3-2021

The Time-to-Graduation Problem (Survival Analysis for Education Outcomes)

Don R. Hush
University of New Mexico - Main Campus, dhush@unm.edu

Tushar Ojha
University of New Mexico - Main Campus, tushar3309@unm.edu

Wisam Al-Doroubi
University of New Mexico - Main Campus, wisamquais@unm.edu

The Time-to-Graduation Problem

(Survival Analysis for Education Outcomes)

Don Hush, Tushar Ojha, Wisam Al-Doroubi
Institute of Design and Innovation
University of New Mexico, Albuquerque, NM 87106
idi.unm.edu

March 12, 2001

Contents

Abstract

This report provides a comprehensive treatment of the *time-to-graduation* (TTG) problem. The goal of this problem is to make predictions about the time it takes for a student to graduate from a particular institution. We develop a complete mathematical description of the TTG problem which includes data synthesis models that reveal details of the problem that are often hidden or overlooked in the literature. This report explores two vastly different approaches to this problem, including detailed algorithmic descriptions and practical examples where specific solution methods are applied to both synthetic data and real UNM student data. The first approach is based on *survival analysis* methods which are the dominant approach in the educational literature. While these methods might appear to be well matched to the TTG problem, their blind application often overlooks important aspects of this problem. One goal of this report is to identify some of these aspects and describe modifications to accommodate them. The second approach is based on *semi-supervised learning*, a type of machine learning that builds models using both *labeled* and *unlabeled* data. In particular we introduce a specially designed *semi-supervised* likelihood function tailored to the TTG problem, and then apply the maximum likelihood (ML) method to build the model. We derive an Expectation-Maximization (EM) algorithm to carry out this optimization. Finally, the application of these methods to UNM data reveals numerous important characteristics of the *time-to-graduation* for UNM students.

Keywords: graduation and dropout times, survival analysis, semi-supervised learning

1 Introduction

The goal of the *time-to-graduation* (TTG) problem is to make predictions about the time it takes for a student to graduate from a particular institution. This report explores two vastly different approaches to this problem. The first approach is based on *survival analysis* methods. *Survival analysis* is a type of *longitudinal* data analysis where the goal is to make predictions about the time at which a particular event is likely to occur. A classic example is a clinical trial where the event of interest is *death*, and the goal is to make predictions about the survival time (i.e. time-to-death) for subjects under different procedures/treatments/drugs (hence the name *survival* analysis). At first glance it would appear that survival analysis is well matched to the TTG problem. Indeed, the literature contains many examples where survival analysis has been applied to graduation and retention problems in education (e.g. see Appendix A). However, blind application of survival analysis methods often overlooks important aspects of the TTG problem. One goal of this report is to identify some of these aspects and describe modifications to accommodate them. Some of these modifications were introduced elsewhere (e.g. see [23]), while others are original contributions. A brief literature review of survival analysis and its application to educational data analysis can be found in Appendix A.

The second approach that we explore is based on *semi-supervised learning*, which is a machine learning method that exploits two types of data samples that are generally referred to as *labeled* and *unlabeled*. There are numerous approaches to *semi-supervised learning*. Our approach here is to build a probability model using the maximum-likelihood (ML) method with a specially designed *semi-supervised* likelihood function that is tailored to the unique type of *"unlabeled"* data found in the TTG problem. To carry out the likelihood optimization we derive an Expectation-Maximization (EM) algorithm.

This report is organized as follows. We develop the full-blown TTG problem one step at a time, starting with the simplest non-trivial version and adding real world attributes as we go. With each version we provide the following.

1. A complete mathematical description of the problem.
2. A synthesis method which provides a way to generate synthetic data, and sometimes reveals details of the problem that are often hidden or overlooked in the literature.
3. A simple *survival analysis* solution method that can be implemented with minimal effort.
4. An example (or examples) where the *survival analysis* solution method is applied to real and/or synthetic data.

In the end there are a total of 6 versions of the TTG problem, where each version is slightly more complex than the previous. Once we have developed the full-blown TTG problem and a corresponding *survival analysis* solution, we develop the *semi-supervised* approach. But before we start all of this development we first discuss the ways *time* will be treated in this report and provide two different definitions of event time.

