each course as predicted by the lmer() and manual linear mixed-effects models. The distribution of the lmer() generated ΔC's clearly have wide variance for each course, signifying that the wrong statistic is being calculated. Specifically, the range of mean ΔC's over all courses is approximately $[-0.4, 0.3]$. It seems that simply supplying lmer with a random intercept called Student Random ID that has a slope and intercept interaction with the current term number is insufficient in the generation of an unbiased ΔC. Removing the max.terms stratification results in the manual models creating similarly biased ΔC's, so perhaps lmer() is not properly recognizing how maximum terms affects slope and instead calculated

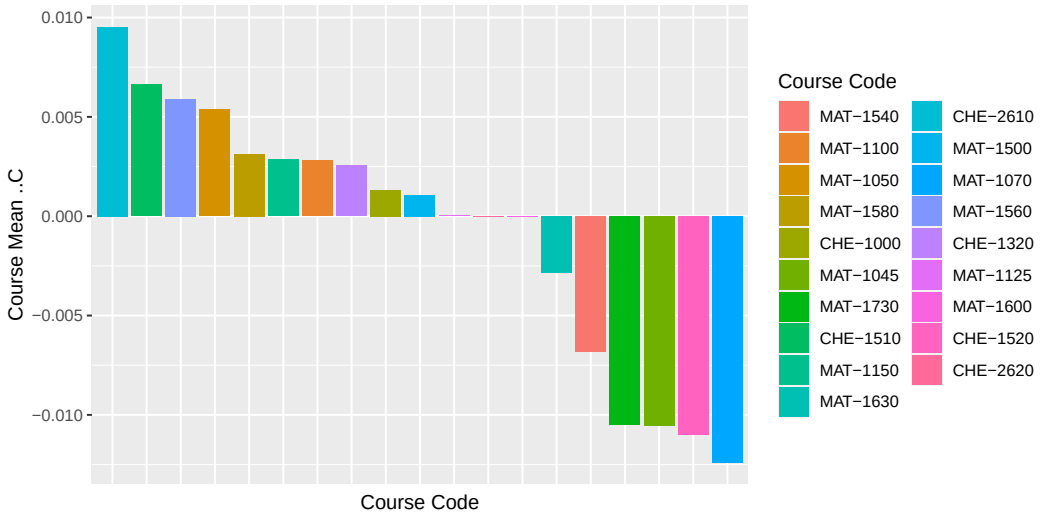

Figure 8: Barchart of course mean Delta C's as generated by manual models

the change in slope on a term-by-term basis. My bet is the maximum terms slope dependency was automatically captured as part of the student-to-student random deviation, but applied as a random intercept rather than a random slope. Ultimately, lmer did not work. In contrast, the mean ΔC for each course generated by the manual models compactly surround zero. The range of course mean ΔC's in the manual models was approximately $[-0.015, 0.01]$. The manual models were better by an order of magnitude, so we will use them exclusively going forwards.

Figures 8 and 9 depict normal probability plots using every student's ΔC calculated in experiments one and two respectively. A perfectly normal distribution would be a straight diagonal line, indicating a one-to-one correspondence between the theoretical normal quantiles and the observed sample quantiles. There is a small amount of curvature on the tails of the experimental distribu-

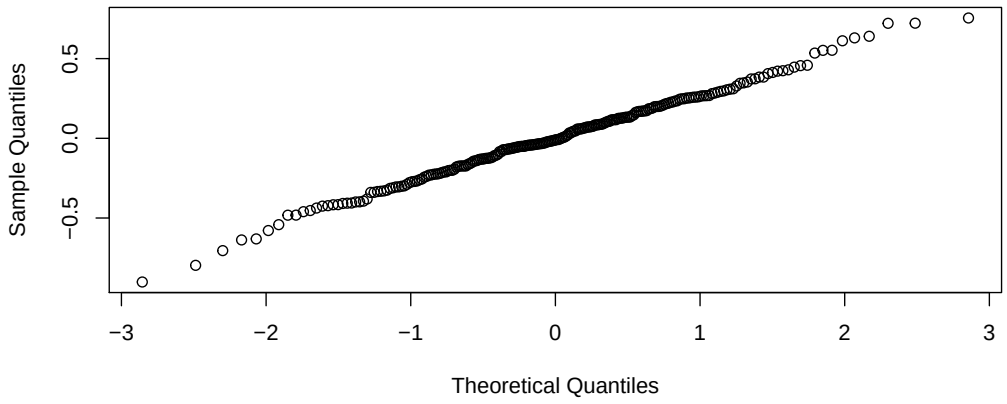

Figure 9: Normal probability plot of all manual professor mean DeltaC's from experiment one.

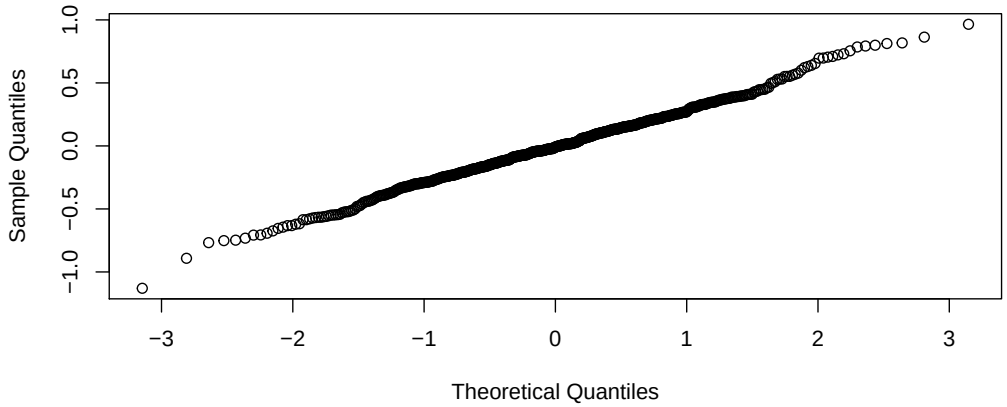

Figure 10: Normal probability plot of all manual professor mean Delta C's from experiment two.

tion, but this is expected from sampling error and the fact that ΔC is a discrete variable bounded on the set $[-4, 4]$. Because the population distribution of ΔC is normal for both experiments we are sanctioned for usage of the one sample t-test. Moreover, because each stratum has at least eight students, we can also use the bootstrap principal for significance testing. These procedures are carried out in accordance with equations (8) and (11). For visualization of how the bootstrap confidence interval is employed for significance testing, I have included figures 10 and 11.

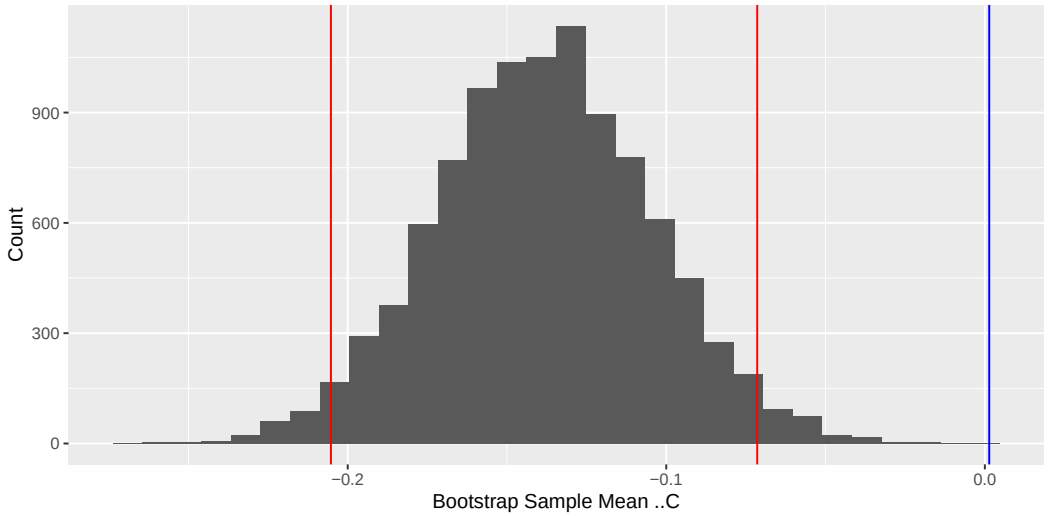

Figure 11: Example histogram of a bootstrap confidence interval that indicates a rejection of the null hypothesis. Simulated using DeltaC among students first placed with faculty number 643249

The overall population mean for ΔC in the manual models is 0.001 with a standard deviation `r`, which means that either: 1) there is at least one confounding variable that affects a student's cumulative GPA unaccounted for, or that 2) the term dependency of a student's cumulative GPA is not quite linear.

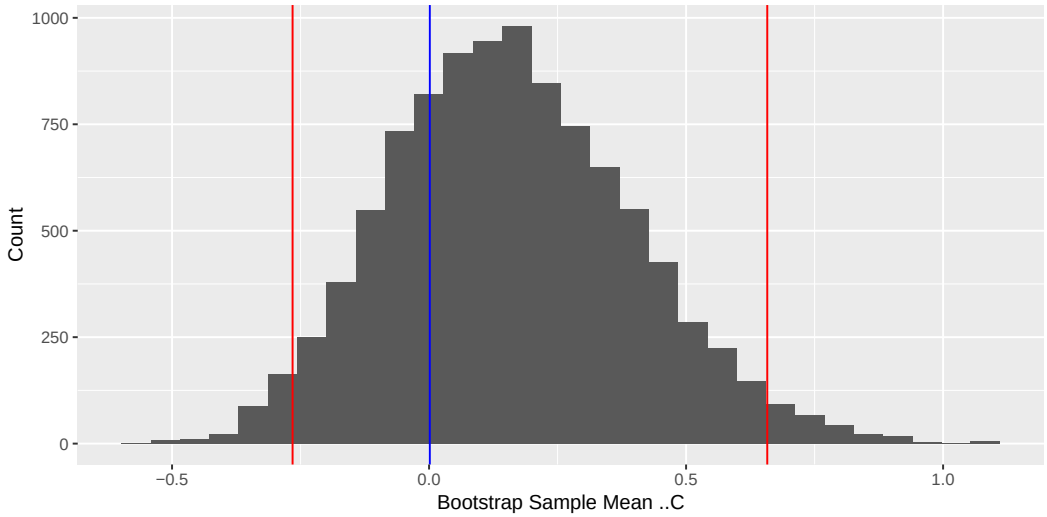

Figure 12: Example histogram of a bootstrap confidence interval that fails to reject the null hypothesis.

I am personally more inclined to believe the latter because GPA is bounded on the interval $[0, 4]$ and thus cannot perfectly follow any affine path over an arbitrary number of semesters (unless the slope were exactly zero). That said, for the mean student taking the mean number of terms the manual models are pretty good. Now that we know the overall mean value for ΔC, we can use the one sample t-test and bootstrap confidence intervals with $\alpha = 0.05$ to determine whether a given faculty member had a mean positive or negative on their students final cumulative GPA's. As an aside, the ΔC statistic we have created is not immune to exceptionally harsh/lenient grading by the first professor, because it is measured assuming that the first observed GPA value was an accurate intercept for the linear mixed effects model. We assume that course grading is standardized in OCC's Math and Chemistry departments.

Delta C Results

Delta C Experiment One Results

Using the confidence intervals outlined in equations (6) and (9), significance calculations are carried out on the manual model.

Table 7: Number of faculty members that had at least eight students start with them in experiment one. We can run significance tests on the 233 with at least eight.

Sufficient.Sample.Size	n
FALSE	27
TRUE	233

Table 8: Discrepencies found in significance tests for experiment one. Out of 233 faculty, the one sample t-test and the bootstrap confidence interval agreed regarding whether a faculty's mean DeltaC was significantly different than the overall mean 224 times, and disagreed 9 times.

T.Test.Only.Reject	Bootstrap.Only.Reject	n
FALSE	FALSE	224
FALSE	TRUE	9

Table 6 shows that out of the 260 faculty that survived filtration, 233 had at least eight students start in one of their courses. In experiment one we can run significance tests on 233 faculty. Table 7 summarizes the differences in experiment one between the overall one sample t-test and bootstrap significance measures for all faculty that had at least eight students begin in their courses and pass filtration. In the vast majority of cases, the bootstrap test agrees with the t-test. The significance implied by the t-test and the bootstrap agreed for 224 out of 233 faculty members, which is approximately 96.137% of all professors. 9 discrepancies occurred where the t-test implied that the observation of mean ΔC for a faculty member was significantly different from the overall mean, while the bootstrap implied that it was not. There were no observed cases where the t-test rejected significance while the bootstrap failed to reject. We would expect more bootstrap rejections than t-test rejections if the sample data were not sufficiently rich due to the sample sizes being too small. The requirement that a faculty have a minimum of eight students start in their courses helps with the richness issue, but the high variance in ΔC between students is something to be mindful of when considering the results.

Here are some relevant statistics regarding experiment one. The average mean ΔC for those faculty who were doubly-significant and had positive effect was 0.348 with standard deviation 0.16. As for the faculty doubly-significant negative mean ΔC values, the average mean ΔC was -0.317 with a standard deviation of 0.17. The sample size (total number of students that passed filtration and started with a professor) had mean 91.635 and standard deviation 89.218. The distribution of the sample size is therefore clearly not Gaussian.

The experimental error rate for experiment one - where such an error is measured by the bootstrap and t-test disagreeing on the statistical significance of a faculty - was $\frac{9}{233} * 100\% = 3.86\%$. This is a bit high in terms of data loss as we lose ratings on nine faculty members, though it's certainly better safe than sorry. The overall accuracy of the experiment is increased because we eliminated nine potentially invalid results.

In an effort to be as conservative as possible with inference, we will now create plots using only those faculty members endorsed by both confidence intervals. Additionally, I will include a plot of sample size and a plot showing the relationship between the number of students that started with a faculty and the mean ΔC.

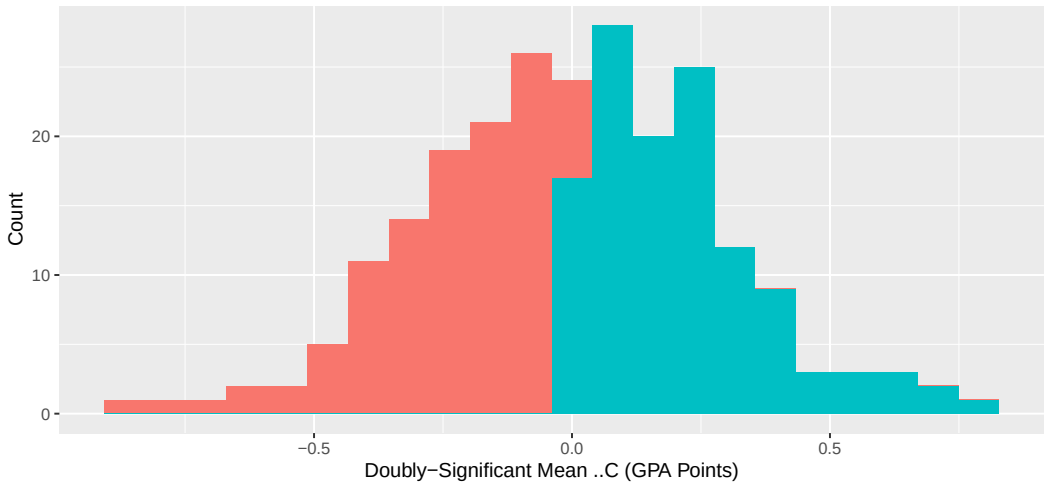

Figure 13: Histogram of all experiment one mean Delta Cs, colored by effect polarity from the mean.

The overall distribution of mean ΔC is shown in figure 12 to be approximately

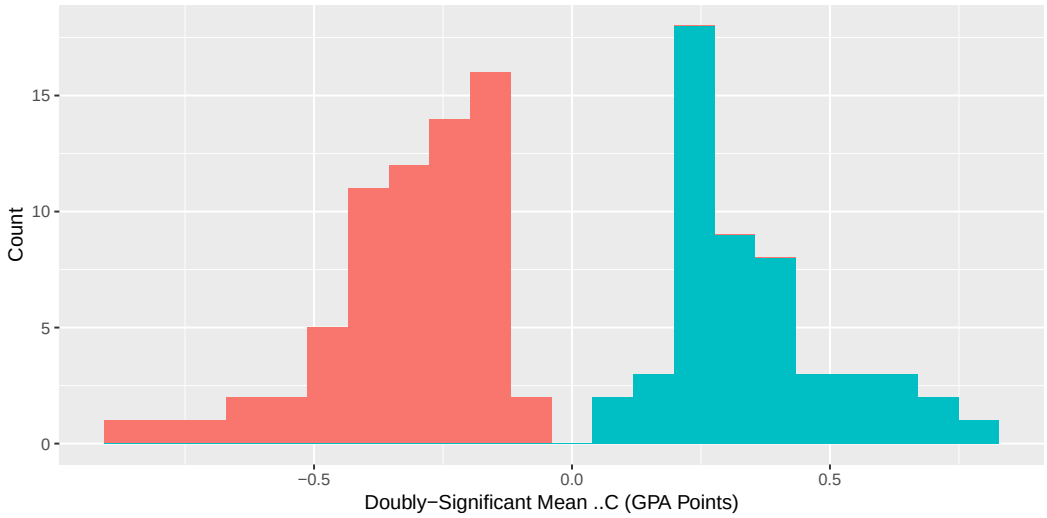

Figure 14: Histogram of only those experiment one mean Delta Cs that were significantly different from the global mean, colored by effect polarity from the global mean.

normal. After removing all the faculty that had effects indistinguishable from zero, we are left with the bimodal distribution in figure 13. I want to be clear: even though the faculty removed did not have an effect on their students *significantly different from the mean effect*, that does not imply that they did not have zero effect. The final mean cumulative GPA slopes for the students of these removed professors are well-represented by figure 4. A word of warning - just because the students placed first with a specific faculty happened to significantly underperform or overperform against the manual model does not necessitate a true effect. It is important to consider the result in context - did a faculty have a mean ΔC of 0.5 among twelve students or among twelve hundred? Both cases are represented equivalently in figures 12 and 13, though one is much more certain and impactful than the other. Figure 14 is intended

to provide such context.

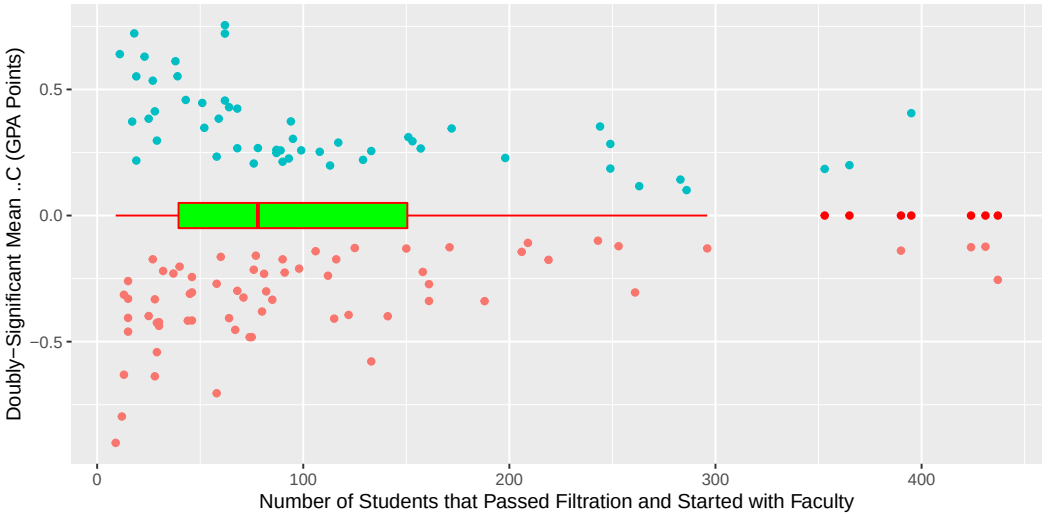

Figure 15: Scatterplot of each doubly significant faculty's mean Delta C against the number of students that passed filtration and started with them. Colored by effect polarity from the mean. Boxplot provided for summary of sample size distribution.

As the number of students trends towards zero, figure 14 indicates that mean ΔC becomes more volatile. This volatility seems to evaporate past the 0.5 quantile of sample size, stabilizing to a rough local average of ± 0.25 for each polarity. Additionally, the mean ΔC tends towards zero as the number of students grows. I can think of two potential causes for this trend. In the order of likelihood:

1. The statistic called mean ΔC is highly susceptible random variance. The approximate CLT given by equation (4) indicates that as the number of samples grows, the standard error decreases as we close in on the true population mean. As the number of students per doubly significant

faculty grows, the range of mean ΔC's decreases, indicating that the range was not so big to begin with. This uses the assumption that faculty that have taught many students have the same distribution of population mean ΔC's as faculty that have not taught as many students.

2. There is a fundamental difference between faculty that have taught many students and those that have not. In particular, the wide variance at low sample size could be seen as "good" low-student-count faculty having radical effective ideas and "bad" low-student-count faculty not knowing how to teach. This option is far less likely when considering the high-student-count behavior of the distribution. Any professor that teaches several courses is bound to learn what is effective and how best to disperse knowledge in a community college setting. This would indicate a universal increase in mean ΔC. While we observe an average increase in the mean ΔC for faculty with doubly-significant negative effects, we observe a *decrease* in the average mean ΔC for faculty with doubly-significant positive effects.

Figure 14 shows that while a faculty's mean ΔC may a good indicator of a real effect for higher sample sizes, it must be taken with a healthy dose of trepidation for low sample sizes as large ΔC's are likely illusory.

Delta C Experiment Two Results

Table 9: Number of faculty-course combinations that had at least eight students start with them in experiment two. We can run significance tests on the 607 with at least eight.

Sufficient.Sample.Size	n
FALSE	382
TRUE	607

Table 10: Discrepencies found in significance tests for experiment two. Out of 605 faculty-course combinations, the one sample t-test and the bootstrap confidence interval agreed regarding whether a faculty's mean DeltaC was significantly different than the course mean 560 times, and disagreed 45 times.

T.Test.Only.Reject	Bootstrap.Only.Reject	n
FALSE	FALSE	561
FALSE	TRUE	36
TRUE	FALSE	8

In experiment two, we increase precision to the point of individual courses. Whereas in experiment one ΔC was averaged across all the students that

passed filtration and started with a specific teacher (as long as at least eight students did), in experiment two the average was restricted across students that started with a faculty *in a course* (as long as at least eight students did). The sample size minimum caused many more omissions in experiment two than in experiment one. Table 8 shows that out of 989 faculty-course combinations, only 607 had at least eight students start in them. Two of these faculty-course combinations were from the omitted courses, so we are left with 605 significance tests to run.

Table 9 shows a larger significance correction than in experiment one. The experimental error rate for experiment two was $\frac{37+8}{37+8+560} * 100\% = 7.44\%$. This higher rejection rate is predictable because the sample size must necessarily go down when we choose not to average over all a faculty's courses - as long as the faculty taught at least two courses in which students started their path through OCC's Math and Chemistry departments. The mean sample size in experiment two was only 33.367 with standard deviation 33.832. Because the ratio of mean to standard deviation decreased from experiments one to two, we predict the distribution of experiment two's sample sizes to be even more skewed than that of experiment one.

The average mean ΔC for those faculty-course combinations that were doubly-significant and had positive effect was 0.431 with standard deviation 0.187. For the faculty-course combinations with doubly-significant negative mean ΔC values, the average mean ΔC was -0.419 with standard deviation 0.185. The sample size had mean 33.367 and standard deviation 33.832. The distribution

of sample size is therefore even more skewed than in experiment one.

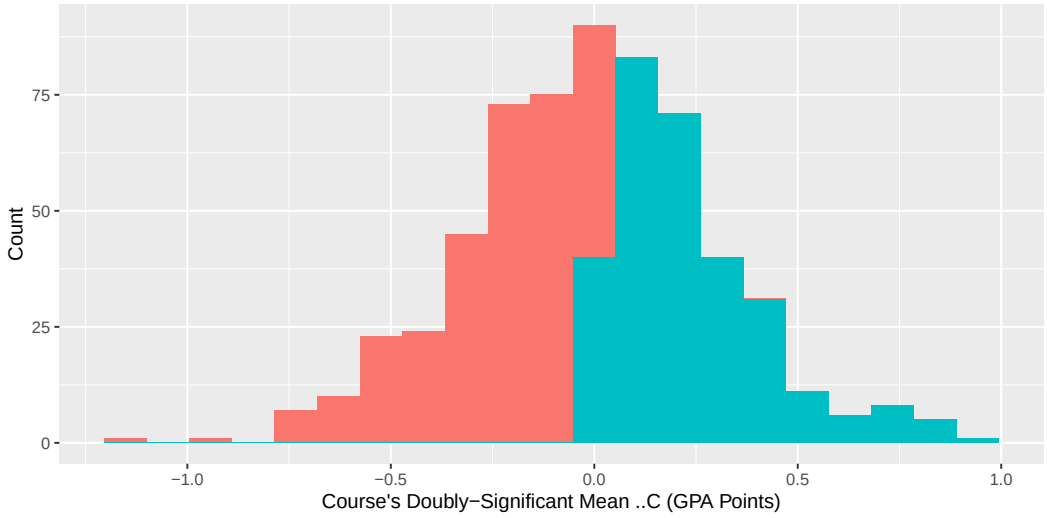

Figure 16: Histogram of all experiment two mean Delta Cs, colored by effect polarity from the mean.

The distribution of all experiment two mean ΔC's in figure 15 is not noticeably different than the distribution for experiment one observed in figure 12. Figures 16 and 13 are likewise similar. Experiment two includes information regarding the course to which an observation of mean ΔC belongs. We will generate the same type of context plot as we did for experiment one, but first let's separate the distribution of mean ΔC into each possible first course.

Figures 17 shows the distribution of mean ΔC for all faculty-course combinations, while figure 18 only depicts doubly significant faculty-course combinations. There are few faculty that had at least eight people start in their high level chemistry courses, indicated by the few dots in the CHE-2610 stratum of figure 17 and the altogether absence of CHE-2620 in either figure. Courses

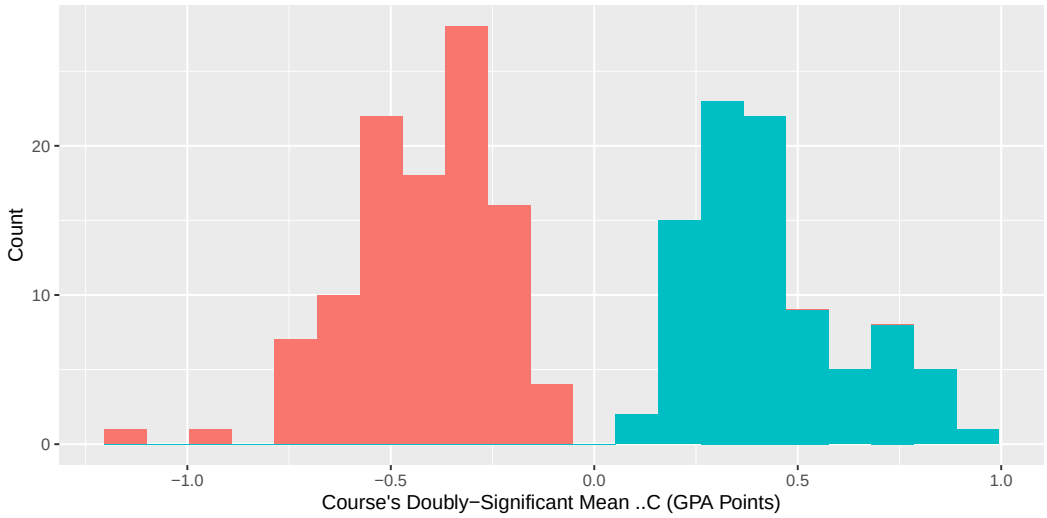

Figure 17: Histogram of only those experiment two mean Delta Cs that were significantly different from the course mean, colored by effect polarity from the course mean.

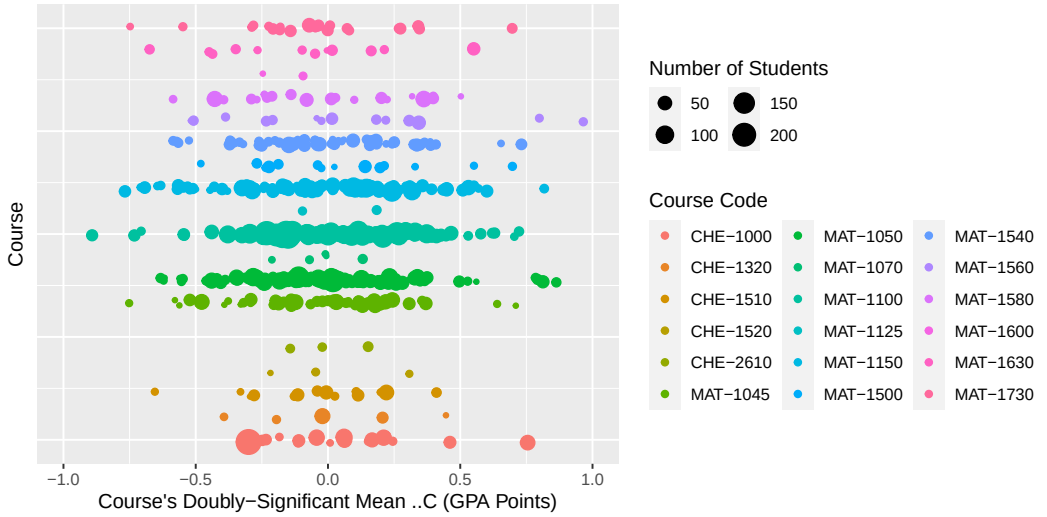

Figure 18: Scatterplot of all mean Delta C's for experiment two

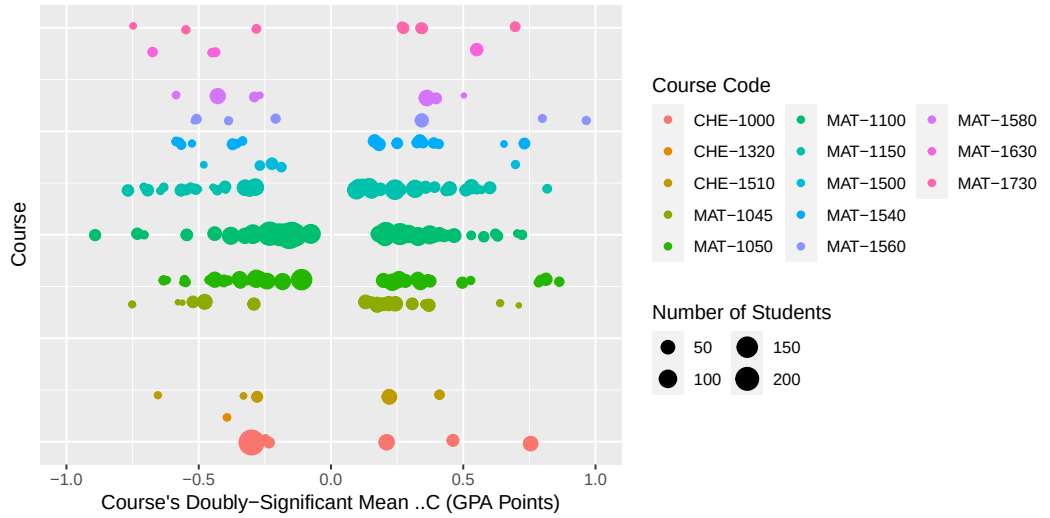

Figure 19: Scatterplot of doubly-significant mean Delta C's for experiment two

disappearing from the legends between figures 17 and 18 indicate that there were no faculty with doubly-significant mean ΔC's in that course. Specifically, while CHE-2610 and MAT-1070 had faculty-course combinations wherein at least eight students started none of these combinations were doubly significant. In experiment one the mean ΔC statistic was covariant with sample size. To get some context regarding the relationship between mean ΔC and sample size for experiment two, let's look at figure 19.