2 Event Time Definitions

All *times* in this report are specified in units of *semesters*. In addition we assume two semesters per year. We accommodate summer semesters by grouping them with the previous spring semester,

unless the summer semester corresponds to the student's very first semester in which case it is grouped with the subsequent fall semester.

This report makes reference to two types of times: *absolute* and *relative*. Absolute times correspond to calendar times. For example Fall 2015 and Spring 2020 are examples of absolute times. Absolute times may be specified in their natural form, e.g. Fall 2015 and Spring 2020, or in numerical form with an arbitrary starting point. For example if Fall 2010 is declared to be absolute semester 1 then Fall 2015 would be absolute semester 9. Relative times are numerical values computed relative to a student's starting semester. For example, suppose Fall 2015 is the starting semester for student A and Fall 2016 is the starting semester for student B. Then Spring 2017 is relative semester number 4 for student A and 2 for student B. The *event* times that we analyze in this report are all relative times, and because of their ubiquity we refer to them simply as "times" without the qualifier "relative". On the other hand, absolute times will be referred to either in their natural form, e.g. Fall 2015, or by using the qualifier "absolute".

We require event times to correspond to a semester in which the student was enrolled, i.e. we deem it impossible to experience an event (grad or drop) at the end of a semester where the student was not enrolled. With this there are two obvious ways to quantify the event time:

1. **NSS** = total number of semesters from the absolute start semester to the absolute event semester, and

2. **NSE** = number of semesters enrolled from the absolute start semester to the absolute event semester.

The relation $NSE \leq NSS$ holds because students may not enroll in every semester. There are several factors to consider when choosing one of these definitions [23].

1. If we are interested in total cost of education, university planning for financial aid, support services, teaching loads, housing demand, etc. then NSE is more appropriate.

2. If we want to examine the *wall clock* time to a degree (e.g. to project flows of new students into the job market) then NSS is more appropriate.

3. The NSS option produces event time probability estimates that are consistent with the standard definition of graduation and retention rates.

4. If we want to predict future enrollments using the method described in Section 9 then the event time must be NSS.

5. From a technical perspective we will see that NSS leads to a simpler data synthesis model.

Much of the development in this report is the same regardless of which definition we choose, but when the development differs due to the event time definition we clearly highlight the differences.

3 TTG Version 1

It turns out that if we wish to make accurate predictions about the time-to-graduate then we must include dropout events in our analysis. Furthermore, dropout statistics are just as important as graduation statistics when it comes to monitoring student progress. Thus, in the TTG problem our goal is to make predictions about both the event type, i.e. *grad* or *drop*, and the time it takes to experience the event. Eventually we will want to know how various student attributes, such as

gender, ethnicity, High School GPA, and financial aid influence these predictions, i.e. we will want to add *covariates* to our model. But for now we omit covariates and seek only to make predictions about the event type and event time for the general student population. To this end we model the (event time, event type) tuple as a random variable where

- (T_e, B) is the (event time, event type) random variable, and (t_e, b) denotes a specific realization (i.e. a sample) of the random variable,
- the event time $t_e \in \{1, 2, ..., \mathsf{T}_e\} = \mathcal{T}$ is a positive integer that represents the semester number at which an event occurs,
- the event type $b \in \{\mathsf{G}, \mathsf{D}\} = \mathcal{B}$ is a categorical variable where $b = \mathsf{G}$ represents the *grad* event, $b = \mathsf{D}$ represents the *drop* event, and
- $P_{T_e,B}$ is the probability distribution on $\mathcal{T} \times \mathcal{B}$ that characterizes the random variable (T_e, B).