As in experiment one, the deviation of mean ΔC for faculty-course combinations in experiment two is covariant with sample size. From a macroscopic perspective, it seems like the data take values under an exponential decay or a very tall sideways Gaussian. The volatility evens out around the 50 student mark down to an average mean ΔC of ± 0.25 as in experiment one. There are

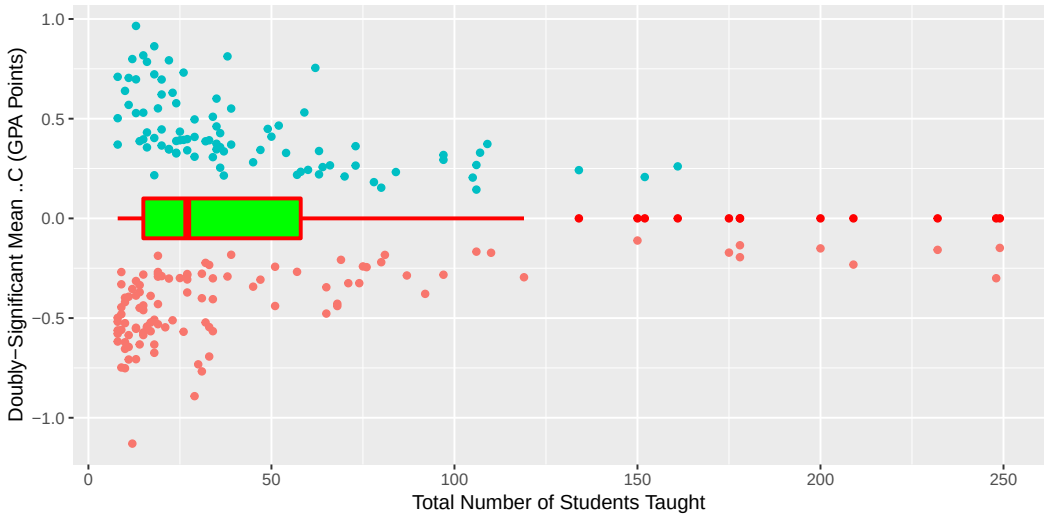

Figure 20: Scatterplot of each doubly significant faculty-course combination's mean Delta C against the number of students that passed filtration and started in it. Colored by effect polarity from the mean. Boxplot provided for summary of sample size distribution.

more data with lower sample size because we have increased the number of strata without increasing the number of viable students. Given more data, it is likely that these radical values would normalize towards zero.

Delta W Results

The definition of ΔW given by equation (2) involves the expected withdraw rate W_E and the observed withdraw rate W_O. The expected withdraw rate in experiment one is the probability 0.238, calculated as the total number of withdraws in the data set divided by the total number of observations. That is, it is the mean withdraw rate among all students between 2010 and 2017 in OCC's Math and Chemistry departments. The observed withdraw rate

is calculated as the total number of students that withdraw from *any* of a particular professor's courses divided by the total number of students taught over all courses. In experiment two W_E is set to the mean withdraw rate for students in the germane course, while W_O is the withdraw rate in a particular course code taught by a faculty member. To reiterate, a positive value of ΔW corresponds to a professor having a lower withdraw rate than the mean, and a negative value corresponds to a higher withdraw rate than the mean. This orientation is selected for ease of comparison with the mean ΔC statistic.

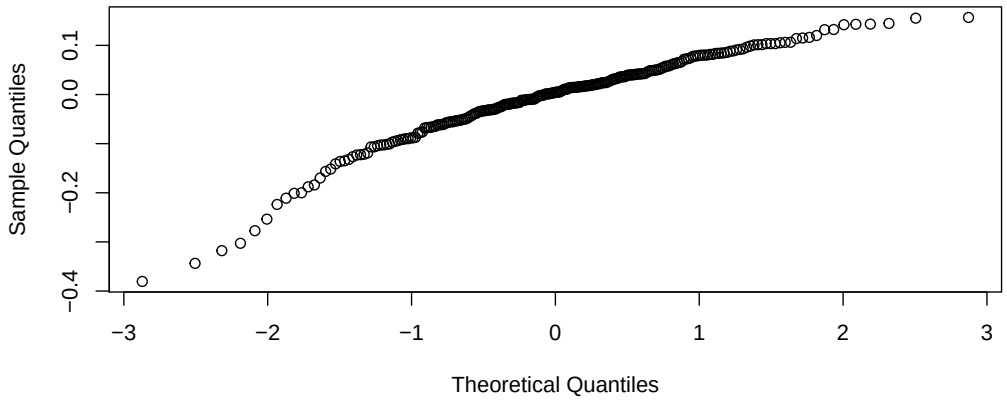

Figure 21: Normal probability plot of Delta W over all faculty for experiment one

Figure 20 is a normal probability plot of ΔW's distribution in experiment one, and figure 21 shows the distribution in experiment two. Both graphs demonstrate a slight left-skew for data approximately to the left of the -1 theoretical quantile. Observations of ΔW greater than the -1 theoretical quantile fit a normal distribution quite well. Regardless, usage of the one sample t-test gen-

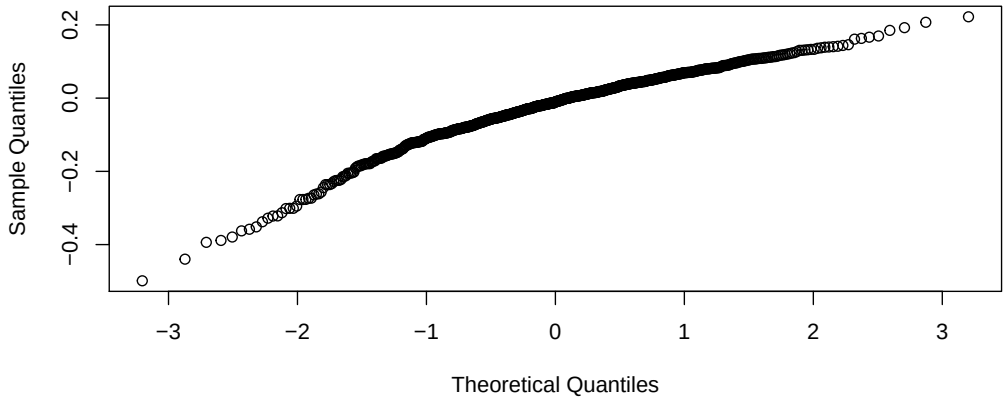

Figure 22: Normal probability plot of Delta W over all faculty for experiment two

erated confidence interval is ill-advised due to the requirement that data come from a normal distribution. Data will only be tested for significance using the bootstrap confidence interval given by equation (9). We stipulate that valid observations of ΔW possess at least eight withdraws so that data are sufficiently rich. This is likely overkill, but simply requiring eight students is not sufficient. Considering the overall withdraw rate of 0.238, the probability of not observing a single withdraw within eight students (which I denote $P(0 \in 8)$) is given by:

$$P(0 \in 8) = P(0 \in 1)^8 = (1 - 0.238)^8 = 0.114$$

In other words, the p-value of a faculty that obtains no withdraws within eight students is 0.114. Because we set $\alpha = 0.05$, we would automatically fail to

reject the H_0 from equation (11). This is clearly unacceptable. Requiring thirty to fourty students is standard guidance for usage of the central limit theorem, but as figures 20 and 21 show, ΔW's theoretical restriction on the domain $[-1, 1]$ induces non-normality onto its distribution. (9, 10) We therefore cannot use the approximate z-test, nullifying the utility of equation (4). The bootstrap confidence interval generated using the quantile() function is the fallback plan, and so we once again concern ourselves with a measure of sufficient richness of data. Michael R. Chernick recommends a larger number of unique data points for binomial data than for highly continuous data to ensure sufficient richness. (13) We've been using the number eight throughout this case study, so we will use a minimum eight withdraws as an aesthetically pleasing arbitrary benchmark for sufficient richness of data. The overall withdraw rate is 0.238, and so the expected class size that contains at least eight withdraws is given by $E[N] \geq \frac{8}{0.238} = 33.6$. The mean course with at least eight withdraws will have at least 33.6 students. For each additional withdraw, the expected class size increases by very roughly five students.

Why is the ΔW distribution left-skewed? One reason may involve the numerical definition of withdraw rate. Withdraw rate is a probability, so the overall mean withdraw rate can theoretically take any value on the set $[0, 1]$. The ΔW statistic must then have a measure of length one somewhere on the set $[-1, 1]$ depending on the observed mean W_O. In experiment one W_O is 0.238, which means the empirical ΔW distribution will have domain $[-0.762, 0.238]$. There is much more room for negative effects than positive effects, and so given a large enough sample there will be disproportionately many negative

ΔWs. The quantity of outliers is a function of sample size, so we expect the empirical distribution of experiment two ΔWs to be more skewed than that of experiment one. As we will show later in figure 32, the bounding of ΔW does not explain all skew behavior. There is either a confounding variable acting on ΔW, or withdraw rate might genuinely come from a non-normal continuous distribution.

Delta W Experiment One Results

Table 11: Number of faculty in experiment one that had at least eight students withdraw from all their courses. We can use bootstrap confidence intervals on the 245 professors with at least eight withdraws.

Sufficient.Sample.Size	n
FALSE	22
TRUE	245

Table 12: Results from bootstrap test. The DeltaWs of 118 faculty were not significantly different from the overall mean, while 136 were.

Significant	n
FALSE	117

Significant	n
TRUE	128

Table 10 shows that in experiment one, of the 263 faculty analysed, 245 survived the requirement of eight students withdrawing from their courses. Note that there were four faculty in Dr. Eckstrom's dataset that did not have any withdraws and so cannot be theoretically evaluated for significance using our bootstrap confidence interval. Three of the four professors listed pass the $p = \frac{\alpha}{2} = 0.025$ benchmark for the rejection of H_0. While they will not be included in the following graphics among the faculty that had at least eight students withdraw, it is important to note that there is an error in assuming that none of the faculty that had fewer than eight students withdraw had a significant ΔW. It is possible to prevent this small data loss by swapping to a type I bootstrapping as outlined in the statistical background section, but for sake of assumption uniformity we will persist with the eight withdraw requirement. Table 11 shows the results from the bootstrap significance tests onto the faculty that had at least eight students withdraw. In experiment one 127 out of the 245 faculty tested had ΔWs that were significantly different from zero at the $\alpha = 0.05$ level. We will now create the same graphics for the ΔW statistic as were created for mean ΔC.

As was predicted by figure 20, the distribution of ΔWs in experiment one has a slight left-skew. Figure 22 shows the overall distribution, and figure 23 shows the distribution of only those ΔWs that were significantly different

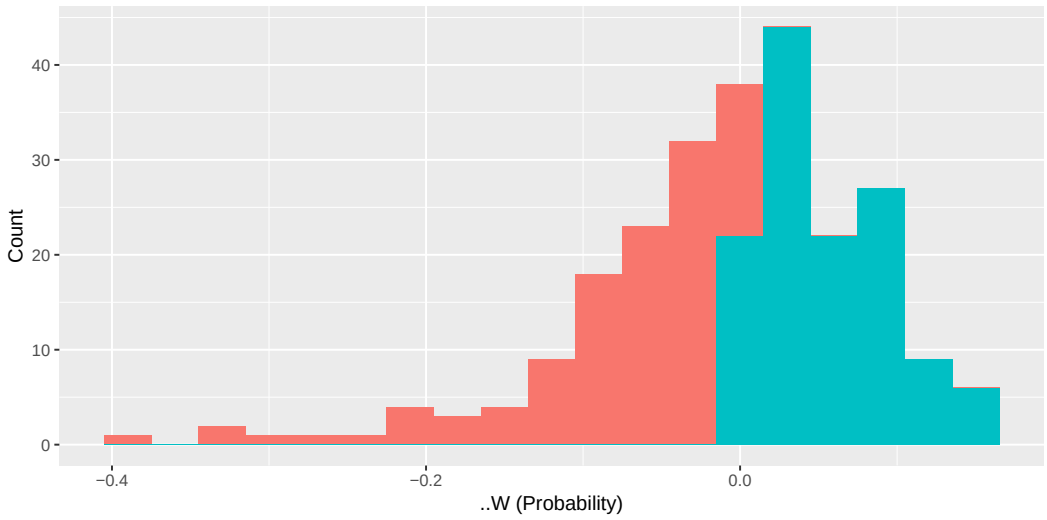

Figure 23: Histogram of all experiment one Delta W's, colored by effect polarity from the mean.

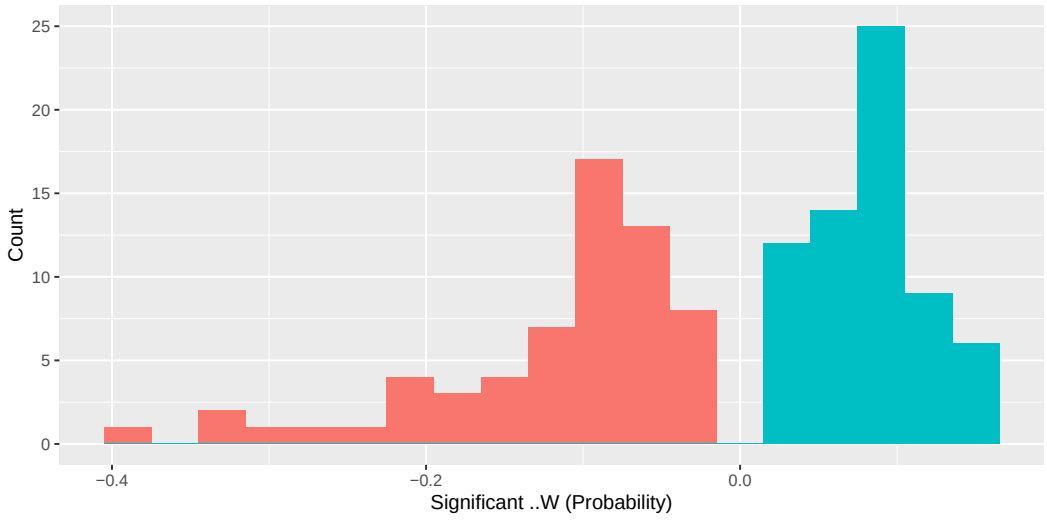

Figure 24: Histogram of only those experiment one Delta W's that were significantly different from the mean, colored by effect polarity from the mean.

from zero. The left-skew will be more pronounced in experiment two when there are more observations to work with. Now we indulge in some summary statistics regarding ΔW experiment one.

In experiment one the ΔW statistic had an overall mean of -0.008, with standard deviation 0.089. Similar to the mean ΔC we expect the overall mean ΔW to be zero with some wiggle room, where divergence from zero is created by random variable discreteness and the removal of some observations. Faculty with significantly positive ΔW values had mean ΔW 0.083 with standard deviation 0.033. Those faculty with significantly negative ΔWs had mean -0.117 and standard deviation 0.082. The significantly negative mean is farther from zero than the significantly positive mean, and the standard deviation for negative ΔW's was larger than the standard deviation of the positive ones. These statistics taken together provide a clear description of a left-skew.

Faculty taught a mean total of 521 students with a standard deviation of 594 students. As was the case in both experiments on mean ΔC, the distribution of sample size is very right-skewed. The deviation of ΔW from zero shown in figure 24 follows a similar pattern as shown in figures 14 and 19. The volatility of ΔW as the number of students approaches the minimum eight is most likely an illusory effect caused by the growth of standard error as $N \to 0$.