Note that $P_{T_e,B}$ is a *discrete* probability distribution that takes only $2\mathsf{T}_e$ distinct values. The joint distribution can be decomposed as

$$P_{T_e,B} = P_{T_e|B} P_B$$

We will use the short-hand notation

$$P_{T_e|b} = P_{T_e|B=b}$$

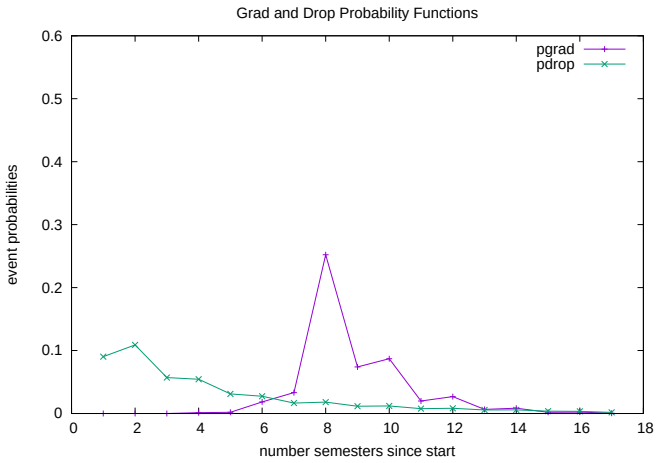
for the distribution given a specific event type b. A more detailed description of the notational conventions used for probability functions in this report can be found in Appendix B.

Next we define the specific predictions we want to make. Numerous possibilities are considered in the list below.

1. **Classification:** One goal might be to predict the most likely (event time, event type) value. The discrete nature of $P_{T_e,B}$ means that this can be cast as a standard *multi-class classification* problem where the number of classes is $2T_e$. However, this prediction has limited utility, since the most likely graduation and drop semesters for the general population are known to be 8 and 2 (or 10 and 2), and this knowledge provides limited insight into the TTG process. Even when we add covariates to allow different (event time, event type) predictions for students with different characteristics, knowing the most likely (event time, event type) often has limited utility. This is partly because there are often several (event time, event type) values with large (and nearly equal) probabilities, so that the most likely (event time, event type) value is often a poor prediction, i.e. it is like knowing the most likely *class* in a multi-class problem that has a very high classification error rate so that predicting the most likely *class* often leads to an incorrect prediction. Knowing the (event time, event type) *probabilities* themselves, instead of just knowing which is largest, provides much more useful information to the end user.

2. **Probability:** In this case we want to estimate the (event time, event type) probabilities as mentioned above, i.e. we want to estimate the probability function $P_{T_e,B}$. This means estimating the probability of both events, grad and drop, at each semester $t \in \mathcal{T}$. This function provides complete statistical knowledge of the random variable (T_e, B), and provides the end user with a useful characterization of student behavior. An example of $P_{T_e,B}$ for UNM FTFT fall cohorts 2013-2017 is shown below[1].

[1]Note that even though these are discrete-valued probability functions the plots in this report connect adjacent probability values with a line to help the reader visualize the probability value trends.



This plot confirms that the most likely graduation semester is 8 and the most likely drop semester is 2. But it also shows other semesters where these events are likely, and more generally provides a complete assessment of the grad and drop probabilities.

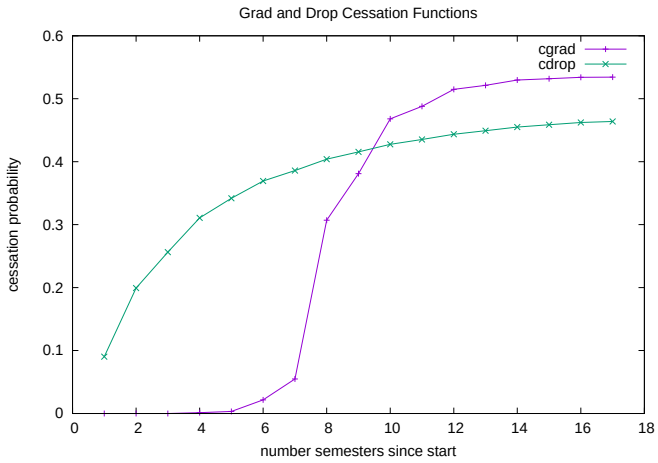
3. **Cessation:** In this case we want to estimate the probability that a student will experience one of the events (grad or drop) on or before the t-th semester, i.e. we want to estimate the cumulative probability function