Delta W Experiment Two Results

In experiment one, a ΔW of zero corresponded to a withdraw rate of 0.238. In experiment two the expected withdraw rate ($\Delta W = 0$) for each faculty-course combination is the overall withdraw rate *of the course*. Figure 25 shows the

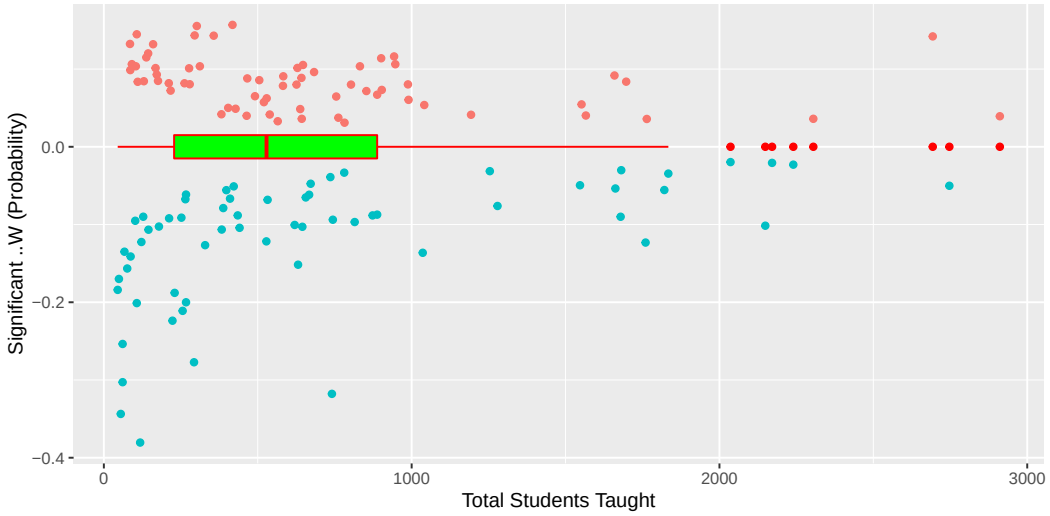

Figure 25: Scatterplot of each significant faculty's experiment one Delta W against the number of students that took their courses. Colored by effect polarity from the mean. Boxplot provided for summary of sample size distribution.

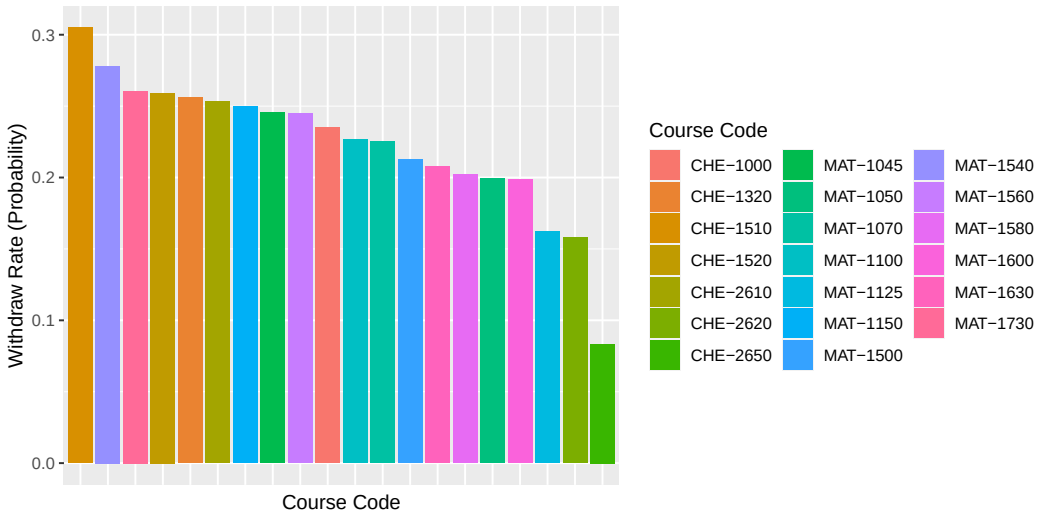

Figure 26: Bar chart of all course withdraw rates.

overall withdraw rate per course calculated as the total number of withdraws divided by the total number of course attempts. In experiment two, a ΔW of zero will indicate that a faculty-course combination had withdraw rate equal to the mean course withdraw rate given by figure 25. Table 12 shows that 736 out of a total 1044 analyzed faculty-course combinations had at least eight students withdraw. Of these 736, only 287 faculty-course combinations resulted in a ΔW that was significantly different than the mean at the $\alpha = 0.05$ level. As in the analysis of the mean ΔC statistic, the lower H_0 rejection rate in experiment two is due to the lower sample size per stratum.

In experiment two the ΔW statistic had an overall mean of -0.024, with standard deviation 0.101. Faculty with significantly positive ΔW values had mean ΔW 0.095 with standard deviation 0.036. Those faculty with significantly negative ΔWs had mean -0.16 and standard deviation 0.092. As in experiment one, these statistics confirm the presence of a left-skew.

Table 13: Number of faculty-course combinations from which at least eight students withdrew. We can use bootstrap confidence intervals on the 736 professors with at least eight withdraws.

Sufficient.Sample.Size	n
FALSE	318
TRUE	736

Table 14: Results from bootstrap test. The DeltaWs of 449 faculty-course combinations were not significantly different from the appropriate course mean, while 287 were.

Significant	n
FALSE	451
TRUE	285

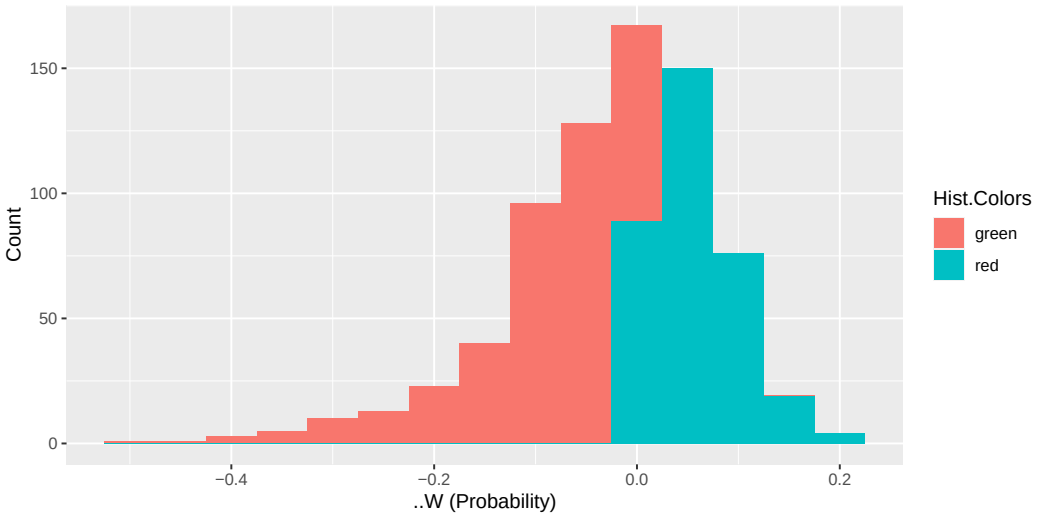

Figure 27: Histogram of all experiment two Delta W's, colored by effect polarity from the mean.

As was the case in experiment one, the left-skew predicted by figure 21 is manifest. The ΔW distributions displayed in figures 26 and 31 are nigh identical to those of figures 22 and 23 respectively. Both figures 26 and 27 are smoother

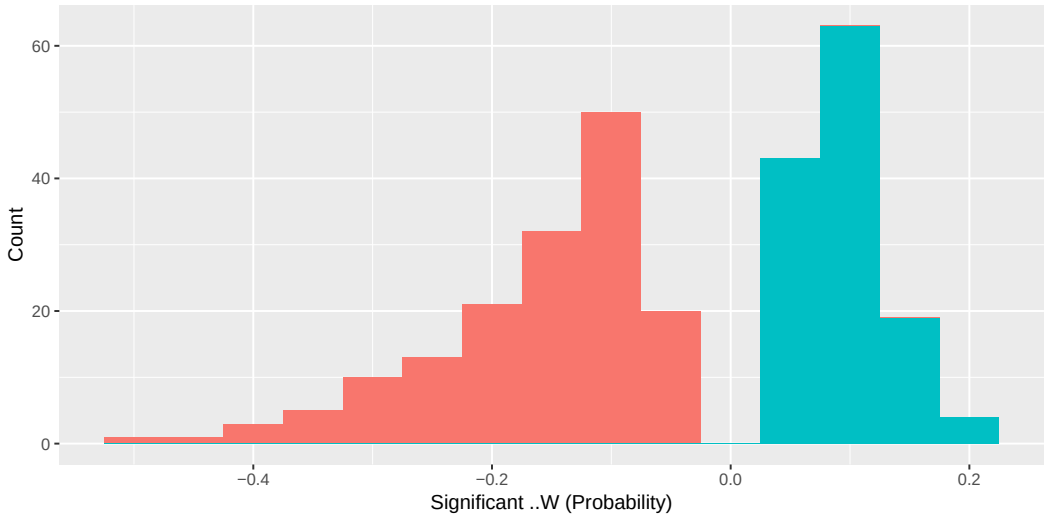

Figure 28: Histogram of only those experiment two Delta W's that were significantly different from the mean, colored by effect polarity from the mean.

than their counterparts, which is to be expected by the larger number of random variables. Because overall histograms don't show how individual courses differed, we now examine scatterplots akin to figures 17 and 18.

In this paper, Δ statistics are fundamentally measured as difference from mean values. The mean ΔC statistic was less subject to the effect of other faculty than ΔW because variance in student's individual ΔCs were averaged. In contrast, the experiment two ΔW a zero-sum game. As is most clearly demonstrated by CHE-1610 and CHE-1620, if a faculty-course combination has a negative ΔW, there must be at least one faculty-course with a positive ΔW. Figure 29 obfuscates this fact for courses such as MAT-1600 and MAT-1630 because all the faculty with better than average ΔWs were not significantly different from zero. Figure 29 does provide a useful visualization

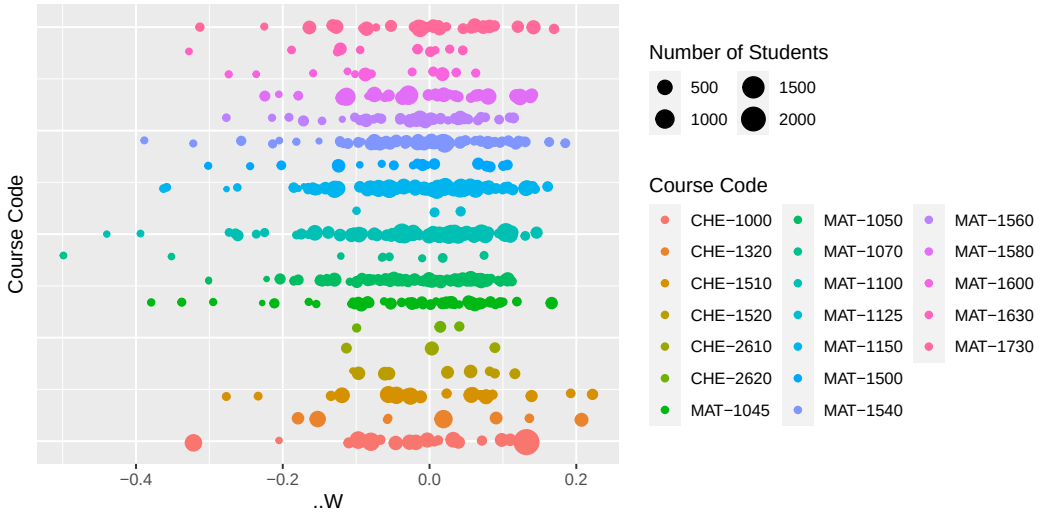

Figure 29: Scatterplot of all Delta Ws by course code. Point sizes correspond to the number of students taught in faculty-course combination.

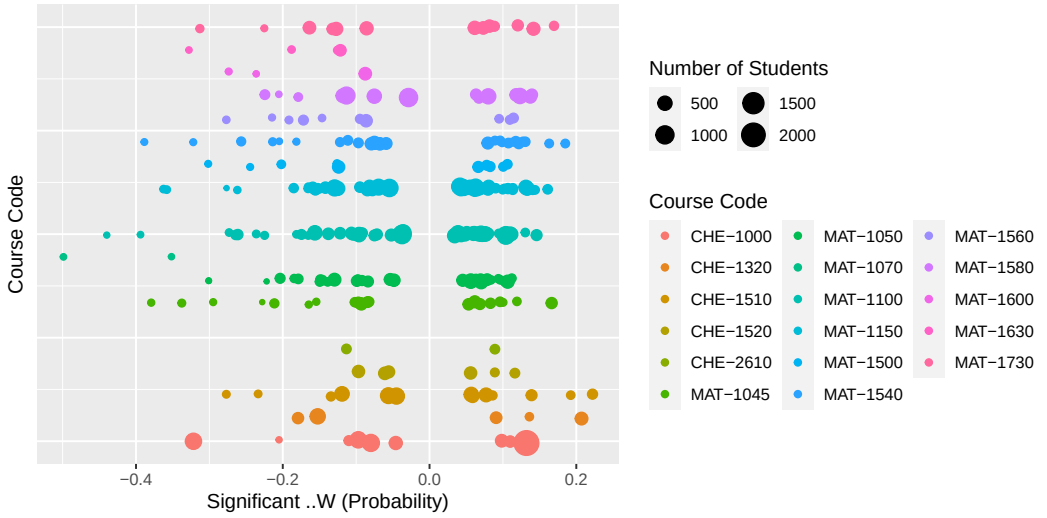

Figure 30: Scatterplot of only those Delta Ws that were significantly different than zero by course code. Point sizes correspond to the number of students taught in faculty-course combination.

of significant ΔW variance by course. That said, my preference is the holistic accounting shown in figure 28.

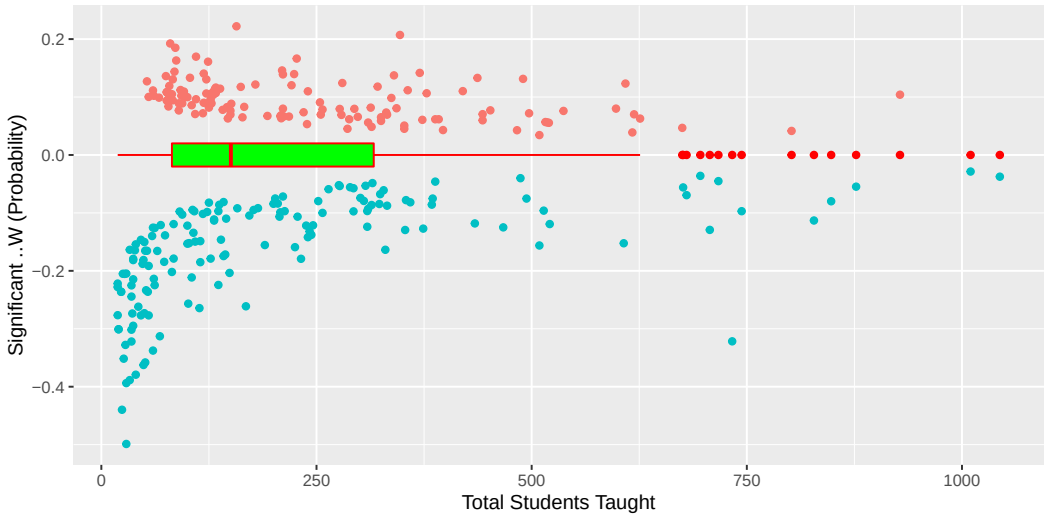

Figure 31: Scatterplot of each significant faculty-course comination's experiment two Delta W against the number of students taught. Colored by effect polarity from the mean. Boxplot provided for summary of sample size distribution.

The experiment two ΔW statistic is the most volatile of all. Unlike in figure 19, the distribution no longer appears as a sideways-Gaussian. The left skew induced by centering ΔW about each course's overall withdraw rate should cause more negative ΔWs to appear. These very negative statistics seem to vanish before the 250 student mark. The positive ΔWs are interestingly not very volatile because they are bounded at a low ceiling. The most remarkable observation was the negative outlier near the 750 student mark, belonging to faculty number 894670 teaching CHE-1000.