$$C(t, b) = \sum_{t'=1}^{t} P_{T_e,B}(t', b) \tag{1}$$

This is sometimes referred to as the *cumulative influence function*, e.g. see [23], but we call it the *cessation* function to emphasize its complementary role to the *survival* function described next. For convenience we define the two cumulative distribution functions

$$C_{grad}(t) = C(t, \mathsf{G}) = \sum_{t'=1}^{t} P_{T_e,B}(t', \mathsf{G}) \qquad C_{drop}(t) = C(t, \mathsf{D}) = \sum_{t'=1}^{t} P_{T_e,B}(t', \mathsf{D})$$

$C_{grad}(t)$ is often called the t-th *semester graduation rate*, and $C_{grad}(8)$ and $C_{grad}(12)$ (i.e. the 4 and 6 year grad rates) are two of the most common measures of performance for an education institution. Furthermore, $1 - C_{drop}(t)$ is often called the t-th *semester retention rate*, and $1 - C_{drop}(2)$ (i.e. 1 year retention) is also a common measure of performance for an education institution. An example of C_{grad} and C_{drop} for UNM FTFT fall cohorts 2013-2017 is shown below



From this plot we see that the 4, 5, and 6 year graduation rates are 32%, 47%, and 51% respectively, and the second semester retention rate is 80% (i.e. $100(1.0-0.2)$). It also shows that in the end, 53.5% of these students will graduate (and 46.5% will drop).

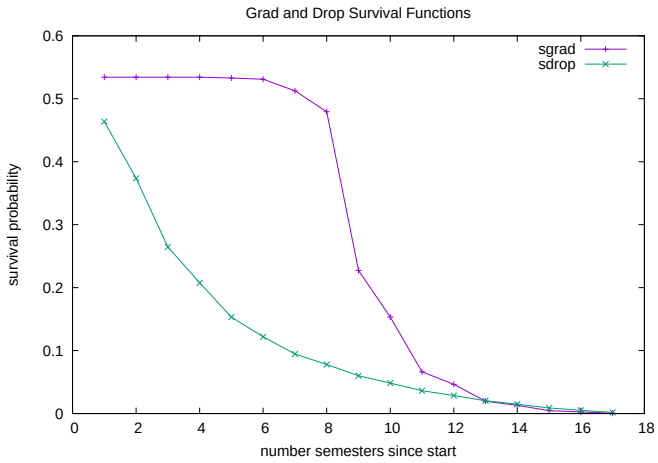
4. **Survival:** In this case we want to estimate the probability that the student will experience one of the events (grad or drop) on or after the t-th semester, i.e. we want to estimate the complementary cumulative distribution function

$$S(t,b) = \sum_{t'=t}^{\mathsf{T}_e} P_{T_e,B}(t',b) \tag{2}$$

Once again, for convenience we define the two survival functions

$$S_{grad}(t) = \sum_{t'=t}^{\mathsf{T}_e} P_{T_e,B}(t',\mathsf{G}) \qquad S_{drop}(t) = \sum_{t'=t}^{\mathsf{T}_e} P_{T_e,B}(t',\mathsf{D})$$

The survival function $S(t) = S_{grad}(t) + S_{drop}(t)$ gives the probability of not graduating or dropping until after semester t, i.e. the probability of "surviving" to semester t. This function can be used to make predictions about future enrollments as described in Section 9. An example of S_{grad} and S_{drop} for UNM FTFT fall cohorts 2013-2017 is shown below



This plot tells us that at the completion of year 5, 15% of the students are still active and will eventually graduate and 5% of the students are still active and will eventually drop.

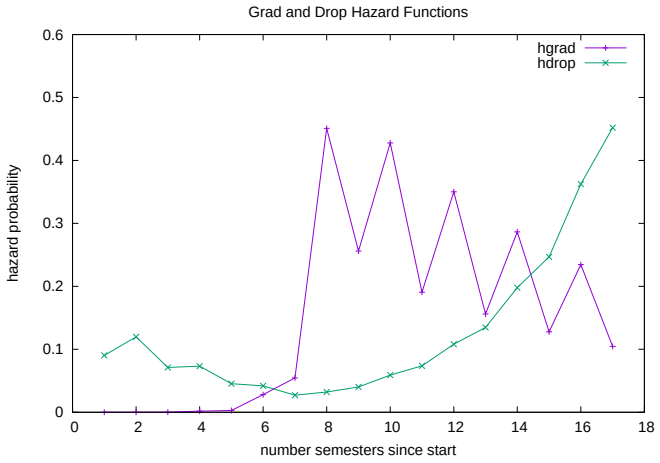
5. **Hazard:** In this case we want to estimate the probability that a student will graduate or drop at the t-th semester given that they have not yet graduated or dropped, i.e. we want to estimate the so-called hazard function

$$h(t,b) = \frac{P_{T_e,B}(t,b)}{\sum_{t'=t}^{\mathsf{T}_e} \sum_{b \in \mathcal{B}} P_{T_e,B}(t',b)} \tag{3}$$

Once again, for convenience we define the two hazard functions

$$h_{grad}(t) = \frac{P_{T_e,B}(t,\mathsf{G})}{\sum_{t'=t}^{\mathsf{T}_e} (P_{T_e,B}(t',\mathsf{G}) + P_{T_e,B}(t',\mathsf{D}))} \qquad h_{drop}(t) = \frac{P_{T_e,B}(t,\mathsf{D})}{\sum_{t'=t}^{\mathsf{T}_e} (P_{T_e,B}(t',\mathsf{G}) + P_{T_e,B}(t',\mathsf{D}))}$$

These functions play a dominant role in survival analysis which will be described later in Section 4 after we introduce the notion of *censoring*. An example of h_{grad} and h_{drop} for UNM FTFT fall cohorts 2013-2017 is shown below.



These functions tell us that, for students who make it to semester 8, the probability of graduating at semester 8 is 45% and the probability of dropping at semester 8 is 3%. The sawtooth pattern of h_{grad} tells us that currently active students are much more likely to graduate during an even semester than an odd semester. The rising trend in h_{drop} in later semesters tells us that currently active students are more likely to drop the longer they remain in school.

Note that the cessation, survival, and hazard functions are all defined in terms of the probability function, so that once we have $P_{T_e,B}$ the other functions can be computed directly. It turns out that the reverse is also true, i.e. the probability function can be derived from each of the other functions. This is trivial in the case of the cessation and survival functions, and in Section 4.3 we show how to derive the probability function from the hazard function. Consequently, if any one of these functions is known (or estimated) then the others can be determined (or estimated) using a straightforward (closed form) computation.

All of the functions above have some utility for the end user, except possibly classification. However, the cessation function may provide the type of information most often sought by the end user. The formal problem statement for TTG Version 1 is as follows.

TTG-V1: *Given a collection* $D_N = ((t_{e_1}, b_1), (t_{e_2}, b_2), ..., (t_{e_N}, b_N))$ *of* N *iid samples from* $P_{T_e,B}$ *that represent event times and event types for students from a particular institution, estimate the prediction functions (hazard, cessation, survival, and probability) for the institution.*

Remark 1. It is important to emphasize that in this report our goal is to produce one or more of the *prediction* functions: probability, cessation, survival, and hazard. We call these *prediction* functions because we plan to use them to make predictions about future students, or about the future of current students that have not yet finished. Furthermore, this will continue to be our goal when we introduce covariates to model student attributes, i.e. we will seek to make predictions that depend on student attributes. In contrast, traditional survival analysis is often more concerned with the problem of determining which student attributes have the most influence on the time-to-graduation (or which attributes are superfluous). This type of analysis is often referred to as factor analysis, analysis of covariance, feature selection, or sensitivity analysis. In this type of analysis the *parameterization* of the prediction functions is often more important than the prediction functions themselves, i.e. the analysis is concerned more with the parameter estimates than the function estimates. For example, this is overtly clear in the traditional survival analysis development of hazard functions for *time-varying* covariates where the hazard functions themselves are not time-varying and therefore cannot be used to make future predictions[2], but the hazard function coefficients are used to assess the influence of student attributes.