Does Delta C predict Delta W?

We created the ΔW statistic in order to create a measure of faculty performance that may not be covered by the cumulative GPA progression of their students. How are the two statistics related? Figures 30 and 31 are graphs of mean ΔC vs ΔW. Because ΔW and mean ΔC each have unique filtration conditions, there is no guarantee that all faculty will have both statistics. Instuctors with only one statistic cannot tell us anything about the correlation of ΔW and mean ΔC, so figures 30 and 31 only include professors with both a mean ΔC and a ΔW. A missing value analysis is warranted, but there are so few data points that no general inference is easily reached. Along with the plots, R is instructed to output a summary of the linear model calculated using the data in each plot.

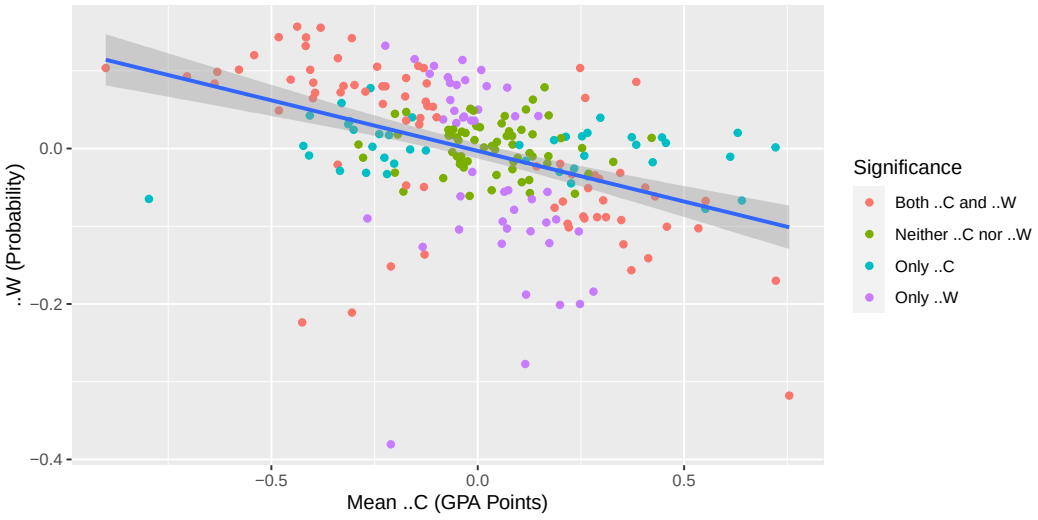

Figure 32: Mean DeltaC Vs. DeltaW for all faculty in experiment one, colored by significance levels.

```
## $coefficients
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) -0.00309    0.00496  -0.623 5.34e-01
## Mean.DeltaC -0.13014    0.01765  -7.374 3.11e-12
##
## $adj.r.squared
## [1] 0.19
```

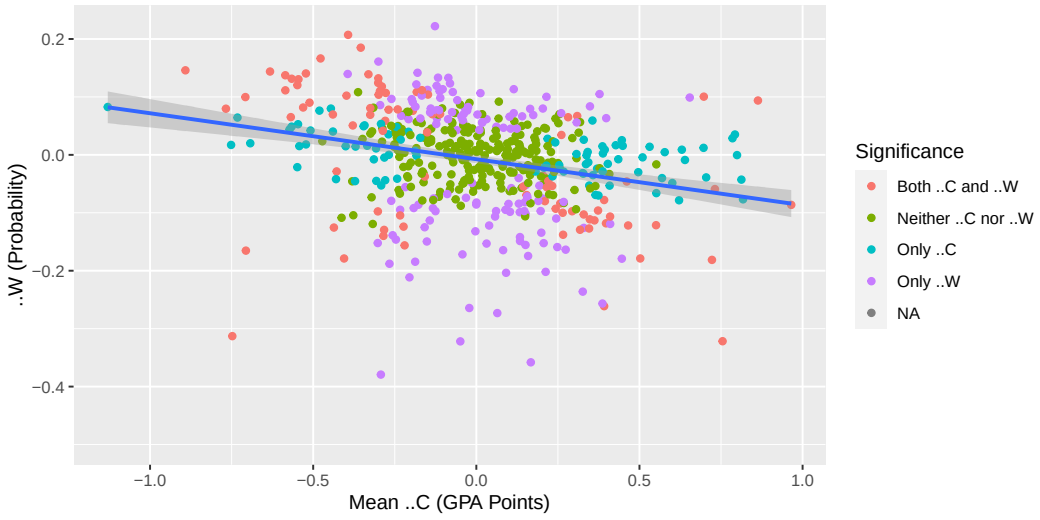

Figure 33: Mean DeltaC Vs. DeltaW for all faculty-course combinations in experiment two, colored by significance levels.

```
## $coefficients
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) -0.00752     0.0035   -2.15 3.20e-02
## Mean.DeltaC -0.07932     0.0119   -6.69 5.65e-11
##
## $adj.r.squared
```

```
## [1] 0.0749
```

According to the adjusted R-squared value for experiment one, approximately 19% of the variance in ΔW can be attributed to variance in mean ΔC. The linear model predicts a $-0.130x$ decline in ΔW for an x-point increase in mean ΔC. For experiment two the adjusted R-squared is markedly lower, suggesting that only 7.5 of variance in ΔW is attributable to variance in mean ΔC. This lower adjusted R-squared can be explained by experiment two having higher volatility due to lower sample sizes. The experiment two model predicts a more moderate $-0.079x$ point decline in ΔW for an x-point increase in mean ΔC. Both statistics are theoretically centered about zero, so it is no surprise that the intercept of both models is within 10^{-3} the origin.

"Good" effects are measured by the same polarity, so in context the models suggest that faculty that have higher than average effects on final cumulative GPA tend to have worse than average withdraw rates. This appears counter-intuitive, but may be explained by selection bias. As a reminder, the filtration conditions on ΔC are:

1. The student must only take one course in their first term,
2. The student must not withdraw from their first course, and
3. The student must take at least two graded terms.

Students that withdraw from the first course do not go into the calculation of mean ΔC. Students that withdraw tend to do so because they are failing (or have unrelated personal conflicts) that may indicate a propensity for low GPA

scores. As more students withdraw, an instructor is left with those that for whatever reason are more able to attend to their studies. Such students may be more motivated than their peers or simply able to devote more time to school, and thus have a better cumulative GPA slope. The adjusted R-squared values may then be reframed as partial measures of the selection bias. We cannot claim that there is precisely an average 19% bias in experiment one because such would discount the possibility of *true* correlation between the ΔW and mean ΔC statistics.

Error Analysis

Table 15: Low estimates of the number of possible values for cumulative GPA for students that take one course per term. Estimates for terms one through six including proportion of students that were enrolled in OCC's Math and Chemistry departments for that number of terms.

Student.Term	Prop.Students	Possible.Cum.GPAs
1	0.512	12
2	0.245	34
3	0.117	86
4	0.065	149
5	0.031	212
6	0.016	284

The bootstrap and one sample t-test have error in different ways. The one sample t-test assumes that: (9, 10)

1. Data come from multiple measurements of a continuously distributed random variable,
2. Any subset of data (an event) is a simple random sample from the overall distribution,
3. The random variable which is being tested has a Gaussian distribution,
4. The sample size is sufficiently large (generally n >= 3), and
5. The random variable has a finite constant standard deviation.

For both experiments one and two, our data does not satisfy condition one. Cumulative GPA is a fundamentally discrete beast. For a student with a cumulative GPA c, the permitted values for cumulative GPA after one more term depend on c, the allowed course credits, and the allowed GPA's obtainable in a course. For example, for a student that took a single course in their first term (which is the filtration assumption) there are only twelve possible values for cumulative GPA: 0, 1, 1.3, 1.7, 2, 2.3, 2.7, 3, 3.3, 3.7, 4, and the NA which corresponds to a withdraw. Table 13 simulates a low estimate for the total possible number of distinct cumulative GPAs as a function of term enrolled in OCC's Math and Chemistry department. The simulation I used does not take into account course credits because most courses have the same 4-credit valuation. The second term already gives 33 possible values for cumulative GPA. In my mind $33 values$ is not quite approximately continuous considering they

are spread across the range $[0, 4]$ with 0.12 GPA points between each discrete possibility. The 86 options that corresponds to three terms is much better, but only 49.7 of the students used in significance testing were enrolled for at least three terms. Additionally, there is no guarantee that students enrolled for three terms recieved grades for three terms, nor is there any guarantee that all possible cumulative GPAs are equally likely. In short, the discreteness of cumulative GPA is a major source of error for both experiments one and two.

The bootstrap principle does not require that its data come from a continuous variable to create a confidence interval, nor must the data come from an approximately normal population. (10) It does however need sufficiently many distinct values in order to accurately depict the sampling distribution. (10, 13) For example, if all eight of a faculty's students obtained $\Delta C = 0$ by some divine providence, the bootstrap would be unable to simulate any possible mean ΔC other than zero, resulting in a mean ΔC of zero with a *standard error of zero.* While this never occurred experimentally in the analysis of mean ΔC, I did find four cases of a faculty-course combination wherein no students withdrew. The bootstrap sampling distribution cannot theoretically include $\Delta W = 0$ in such cases because no resampling will generate a withdraw. (10) As said before, the eight student minimum helps mitigate such errors. (13)

The probability that any student would have a negative ΔC is about $\frac{1}{2}$. The probability that all eight students of a faculty with the minimum acceptable sample size have negative ΔCs is about $(\frac{1}{2})^8 = \frac{1}{256}$. Doubling this quantity gives the probability that all eight students have either positive or negative

ΔCs as $\frac{1}{128}$. In experiment two there are 605 faculty-course combinations that get tested by the bootstrap confidence interval, so we would expect roughly 4.72 instances of a faculty being randomly assigned eight students with matching ΔC polarity. The variance in random assignment of mean ΔC clearly must decrease with sample size. Considering figures 14 and 19 show convergence of mean ΔC towards zero as sample size increases, we have evidence that a faculty's mean ΔC may be discounted for low sample sizes. The art is determining what the minimum acceptable sample size is for a "real" mean ΔC effect.

Usage of a linear mixed-effects model may not be appropriate for the ΔC statistic. No affine can properly predict a student's cumulative GPA progression through an arbitrary number of terms. If a student failed their first course and that course had a mean negative cumulative GPA slope after adjusting for the number of terms the student was enrolled in the Math and Chemistry departments, the student's ΔC would be positive even if they failed every single course they were enrolled in. Similarly, a student that received a perfect 4.0 cumulative GPA would have a negative ΔC if the mean course slope were positive. These scenarios imply that stratification by first GPA received may be warranted. The first GPA received is highly covariant with first course and so there may be an issue of double counting. GPA received in the first course may be a suitable replacement for the first course stratification, but that would be a different project.

Faculty beyond the first encountered also have an effect on student outcomes.

The ΔC statistic operates on the assumption that future faculty effects average out over sufficiently many possible future instructors. This assumption is flawed for faculty that had few students start in their course due to high variance. Higher-level courses have fewer instructors because fewer students take the courses. For example, the effect of future faculty may not be random for students whose first course in the dataset was CHE-2610, because there were only seven unique faculty that teach CHE-2620 between 2010 and 2017. Moreover the maximum number of faculty teaching CHE-2620 in a given term was three; there are not a continuous distribution of future professor effects for CHE-2620. If the three faculty assigned different mean GPAs in their courses, a bias would be created on the mean ΔC statistic for CHE-2610 instructors.

Table 16: Frequency and relative frequency table of number of terms enrolled in OCC's Math and Chemistry departments for students starting in CHE-2620.

max.terms	Num.Students	rel.freq
1	54	0.628
2	24	0.279
3	3	0.035
4	2	0.023
5	2	0.023
7	1	0.012

Table 17: Frequency and relative frequency table of number of terms enrolled in OCC's Math and Chemistry departments for students starting in MAT-1540.

max.terms	Num.Students	rel.freq
1	1966	0.476
2	993	0.240
3	514	0.124
4	315	0.076
5	168	0.041
6	68	0.016
7	50	0.012
8	28	0.007
9	13	0.003
10	6	0.001
11	5	0.001
12	1	0.000
14	1	0.000
16	1	0.000
17	1	0.000

Faculty which predominately teach higher level courses are less likely to be assigned a mean ΔC because the mean ΔC statistic requires at least eight students to pass the imposed filtration conditions and start with the professor.

Table 14 gives the frequency and relative frequency of total terms enrolled in the Math and Chemistry departments for those students that started in CHE-2620. Table 15 does the same with students that began in MAT-1540. CHE-2620 is the highest level chemistry course offered, whereas MAT-1540 is OCC's equivalent of high school algebra II. Not only does CHE-2620 contain many fewer cases overall, a larger proportion of students that started in CHE-2620 took only one term as compared to MAT-1540. It must be reiterated that because OCC is a community college, students are likely to transfer to a school offering a four year degree after completing necessary prerequisites. There is no way to calculate the effect CHE-2620 faculty had on students that transferred out of OCC not because they had no effect, but because the grades received are not recorded in Professor Eckstrom's data. The mean ΔC statistic is then given more credence within low level courses like MAT-1540 in which students are both: 1) likely to start in, and 2) likely to take more courses after. This error is amplified in experiment two, contributing to the higher rejection rate shown in table 11 than that of table 9.

The primary source of error in the ΔW analysis originates from the discrete domain on which the statistic is measured. The ΔW distributions shown in figures 22 and 26 have a evident left-skew caused by uneven opportunity for outliers. That said, it can be shown that ΔW does not come from a normal distribution. Figure 33 shows the histogram of a normal distribution with the same mean and standard deviation as the experiment one ΔW distribution. Moreover, the normal distribution is restricted on the domain $[-0.762, 0.238]$ just as the experiment one ΔW distribution is. Figure 32 clearly demonstrates

that while the upper tail behavior of the bounded normal mimics that of ΔW, the lower tail behavior fails to capture the same trend. In particular, faculty withdraw rates tend to be more extremely negative than would be predicted by a normal distribution. The error induced by the limitation on the domain of ΔW can then only be said to affect faculty with lower-than-average withdraw rates.

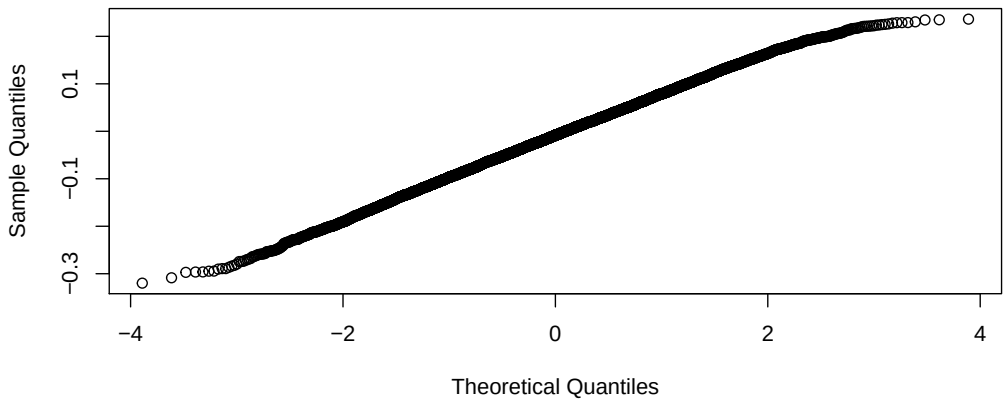

Figure 34: Normal probability plot for simulated normal distribution with the attributes of the emperical experiment one DeltaW distribution.

Finally, I must provide a warning against my own statistics. Any arbitrarily created statistic may become meaningless when subjected to too many stipulations. Both ΔW and mean ΔC have their own caveats which are fundamentally different. By restricting these statistics, we may be hiding trends in the data that are far more illuminating. It is moreover possible to apply arbitrary filtration conditions onto data to produce any myriad desired results. We must then be careful in assessing our own selection biases.