3.1 A Simple Nonparametric Estimator for TTG-V1

As stated in the TTG-V1 problem above, let $D_N = ((t_{e_1}, b_1), (t_{e_2}, b_2), ..., (t_{e_N}, b_N))$ be a collection of N iid samples from $P_{T_e,B}$. Then a simple non-parametric estimate of $P_{T_e,B}$ can be formed as follows

$$\hat{P}_{T_e,B}(t, b) = \frac{n(t, b)}{N}, \qquad \forall (t, b) \in \mathcal{T} \times \mathcal{B} \tag{4}$$

where

$$n(t, b) = \text{ number of samples from } D_N \text{ where } (t_{e_i}, b_i) = (t, b)$$

This estimate is unbiased, consistent, and reliably accurate when N is sufficiently large. Estimates of the other prediction functions can be obtained by substituting $\hat{P}_{T_e,B}$ for $P_{T_e,B}$ into (1), (2), and (3).

Example 1. Consider the UNM FTFT cohort that started in the Fall of 2006. The total number of students in this cohort is 3026. A record of their grad/drop status is kept in a UNM data table for 10 years (20 semesters) and is shown below.

ngrad:	0	0	1	0	2	12	36	330	338	399	108	156	59	55	34	34	28	30	7	10
ndrop:	257	291	140	146	73	71	42	46	39	37	34	33	17	33	17	19	13	15	10	54
semester:	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20

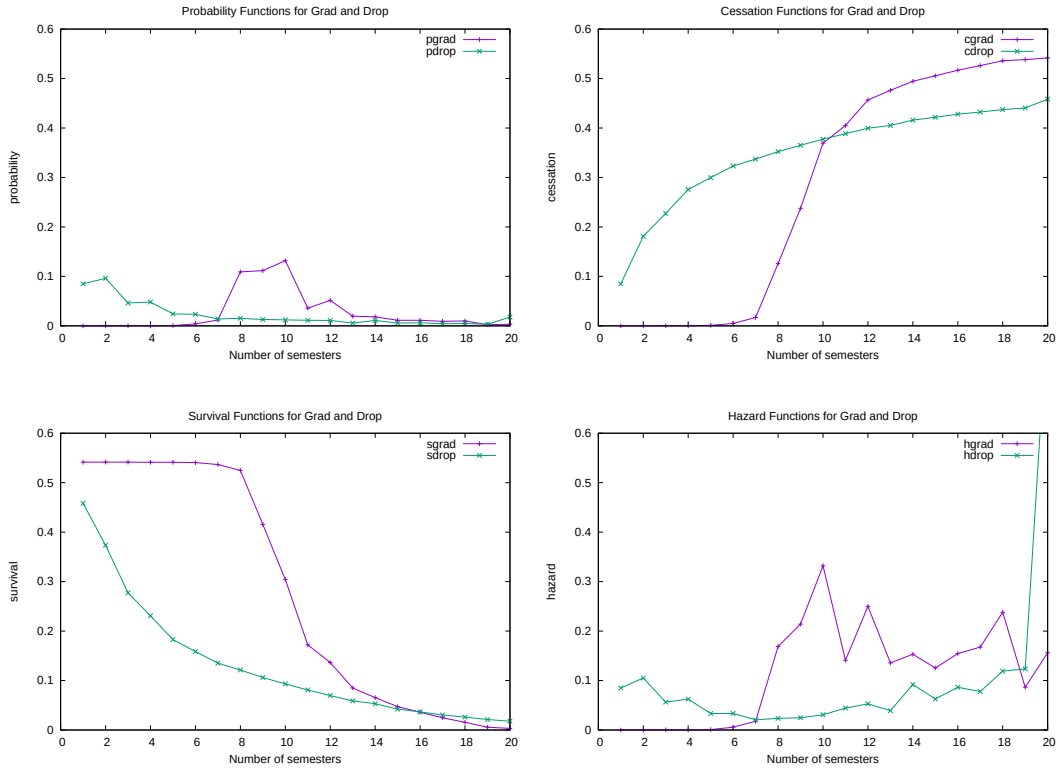
Note that the *ndrop* value in the last semester is suspiciously large, probably because some of these students have not yet dropped or graduated. That is, some of them remain active after 20 semesters, but the UNM table fails to keep track of their enrollment beyond 20 semesters. Future sections of this report provide mechanisms to correct for this phenomenon, but for this example we