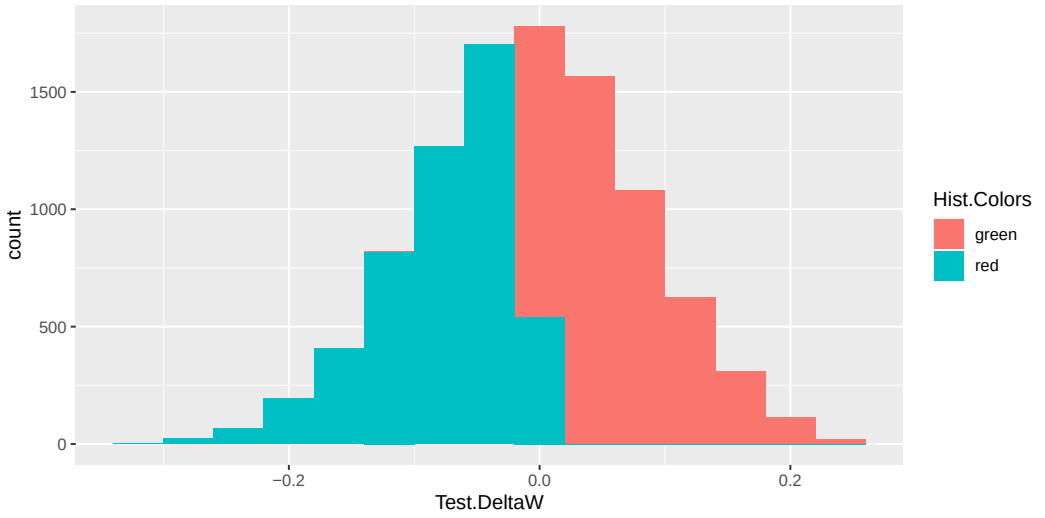

Figure 35: Histogram of simulated normal distribution with the attributes of the emperical experiment one DeltaW distribution. Histogram is colored by effect polarity from the mean.

Conclusion

Professor Eckstrom's dataset was restricted to the Math and Chemistry departments of Oakland Community College. The ΔW and mean ΔC statistics may entirely change when calculated using data for *all* a students courses. This experiment should be repeated using data from all departments at OCC. Additionally, it would be beneficial to confirm whether the same trends hold between community colleges. Moreover, data from a four year college would give a more clear and more widely applicable measure of first faculty effect on final cumulative GPA.

I would love to one day carry out this analysis on K's student data. One issue with analyzing K's student data lies in anonymity and richness of data. For example, Dr. Barth is the only professor to teach differential equations and Dr. Nordmoe is the only professor to teach mathematical statistics. Not only would these teachers be easily identifiable by an analyst, the course effects would generally be zero (and therefore meaningless) because there are no faculty to compare against.

Another good statistic that I considered measuring was mean ΔN, the average number of additional course codes a certain first instructor's students took in the same department. This statistic would not include repeats for course attempts that resulted in fails or withdraws, and as such is a partial measure of how a professor inspired their students to more thoroughly investigate their subject. ΔN would be an excellent SIP in and of itself for students looking to

analyse a similar dataset.

In starting this project I had hoped to define a few numerical indicators of instructor effect on student outcomes. A desire to understand the effect I was having on my own tutoring students developed into an opportunity to investigate a rich and complex dataset. In the process of constructing my analysis I learned valuable skills in R programming, tidyverse data manipulations, and grit. I was inspired to pursue a masters degree in data science, and am excited to pursue it as a career.

Appendicies

Appendix A: R Programming

As previously stated, when I first began this project I had never taken a programming course. A large chunk of the project boiled down to creation and manipulation of new variables through the tidyverse method, but there was still many pieces of knowledge I lacked to make the analysis run. Coursera offers many excellent online courses on a wide variety of subjects, among them computer science and data science. To learn the programming skills I required to analyse Professor Eckstrom's data, I enrolled in and completed the first two modules of the *Data Science: Foundations Using R Specialization* offered by Johns Hopkins University on the Coursera platform: The Data Scientist's Toolbox, and R Programming. My certificates and grades I received are available in the appendix along with synopses and work product from the two courses. This section is a summary of my learning from these courses and my previous exposure to R, intended as a introduction to all necessary concepts used in my program. For readers new to the R language, I recommend keeping the R Documentation open for reference when learning the syntax and usage of new functions. The course instructor Dr. Peng's book *R Programming for data science* is the singular source for Appendix A.

The Atomic Data Types

In the R language, there are five atomic data types, also called class invariants:

1. Numeric data, often also referred to as double-precision data, are real numbers. They can have up to fifteen significant digits, but generally are limited by observation to many fewer. In RMarkdown it is common practice to include the option "digits = k", which restricts the printing of doubles to k decimal points. Numeric data have the additional option of having the value NAN (for not a number), which most often corresponds to undefined in mathematical language. For instance, $\log(-1)$ yields the output NaN.

2. Integer data can only be real integers, and are useful in programming aspects such as indexing and preventing rounding errors. Integer math is a common focus of number theory.

3. Character data, or strings, are the broadest class. Any set of alphanumeric+ characters can be represented in a *string* as a character object. Packages such as stringr are available to manipulate character variables with plentiful functions equipped for regular expressions.

4. One of if not the most important data type for programming in R is the logical, or Boolean. A logical datum either has the value TRUE or FALSE. When two logicals b_1 and b_2 are compared over equality - i.e. $b_1 = b_2$ - R will output TRUE if both data have the same value. We will momentarily investigate the behavior of logical data. As will be apparent when discussing subsetting, the ability to programmatically compare large quantities of logical data allows for fast and convenient indexing, sorting, and filtering.

5.The final class in the R language is complex, an element of which has the form $a + bi$, where a and b are numeric data. I personally never used complex data beyond the single Coursera quiz that asked for the identification of a complex datum among a list of options.

One last important note - data can be missing in any class. A missing value is denoted with NA (for not available). The NA can therefore be represented in each class, and converted using the as.xxx() family of functions. R treats NA values as if they did have a value, but that the value is not known. For instance - the mean of the numeric vector c(.5, 2, NA) is NA with class numeric, because R cannot determine the third element's effect on the average but assumes it was a number due to how vectors are defined.

Assignment

The cornerstone of any function-oriented programming project is the ability to save a created object to memory. A user can accomplish this by using one of R's assignment methods. There are five different assignment operators:

1. The equals operator "=" which cannot be used within a function call,
2. The left assignment operator "<-" and the right assignment operator "->" which *can* be used within a function call, and
3. The left and right super-assignment operators "«-" and "-»", which store the created object to the parent frame. Super-assignment is most commonly found in functions where a useful object is created, such as in the process of caching to prevent the duplication of effort.

Vectors

On that note, let's discuss one of the primary weapons of the R language - vectors. R can store any sequence of data from a single class as a vector and do operations on it. Such operations are performed element-wise unless otherwise specified. The vector's class will be the same as that of its constituent data. There are several ways to create a vector including the concatenation operator c(), the sequence generator seq(), the replicate function rep(), the colon operaton :, and extracting a column out of a dataframe (a rectangular matrix with column names) either by using pull() or by subsetting. Individual elements can have names, for instance:

```
vector.names <- c("Jimmy", "Sammy", "Katrina")
my.vector <- c(17, 18, 17)
names(my.vector) <- vector.names


my.vector
```

```
##   Jimmy   Sammy Katrina
##      17      18      17
```

Vectors can sometimes be changed from one class to another through a process called coercion. The family of functions as.class() performs this conversion. Below is some sample output demonstrating the coercion capabilities of different classes using three demo vectors: num for numeric, chr for character, and log for logical:

```r
## Define demo vectors. num is a numeric vector, chr is a


## character vector, and log is a logical vector.
num <- c(-1, 0, 1, 2)
chr <- c("a", "b", "0", "FALSE")
log <- c(TRUE, FALSE)


## Coercion of numeric vector to character
as.character(num)
```

```
## [1] "-1" "0"  "1"  "2"
```

```r
## Coercion of numeric vector to logical
as.logical(num)
```

```
## [1]  TRUE FALSE  TRUE  TRUE
```

```r
## Coercion of character vector to numeric
as.numeric(chr)
```

```
## [1] NA NA  0 NA
```

```r
## Coercion of character vector to logical
as.logical(chr)
```

```
## [1]    NA    NA    NA FALSE
```

```r
## Coercion of logical vector to numeric
as.numeric(log)
```

```
## [1] 1 0
```

```
## Coercion of logical vector to character
as.character(log)
```

```
## [1] "TRUE"  "FALSE"
```

When coercing a numeric object into the character class, each element becomes represented as a string and loses its ability to partake in arithmetic operations. Conversion from double to logical is possible, but one must be careful - only zeroes get mapped to FALSE, while all other numbers get mapped to TRUE unlike in some other languages. Character vectors are the least coercible. While each unicode character is stored in memory as a very large integer, R doesn't want to return these for user safety. As a result, coercion of a vector from the character class to the numeric class will result in a missing value unless R thinks the string already represents a number. Strangely, TRUE / FALSE values stored in a character vector will be mapped to missing values in this coercion, even though they can be coerced into numerics. Calling as.logical() will similarly fail to notice 0 as FALSE. If a vector's class is not clear, the class() method is an excellent tool, and can also be used programmatically to obtain the class(es) of an object. In my project I used the as.class() functions rather rarely, but they serve to elucidate the inner workings of the classes themselves.

Logic

Before moving on to subsetting vectors and lists, we first take a foray into R's evaluation of logical statements. The most straightforward way to do this is to look at some examples.

```
## Practice with logical data
TRUE == TRUE
```

```
## [1] TRUE
```

```
TRUE != TRUE
```

```
## [1] FALSE
```

```
FALSE < TRUE
```

```
## [1] TRUE
```

```
FALSE > TRUE
```

```
## [1] FALSE
```

The double equals is a binary operator that outputs TRUE i.f.f. both sides of the operator have the same evaluation. The less than and greater than operators work in a similarly intuitive manner. ! is the negation operator, which inverts the meaning of the logical operator it is paired with. Additionally a logical datum itself can be negated, such that !TRUE evaluates to FALSE. Note that the expression FALSE $<$ TRUE evaluates to TRUE and FALSE $>$ TRUE evaluates to FALSE because R considers FALSE to be the number 0 and TRUE to be the number 1. This feels less than intuitive because as seen above

any nonzero number coerces to TRUE under the method as.logical(). Just as $0 < 1$ gets a TRUE, when character data are compared over the binary operators, R considers the lexicographic ordering. This is worth exploring, but not relevant to my project. More pertinent is the evaluation of vector comparisons. Let's do another example.

```
## Define some vectors to compare with different lengths.


## Vector a contains the sequence 1, 2.
a <- c(1, 2)
## Vector b contains the sequence 1, 2, 3, 4, 0.
b <- c(1:4, 0)


## Compare a and b over equality
a == b
```

```
## [1]  TRUE  TRUE FALSE FALSE FALSE
```

```
## Compare a and b over a formula
a + 2 == b
```

```
## [1] FALSE FALSE  TRUE  TRUE FALSE
```

```
## Use the which() function
which(b >= 3)
```

```
## [1] 3 4
```

R evaluates vector comparisons element-wise and outputs a vector with

length equal to the longest input vector. If the compared vectors are not the same length, R repeatedly replicates the shorter vector end-to-end until the lengths are equal. In the above example, R changed the expression $\{1, 2\} == \{1, 2, 3, 4, 0\}$ into $\{1, 2, 1, 2, 1\} == \{1, 2, 3, 4, 0\}$ before evaluating. The shorter length of 2 does not evenly divide the longer length 5 so R throws a warning, but continues with evaluation regardless. The output of any logical comparison is a logical vector; the previous comparisons were simply a special case where the input vector lengths were each one. $a == b$ returns a five element output vector, because the longer of the two input vectors had five elements.

The final complexity to logical evaluation in R are conditional expressions. R is equipped with three basic binary conditions: the and statement "&", the or statement "|", and the xor function xor(). A more complicated binary operator I use for the data analysis is the in operator "%in%", which checks whether the left object is within a range of values specified by the right object. These conditions work just as they do in set theory. "|" has low priority in R's evaluation queue while "&" has high priority so that conditions evaluate in accordance to the generally desired order of operations. Just like in mathematics, parenthetical statements are evaluated before their exterior for additional control over the evaluation queue. The which() function is the odd-code-out in this example, because it does not return a logical vector. Instead it takes a logical vector as *input* and returns the indices of the TRUE elements. This is particularly useful in subsetting. One final note, R allows for the user to define their own logical operators, a privilege I used only one time in conjunction

with the Negate() function to create a negated version of the "%in%" operator called "%notin%".

Subsetting

CRAN outlines three operators for subsetting sequences in R:

1. The single bracket operator $X[i]$, where X is some array and i is a set of indices.

Single bracket subsetting is generally used with vectors to extract the i^{th} elements of X. When used on a list, the single bracket will return the i^{th} elements as a list. The user can supply indices either as an integer vector or a logical vector. To see how this works, inspect the sample output below.

```
## We will be using my.vector for subsetting practice.
my.vector
```

```
##   Jimmy   Sammy Katrina
##      17      18      17
```

```
## Define some integer indices to subset
int.index <- c(2, 3, 4)


## Define some logical indices to subset
lgl.index1 <- c(TRUE, FALSE)


## More practical application of logical indexing
```

```
lgl.index2 <- names(my.vector) == "Jimmy" | names(my.vector) == "Katrina"


## Define some desired names for subsetting

names.index <- c("Jimmy", "Sam")


## Use the single bracket operator to subset using


## integer indices

my.vector[int.index]
```

```
##   Sammy Katrina    <NA>
##      18      17      NA
```

```
## Use the single bracket operator to subset using


## logical indices

my.vector[lgl.index1]
```

```
##   Jimmy Katrina
##      17      17
```

```
## Practical example of subsetting with logic. This


## returns the same output as previous example, but


## has a more legible index.
```

```
my.vector[lgl.index2]
```

```
##   Jimmy Katrina
##      17      17
```

```
## Use the single bracket operator to subset by name
my.vector[names.index]
```

```
## Jimmy  <NA>
##    17    NA
```

R uses 1-based array indexing, so when fed integer indices it will attempt to extract elements from the object based on the position considering the index 1 to be first position. When we told R to take 4^{th} element of my.vector it determined that there *should* be a 4^{th} element, but that there was insufficient data to determine its value. R returns NA for the fourth index of my.vector. If R instead is given a logical vector to use as indices, it decides it needs to do a logical comparison. R will first (if necessary) replicate the indices until the length of the index vector has at least the same length as the object to be subset; my.vector has three elements but lgl.index only has two, so R will turn $c(T, F)$ into $c(T, F, T, F)$. The elements corresponding to TRUE indices are returned. The true power of this concept arises from the combination of R's robust logical framework and element-wise operations, as can be seen in the second logical example output.

The single bracket operator can also be used to subset by name. When the single bracket is fed a character vector, R will look for an element whose name

is a literal match. One of the later methods allows for partial matching, but not the single bracket. If there is no element with a name matching a supplied argument R will return an NA of the appropriate class as stand-in. Note that R does not name the NA value with the assumed name from the index.

2. The double bracket operator $X[[i]]$.

Just as the single bracket variant is generally used on vectors, the double bracket is most often used on lists. Before discussing the double bracket method, we must first ask what is an R list? Lists are another common form of array that can be thought of as a hierarchical pyramid that can hold any type of object, including other lists. Just like vectors, list elements can have names. Lists are excellent organizational tools because they are not beholden to the stipulation that each element have the same class invariant; a two element list may for example contain one function and one dataframe. Because lists may contain a variety of information, a couple good ways to obtain a sense of the contents are the str() and glimpse() commands. I am personally partial towards glimpse due to its good behavior with other dplyr verbs.