[2]That is, a static hazard function cannot be used to predict the time-varying behavior of an incoming student. Instead a time-varying function, such as a Markov model, is required to make such predictions.

accept the data at face value. The probability estimate in (4) can be computed directly from the data table. For example the estimate of the probability of graduating at semester 8 is

$$\hat{P}_{T_e,B}(8,\mathsf{G}) = \frac{330}{3026} \approx .11$$

By first estimating the complete probability function, and then substituting these estimates into (1)-(3) we obtain the estimates of probability, cessation, survival and hazard functions shown below.



Remark 2. In addition to producing estimates of the probability, cessation, survival, and hazard functions it is important to quantify the accuracy of these estimates. For example the accuracy of the estimate in (4) might be obtained as follows. Equation (4) produces $2\mathsf{T}_e$ estimates, one for each distinct value of (t, b). These estimates $n(t, b)/N$ are the *sample means* from N trials of a multinomial random variable with $2\mathsf{T}_e$ distinct values. Thus the true means, i.e. the true values of $P_{T_e,B}(t, b)$, are contained within *intervals* $[L_\alpha(t, b), U_\alpha(t, b)]$ with some confidence $1 - \alpha$, where $L_\alpha(t, b)$ and $U_\alpha(t, b)$ are derived using the standard confidence interval analysis for multinomial distributions, e.g. see [18]. The accuracy of the estimates $\hat{P}_{T_e,B}$ are then determined by the size of these confidence intervals $[L_\alpha(t, b), U_\alpha(t, b)]$. To determine the accuracy of the other function estimates we first determine the distribution of the random variables obtained by passing the multinomial random variables through the computations used to determined the other function estimates. Then we apply a standard confidence interval analysis based on these distributions. Many of the function estimates developed later in this report also take on a simple *sample mean* form like (4), and so their accuracy analysis will follow a very similar path. The details of this analysis is beyond the scope of this report.

4 TTG Version 2 (Censoring)

One of the unique and distinguishing characteristics of the time-to-graduation problem is that a complete set of ground truth samples for a cohort of students is not available until all the students have either graduated or dropped, i.e. several years after they start. The problem with waiting long enough (or going back far enough in historic data) to get a complete set of ground truth samples is that the statistics may have changed during this time, so that the results obtained with this data may not be representative of current student behavior (and therefore have little predictive value). On the other hand, the problem with using more recent student data is that ground truth is not available for all the students, i.e. many students have not yet graduated or dropped. Nevertheless, once a student has reached semester k we know that their NSS event time is greater than or equal to k, and this is useful information regarding the timing of events. The question is how to use this information to produce reliable estimates of the prediction functions. It is important to note that simply discarding the students that have not yet experienced an event, and then producing function estimates using only the remaining student data (e.g. by employing the estimation method in Section 3) produces biased results that are both inaccurate and unreliable.

Consider the students that have not yet graduated or dropped. The number of semesters they have completed so far is called the *censor time*[3]. Note that this is either an NSS time or NSE time depending on the type of event time chosen. To incorporate censoring, the data samples now take the form (t, c, b) where $c \in \{0, 1\}$ is a censor flag defined as follows.

$$c = \begin{cases} 0, & t = t_e \text{ is the actual (uncensored) event time} \\ 1, & t = t_c \text{ is the censor time (i.e. number of semesters so far)} \end{cases} \tag{5}$$

Note that if $c = 1$ then the event type value b has no meaning. Following the traditional survival analysis approach we let T_c be the censor time random variable taking values $t_c \in \{1, 2, ..., \mathsf{T}_e - 1\}$. Then we define $P_{T_e,T_c,B}$ to be the joint (event time, censor time, event type) distribution model. With this we assume that the observed samples (t, c, b) are formed according to the sample plan below.