The double bracket drops all formatting on elements - for a list this includes removal of the top level list specification as well as any names associated to the level. In the sample output below, my.list contains two branches. The first branch is another list of two elements, while the second is a numeric vector of length 1. By applying [[1]] once, R extracts the branch title Element_One. The second application of [[1]] extracts the contents of the first object stored in Element_One, which happens to be a function that returns the mean of an

input. By supplying a numeric vector to this subset R will output the mean of the numeric vector. In my project I use lists to store multiple stratified mixed-effects models, and subsetting those lists allows tidy storage of slopes and t-values.

```
## Create dummy functions for list
test.mean <- function(x) {
    mean(x)
}
test.median <- function(x) {
    median(x)
}
## Create list with multiple layers
my.list <- list(list(test.mean, test.median), 100)
## Name some elements of my.list
names(my.list) <- c("Element_One", "Element_Two")
names(my.list[[1]]) <- c("Mean", "Median")
## Print the list
my.list
```

```
## $Element_One
## $Element_One$Mean
## function(x) {
##     mean(x)
## }
```

```
##
## $Element_One$Median
## function(x) {
##     median(x)
## }
##
##
## $Element_Two
## [1] 100
```

```
## The subset my.list[[1]][[1]] returns a function that takes


## the mean of an input vector.
my.list[[1]][[1]](c(1, 2, 3))
```

```
## [1] 2
```

3. The dollar sign operator $X\$i$ is the most straightforward subsetting operator.

"$" is most often used to extract variables (columns) from a dataframe by name. It can do the same for any non-atomic array, so we will do an example with the object my.list.

```
## $Mean
## function(x) {
##     mean(x)
```

```
## }
##
## $Median
## function(x) {
##     median(x)
## }

## [1] 2
```

Like the double bracket operator, the dollar sign drops formatting on the extracted object. As such, named functions stored within lists can be immediately called, and atomic data can be immediately operated on. While I never perform a double dollar sign extraction in my program, I do perform extractions from lists using a combination of subsetting methods.

Control Structures

Complicated functions can be built by chaining and/or nesting conditional execution statements. In R, conditional execution arises from the standard if-else statement. The most widely used control structures available in R are the if-else chain, the for while and repeat loops, and the reserved words break and next. According to the R documentation for these constructs, "They function in much the same way as control statements in any Algol-like language." For loops run over a predetermined set of indices supplied to the loop, and are therefore a fundamentally finite process. In my project for loops are most widely implemented for program stability. For loops are also easiest to troubleshoot, because R records the loop index in a dummy variable. While loops

are more dangerous because a set condition is checked before each cycle and there is no guarantee that such a condition will ever flag. That is not to say these are not useful - the ability to iterate an arbitrary number of times is very powerful. The last type of loop is the repeat loop, which loops continuously until an internal break is called. As with the while loop, a user must be cautious to avoid an infinite loop. The advantage of a repeat loop is manual control over when the break condition gets evaluated. To highlight the differences between while and repeat, let's look at an example:

```
x1 <- 0
while (x1 < 0) {
    x1 <- x1 + 1
}
x1
```

```
## [1] 0
```

While loops check the evaluation condition at the beginning of the loop. Because zero is not less than zero, the loop immediately flags and therefore the value of x remains zero.

```
x2 <- 0
repeat {
    if (x2 >= 0) {
        break
    }
    x2 <- x2 + 1
```

```
}
x2
```

```
## [1] 0
```

Repeat loops require their user to place the break condition manually. By placing the condition at the beginning of the loop, we have recreated a while loop. Note that the condition used is inverted with respect to the while loop for usage of the if() construct.

```
x3 <- 0
repeat {
    x3 <- x3 + 1
    if (x3 >= 0) {
        break
    }
}
x3
```

```
## [1] 1
```

Even though the condition used in this loop is identical to the previous loops, the output is different. Placing the loop break at the end of a loop causes R to run one iteration before quitting. Often this difference can be overcome by a change of condition to a while loop, which is often preferable for legibility. Changing the condition $x < 0$ to $x < 1$ in the while loop makes it equivalent to the second repeat loop. One final note - the next statement ends the

current iteration and advances to the next. One application lies in the handling of missing values. In the following example, the is.na() function returns TRUE if the supplied argument is a missing value and FALSE otherwise. The seq_along() method creates a set of integer indices from 1 to the length of the supplied vector. We now seek to build a loop that counts how many TRUE statements exist in a row while ignoring missing values:

```
## Create vector to iterate over
x4 <- c(TRUE, TRUE, NA, FALSE, TRUE)
cntr <- 0
for (i in seq_along(x4)) {
    ## seq_along(x4) generates the vector 1:5 If the ith element
    if (is.na(x4[i])) {

        ## of x4 is missing, skip the iteration
        next
    }
    if (x4[i] == TRUE) {
        ## If the ith element of x4 is TRUE increment

        ## the counter by one
        cntr <- cntr + 1
    } else {
        ## By the process of elimination the current element
```

```
        ## must be FALSE, so we end the loop.
        break
    }
}


cntr
```

```
## [1] 2
```

Functions

In his 2008 book *Software for Data Analysis: Programming with R*, John Chambers said: (18)

> Nearly everything that happens in R results from a function call. Therefore, basic programming centers on creating and refining functions.

It often happens that a programmer wishes to change a couple parameters before reusing some existing code. Functions accomplishes this task efficiently. Such objects can take inputs, no inputs, or have a pre-set but changeable parameter. These arguments are called the "formals" of the function. The "body" consists of the code executed upon a function call using the supplied formal arguments. Functions must exist either in the current working directory or a higher directory in order to be called under R's scoping rules.

Let's say we had 500 logical vectors of length five and wished to find the mean number of first TRUE's before the first FALSE over all such vectors. The code

in the above example loop serves this purpose quite well, but copy/pasting the entire block would be neither tidy nor runtime efficient. We will store the chunk as a function, then apply it across all vectors to find the mean.

```
results <- rep(0, 500)  ##Create a vector to store the number

## of TRUE's before the first FALSE for each vector of length 5.

Trues.Before.False <- function(vector) {
    cntr <- 0
    for (i in seq_along(vector)) {
        ## seq_along(vector) creates indices to iterate over

        ## with length equal to the length of the vector

        if (is.na(vector[i])) {
            ## If the ith element of the vector is missing,

            ## skip the iteration
            next
        }
        if (vector[i] == TRUE) {
            ## If the ith element of the vector is TRUE,

            ## increment the counter by one
```

```
            cntr <- cntr + 1
        } else {
            ## By the process of elimination the current

            ## element must be FALSE, so we end the loop.
            break
        }
    }
    return(cntr)
}

## Apply Trues.Before.False manually across 500

## randomly generated vectors of length 5.
for (j in seq_along(results)) {

    current.vector <- as.logical(rbinom(n = 5, size = 1, prob = 0.8))
    ## Create a logical vector of length 5 with

    ## TRUE probability 0.8.
    results[j] <- Trues.Before.False(current.vector)
    ## Store the number of TRUE's before the first FALSE to the

    ## jth element of results.
```

```
}


mean(results)
```

```
## [1] 2.64
```

```
# Output the mean number of TRUE's before the first FALSE.
```

Instead of Trues.Before.False onto each vector individually, we have the option to use one of R's built in apply functions. These take an input object, an input function, and sometimes additional specifications to achieve the same result in a more efficient manner. Apply functions can make the code mode legible as well, which helps for reproducibility. The method lapply() takes a list and a function as input then returns a list wherein one item is the output of one function application. vapply specifically takes a list or a vector as input, and allows the user to specify what type of object should be returned using the argument FUN.ARGS. sapply() automatically determines what output is best, preferentially choosing vectors and matrices if possible. In my analysis I will use sapply() a couple times, but more often use manual application for conciseness. As an example, we will use sapply() to recreate the mean calculation. We will also put the entire command within another function environment for ease of duplication with a modifiable simulation number set to a default of five hundred.

```
## We need to supply sapply with a list of logical vectors with
```

```r
## length 5. We will use the same for() loop as before

## to accomplish this.

Make.Trues.Before.False.Mean <- function(n.sims = 500) {

    five.numbers <- rep(as.numeric(NA), 5)
    ## Create a dummy variable to ensure list class is numeric.

    ## We initialize this vector using the coercion method

    ## as.numeric() on missing values as a safety measure to

    ## ensure the initialization does not affect the outcome.
    sapply.list <- rep(list(five.numbers), n.sims)
    ## Create a list with n branches to store individual

    ## simulations. The individual branches contain one numeric

    ## vector of length five. For this output we set n.sims=500
    for (j in seq_along(sapply.list)) {
        sapply.list[[j]] <- as.logical(rbinom(n = 5, size = 1, prob = 0.8))
        ## Create a logical vector of length 5 with TRUE
```

```
        ## probability 0.8 and save it to the jth branch

        ## of sapply.list.
    }

    return(mean(sapply(sapply.list, Trues.Before.False)))
    ## Use sapply to determine the number of TRUE's before

    ## the first FALSE for each simulation, then return

    ## the mean of all n.sims simulations.
}

Make.Trues.Before.False.Mean()
```

```
## [1] 2.76
```

Even though the procedure is identical, the output is different. This is because the five hundred vectors of length five get resimulated, and the number of TRUE's before the first FALSE among those five numbers has some variance in accordance with the standard deviation of the sampling distribution. Recall that this is the standard error $se = \frac{\sigma_x}{\sqrt{N}}$. The deviation between instances of Make.Trues.Before.False.Mean() will then decrease monotonically as the number of simulations increases ($N \in \mathbb{N}$). Two histograms using Make.Trues.Before.False.Mean() with different number of simulations are pre-

sented below for visualization of this idea as figures 33 and 34.

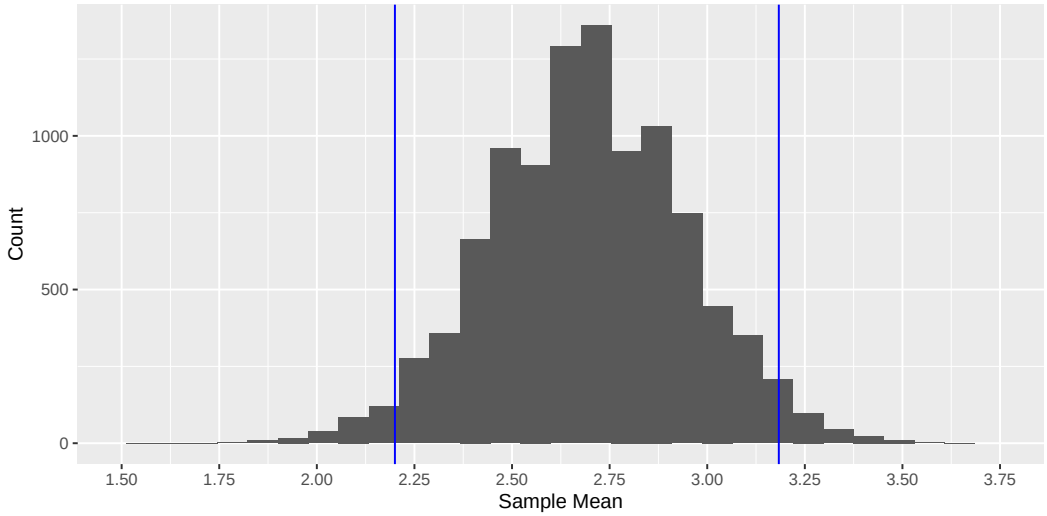

Figure 36: Sampling distribution of the function Make.Trues.Before.False.Mean for sixty vectors of length five, with 95% confidence interval.

Scoping Rules

The function Trues.Before.False() is not defined locally within Make.Trues.Before.False.Mean(). R's scoping rules allow for a function call to access not only the local environment but also any higher environment such as a package all the way up to the global environment. That is, a user can assign a value to an object in a function call that will not influence the representation of a preexisting object with the same name. When R encounters a reference to an object, it begins by looking among those locally defined before proceeding upwards. As a consequence, the following R code returns 100 instead of 10:

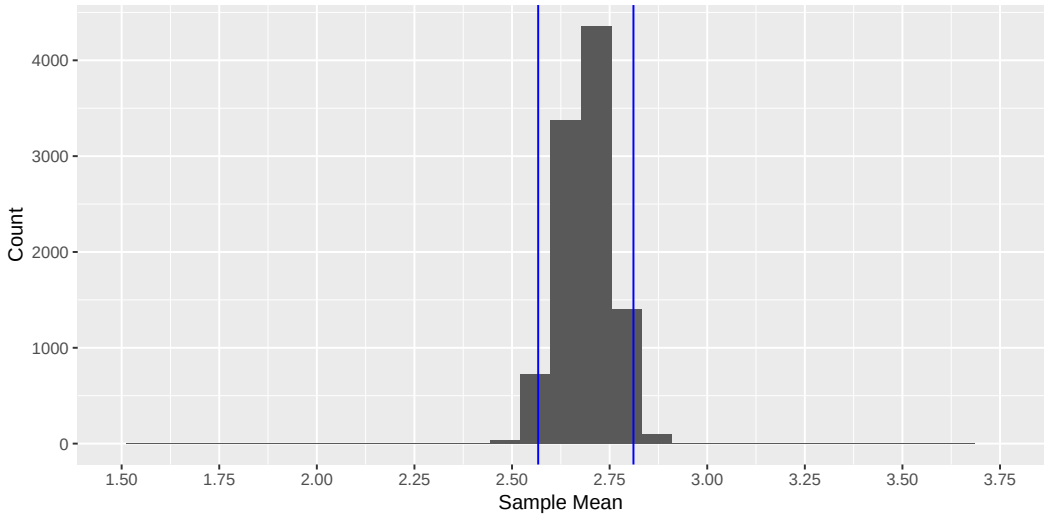

Figure 37: Sampling distribution of Make.Trues.Before.False.Mean for one thousand vectors of length five, with 95% confidence interval.

```
cntr <- 0  ## Create a counter
for (i in 1:10) {
    ## Define the index i to run over the vector 1:10
    for (i in 1:10) {
        ## Define ANOTHER index ALSO CALLED i that runs over

        ## another vector 1:10. R's scoping rules check the

        ## local value i's local value before incrementing at

        ## the beginning of the for() loop.
        cntr <- cntr + 1
```

```
    }
}
cntr
```

```
## [1] 100
```

In RMarkdown objects that are created in code chunks are saved to the global environment. We can therefore call functions that are not locally defined, which is immensely useful in functional programming.

Simulation

The modus operandi of simulation in R is to model how multiple random samples interact over a large number of iterations. R can generate random values from a slew of of common probability distributions including the normal, Poisson, uniform, gamma, and as has already been demonstrated the binomial. Random values can either serve many purposes, such as serving as an a-priori probability distribution from which to create a model, or as a theoretical sample against which to check an empirical distribution. A user can also calculate the PDF, CDF, and quantiles of these various distributions after supplying the necessary formal arguments. I will use the t distribution's quantile function qt() when creating confidence intervals, and the normal probability plotter qqnorm() when checking my empirical quantiles for normality. For probability plotting of other distributions see the R documentation for the function qqPlot() in EnvStats.

R uses the psuedo-random number generator (PNRG) algorithm called "The

Mersenne Twister" created by Drs. Matsumoto and Nishimura in 1998.(19) The Mersenne Twister begins generating numbers from a given starting point. Furthermore, the sequence of numbers generated is entirely deterministic. Actions taken in an R script advances the "seed" of the Twister by one. By controlling the initial seed of the Mersenne Twister, the random numbers generated in any set of commands can be exactly recreated. The set.seed() method lets the user specify which seed R should use next, and therefore is an excellent tool for reproducibility of experiments. In my analysis I set the initial seed to 1000, and then let the Mersenne Twister twist.

As the name suggests, R's sample() function allows its user to create random samples from a supplied atomic vector. The four main arguments of sample() that must be considered are the initial population, the sample size, the replacement condition, and whether certain elements should have weighted probabilities of being drawn. Once parameters are selected, we are ready to simulate. In the data analysis we will use simulation in two contexts - the first is the creation of bootstrap sampling distributions, and the second is to model the approximate number of possible unique values of cumulative GPA as a function of graded term (under some assumptions). I leave these as examples of the functionality of sample(). There are two bootstrapper functions in the analysis - one for testing the significance level of a faculty's mean ΔC and one for ΔW - and one cumulative GPA simulator. By name they are make.cum.bootstrap.means(), make.withdraw.bootstrap.props(), and Possible.Cum.GPAs().

Appendix B: The Tidyverse Arsenal

While the R Programming tools I learned from Coursera, probability, and mathematical statistics are sufficient for the creation of the necessary functions to analyse Professor Eckstrom's dataset, many smart people (Hadley Wickham and Yihui Xie primary among them) have built specialized tools in R for the tidying, manipulation, and presentation of data. The tidyverse is a collection of R packages including ggplot2, dplyr, readr, tibble, and forcats among others. Dr. Nordmoe's data science course MAT-295 is an introduction to the tidyverse approach of data analysis in R, so I will only give a brief summary of the methods I call. I highly recommend reading the following sections in conjunction with the linked tidyverse cheatsheet, as these guides contain information about the syntax and usage of methods found in one package.

Piping

Having said that, there is unfortunately no cheatsheet for the tidyverse package magritrr. In lieu I will try to clearly indicate the pipe operators syntax and usage. Complicated functions become quite illegible very quickly. For example, the proper base R syntax for the composition of a function A $f_A(x, A_1, ...)$ with a function B $f_B(x, B_1, ...)$ called on an object x is $f_A(f_B(B(x, B_1, ...)), A_1, ...)$. When composing even as few as four or five functions it quickly becomes difficult to discern which formals are being supplied to which functions. When the tidyverse is loaded using the library() function, the pipe operator `%>%` is exported from the R package magritrr. Using the pipe changes the func-

tion composition into a function ordering; $x \ \% > \% \ f_A(A_1, ...) \ \% > \% \ f_B(B_1, ...)$ clearly indicates that the object x is first operated on by the function f_A with formal arguments $\{A_1, ...\}$ before being sent into f_B with formals $\{B_1, ...\}$. Many tidyverse methods are built for usage with the pipe operator, allowing for legible data manipulations and transforming R from a functional language into a an object-oriented language.

dplyr

The tidyverse package dplyr provides many streamlined, composable functions for the manipulation of data sets. The filter() function removes all rows of a data set that do not obey a supplied logical condition. Similarly the slice() method can be used to choose rows by index. mutate() and transmute() create new columns or alter existing ones, and rename() is a streamlined wrapper for mutate() that only changes the variable name. Conversely, the select() method can either remove variables entirely or just reorder columns. A user can use the arrange() function to reorder the rows of a data set from highest to lowest or vice versa based on a supplied set of variables. The verb summarise() creates a new data frame with columns generated procedurally from its formal arguments. Each of these methods is referred to as a dplyr verb, and works in conjunction with the group_by() function.

In my data analysis I used group_by() and its reciprocal ungroup() cumulatively more than any other single dplyr verb. The group_by() method converts a dataframe into a grouped dataframe. By itself this does nothing, but when combined with the other dplyr verbs a user gains enormous analytical

power. Using Professor Eckstrom's dataset which I have named Gen_Data as an example;

Table 18: All rows returned by slice(1) acting on Gen_Data

Student Random ID	Course Code	Grade	Faculty Random ID	Semester	Total.Term	GPA Assigned
31	MAT-1540	F	643249	2012/SU	6	0

Table 19: First six rows returned by slice(1) acting on Gen_Data grouped by Student Random ID

Student Random ID	Course Code	Grade	Faculty Random ID	Semester	Total.Term	GPA Assigned
31	MAT-1540	F	643249	2012/SU	6	0.0
48	MAT-1500	W	105920	2012/SU	6	NA
50	MAT-1100	A	723344	2011/WI	2	4.0
67	MAT-1100	W	207753	2011/WI	2	NA

Student Random ID	Course Code	Grade	Faculty Random ID	Semester	Total.Term	GPA Assigned
72	MAT-1100	F	144199	2016/FA	19	0.0
75	MAT-1540	B-	514002	2014/FA	13	2.7

If possible, dplyr verbs are applied over each group. Tables 16 and and 17 show how grouping allows for greater control of data wrangling. When slice(1) is called on Gen_Data without any grouping, only the first row of the dataset is returned. When slice(1) is called on Gen_Data after the data have been grouped by Student Random ID, the first observation for each student is returned. The row(s) returned therefore depends on the current ordering of Gen_Data, which is controlled in dplyr through the arrange() method. group_by() is also commonly used with the summarise() verb to create custom variables that vary based on group. X %>% group_by() %>% count() is a convenient wrapper for summarise that is generally equivalent to X %>% group_by() %>% summarise(n = n()).

There are two general types of join commands available in dplyr. The mutating joins allow for two datasets with at least one common variable called a key to be combined combinatorically into one new dataset. The filtering joins remove rows from a dataset based on the presence or absence of rows in a second by key. These join commands are crucial to the execution of complex functions

onto datasets where mutate is insufficient or inefficient.

ggplot2

With the notable exception of normal probability plots, every graphic I create in this report is created by the tidyverse package ggplot2. As in dplyr, the methods of ggplot2 are built to work together. Each graphic made in ggplot requires a dataframe (or tibble) and a geom. Geoms are the various graphic methods available; geom_histogram() creates a histogram and geom_point() creates a scatterplot. The nomenclature for most geoms are self-identifying. Ggplot2 uses "aesthetics" to define which variables of the supplied data get mapped to which quantities graphically. Some examples of aesthetics are color, shape, point size, and the x and y coordinates. The + operator is a general symbol which tells R to look for more commands before finishing evaluating. ggplot graphics are constructed using multiple commands connected by the + operator.

Other packages

Other tidyverse functions I used include tibble() from Tibble as an alternative to data.frame(), str_sub() from stringr to create department names, and factor() in conjunction with fct_reorder() from Forcats for the rearranging of categorical variables in bar charts. Finally, the impact of Yihui Xie et al.'s RMarkdown and knitr cannot be overstated. This SIP would not have compiled - and therefore would not have been possible - without them.

Appendix C: Coursera Certificates, Grades, and Functions

Figure 38: Certificate of completion for Coursera module "The Data Scientist's Toolbox"

Appendix D: Course Code Lookup Table

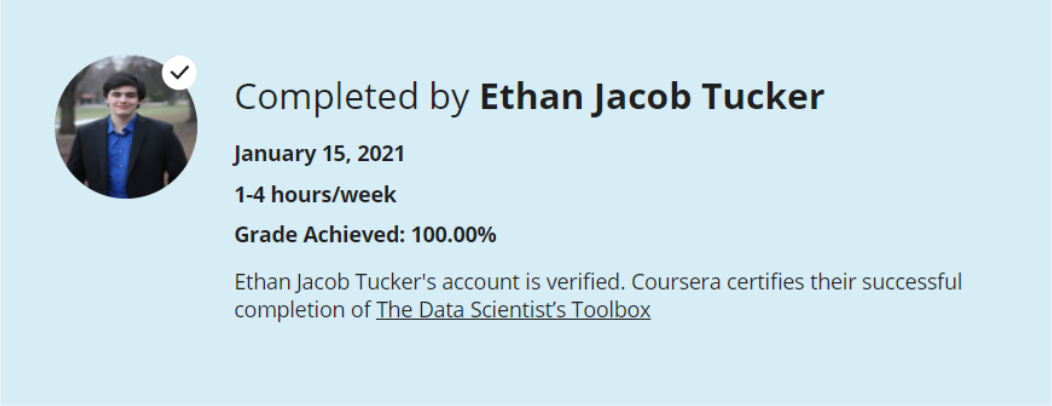

Figure 39: Grade recieved in "The Data Scientist's Toolbox"

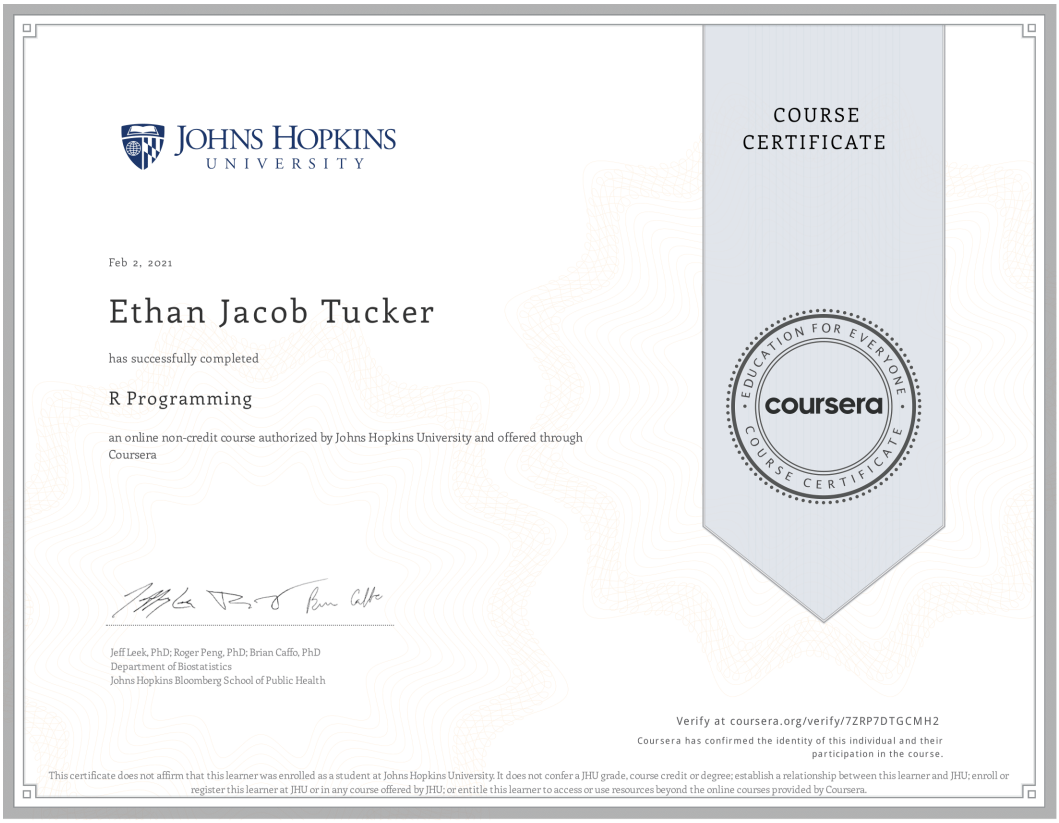

Figure 40: Certificate of completion for Coursera module "R Programming"

Figure 41: Grade recieved in "R Programming"

Table 20: Course lookup table containing course code, the OCC credits the course is worth, and the course name.

Course Code	Course Credits	Course Names
CHE-0950	4	Preparation for Chemistry
CHE-1000	4	Introductory Chemistry
CHE-1320	4	Survey of Organic and Biochemistry
CHE-1510	4	General Chemistry I
CHE-1520	4	General Chemistry II
CHE-2610	4	Organic Chemistry I
CHE-2620	4	Organic Chemistry II
CHE-2650	3	Organic Chemistry Lab
MAT-1045	4	Fundamentals of Arithmetic
MAT-1050	4	Preparation for Algebra

Course Code	Course Credits	Course Names
MAT-1070	3	Buisness Mathematics
MAT-1100	4	Elementary Algebra
MAT-1125	4	Math Literacy
MAT-1140	3	Plane Geometry
MAT-1150	4	Intermediate Algebra
MAT-1500	4	Finite Mathematics
MAT-1525	4	Quantitative Reasoning
MAT-1540	4	College Algebra
MAT-1560	3	Trigenometry
MAT-1580	4	Statistics
MAT-1600	4	Applied Calculus
MAT-1630	5	Precalculus
MAT-1730	4	Calculus I

Appendix E: GitHub Repository Info and Contact Information

The GitHub repository containing this project can be found here. The output of this document is found in the SIP_Report.Rmd file. All data necessary are included in the repository. It takes a long time to compile due to all the simulations that need to run. R must have the tidyverse installed to compile. For any files and/or questions, my email is firstrider55@gmail.com.

References

[1] Strategies to increase course evaluation response rates. (n.d.). Retrieved April 26, 2021, from https://www.marquette.edu/institutional-research-analysis/moces/response-rates.php#: :text=Instructors%20who%20provide%20students%20time,evaluations%20are%20important%20to%20you.

[2] Thomas Timmerman (2008) On the Validity of RateMyProfessors.com, Journal of Education for Business, 84:1, 55-61, DOI: 10.3200/JOEB.84.1.55-61

[3] Michael E. Sonntag, Jonathan F. Bassett & Timothy Snyder (2009) An empirical test of the validity of student evaluations of teaching made on RateMyProfessors.com, Assessment & Evaluation in Higher Education, 34:5, 499-504, DOI: 10.1080/02602930802079463

[4] Oakland Community College. (n.d.). Retrieved April 26, 2021, from https://datausa.io/profile/university/oakland-community-college

[5] Blog, N. (2019, April 4). Research Center Snapshot Report Showcases Yearly Success and Progress Rates for Fall 2012 Freshman Class. Retrieved April 25, 2021, from https://www.studentclearinghouse.org/nscblog/research-center-snapshot-report-showcases-yearly-success-and-progress-rates-for-fall-2012-freshman-class/

[6] R For Data Science: Import, Tidy, Transform, Visualize, and Model Data. (2017). Sebastopol, CA: O'Reilly.

[7] LaMorte, W. W. (2016, July 24). The role of probability. Retrieved April 26, 2021, from https://sphweb.bumc.bu.edu/otlt/mph-modules/bs/bs704-probability/BS704-Probability12.html

[8] Wagaman, A. S., & Dobrow, R. P. (2014). Probability: With applications and R. Hoboken, NJ: Wiley.

[9] Ganti, A. (2021, March 31). What is the Central Limit Theorem (CLT)? Retrieved April 26, 2021, from https://www.investopedia.com/terms/c/central-limit-theorem.asp#: :text=Sample%20sizes%20equal%20to%20or,characteristics%20of%20a%20population%20accurately

[10] Chihara, L. M., & Hesterberg, T. C. (2019). Mathematical statistics with resampling and R. Hoboken, NJ: John Wiley & Sons.

[11] Research Essentials for Massage in the Healthcare Setting, Glenn M. Hymel, in Clinical Massage in the Healthcare Setting, 2008

[12] Zhu, W. (2021, April 14). Confidence Interval. Lecture presented at Lecture on Mathematical Statistics in State Univerity of New York at Stony Brook, Stony Brook.

[13] Michael R. Chernick (https://stats.stackexchange.com/users/11032/michael-r-chernick), Determining sample size necessary for bootstrap method / Proposed Method, URL (version: 2018-03-23): https://stats.stackexchange.com/q/33302

[14] A5C1D2H2I1M1N2O1R2T1 (https://stackoverflow.com/questions/22746508/r-

simplifying-code-to-convert-letter-grades-to-numeric-grades), R: Simplifying code to convert letter grades to numeric grades, URL (version: 2014-03-30): stackoverflow.com/questions/22746508

[15] Bates, Douglas, Martin Mächler, Ben Bolker, & Steve Walker. "Fitting Linear Mixed-Effects Models Using lme4." Journal of Statistical Software [Online], 67.1 (2015): 1 - 48. Web. 26 Apr. 2021

[16] Clark, M. (2020, November 30). Mixed models with r. Retrieved April 26, 2021, from https://m-clark.github.io/mixed-models-with-R/

[17] Peng, R. D. (2016). R Programming for data science. Victoria, British Columbia, Canada: Leanpub.

[18] Chambers, John M. (2008). Software for data analysis programming with R. Berlin: Springer. ISBN 978-0-387-75935-7.

[19] Makoto Matsumoto and Takuji Nishimura. 1998. Mersenne twister: a 623-dimensionally equidistributed uniform pseudo-random number generator. ACM Trans. Model. Comput. Simul. 8, 1 (Jan. 1998), 3–30. DOI:https://doi.org/10.1145/272991.272995