Sample Plan 1: First generate N (unobserved) iid samples of the form (t_e, t_c, b_e) according to the distribution $P_{T_e,T_c,B}$. Then compute the observed samples (t, c, b) using

$$(t, c, b) = \begin{cases} (t_e, 0, b_e), & t_e \leq t_c \\ (t_c, 1, b_e), & t_e > t_c \end{cases} \tag{6}$$

Note that the observed time satisfies $t = \min(t_e, t_c)$. Also, if $c = 1$ then the true value of the event type b is not observed in practice even though it is produced as a part of this synthesis process.

Remark 3. Note that the censor time t_c cannot exceed $\mathsf{T}_e - 1$, since T_e is the last possible event time and $t_c = \mathsf{T}_e$ would imply a true event time larger than T_e. Note also that in practice the true value of T_e is generally unknown, but finite. Indeed, most universities allow students to remain active for a very long time. For example, in some cases students remain active for such a long time that they may be forced to repeat early classes that have become outdated, and this kind of behavior can extend the true value of T_e substantially. Such cases make it difficult to know the true

[3]The true event time is viewed as having been *censored*.

value of T_e, even though it is obviously finite because all students will eventually become inactive, through death if nothing else.

The formal problem statement for the TTG problem with censored data is as follows.

TTG-V2: *Let $D_N = ((t_1, c_1, b_1), (t_2, c_2, b_2), ..., (t_N, c_N, b_N))$ be a collection of N samples that represent (time, censor flag, event type) values for students from a particular institution. Assume that these samples are generated according to Sample Plan 1 with an unknown distribution $P_{T_e,T_c,B}$. Given D_N the goal is to estimate the prediction functions (hazard, cessation, survival, and probability) for the institution.*

Before developing a solution to this problem it is worth mentioning some simple, but flawed, methods.

1. The first flawed method simply discards the censored samples and estimates $P_{T_e,B}$ using only the uncensored samples.

2. The second flawed method assumes that the event type is either known or can be reliably estimated for every censored sample. It then treats the censor times as true event times and estimates $P_{T_e,B}$ using the full set of samples.

In both cases the modified data set contains only (event time, event type) samples so that the method described in Section 3.1 could be used to estimate $P_{T_e,B}$. Both of these methods produce *biased estimates* that artificially inflate the probabilities of smaller event times.

Successful solution methods must include both censored and uncensored samples, and must be able to extract the relevant information from them in an unbiased way. This report describes two vastly different approaches that are capable of producing such solutions.

1. **independent censoring:** If censoring is *independent* then we can prove that the *observed* hazard function (defined below) is equal to the *true* hazard function, and since an unbiased estimate of the *observed* hazard function can be obtained directly from the observed data D_N, an unbiased estimate of the *true* hazard function follows. Definitions of *independent* censoring and the *observed* hazard function are provided in Section 4.1, and a method for producing an unbiased estimate of the *observed* hazard function is provided in Section 4.2. Once we have (an estimate of) the hazard function the other prediction functions (probability, cessation, and survival) can be obtained using simple closed form computations (see Section 4.2). The equivalence of the *observed* and *true* hazard functions under the independent censoring assumption is a very powerful result that is heavily exploited in survival analysis. Indeed, the exploitation of this result by employing methods that estimate the *observed* hazard function dominates virtually all traditional methods of survival analysis. In fact this approach is so well developed that it is often used even when censoring cannot be shown to be independent. This helps explain the dominance of the *hazard function* in the survival analysis literature.

2. **semi-supervised learning:** This approach produces an estimate of $P_{T_e,B}$ using the maximum likelihood (ML) method with a so-called *semi-supervised* likelihood function. This approach does not require independent censoring, but typically does require a parametric model of $P_{T_e,B}$[4]. The censored data samples represent a type of missing information that can

[4]It is generally believed that the ML method can only be applied when the distribution model is parametric, and while this is not strictly true, application of the ML method with a nonparametric model is often problematic.

be inferred through the use of the Expectation-Maximization (EM) algorithm to optimize the likelihood function. Indeed, in addition to estimating the parameters of $P_{T_e,B}$, the EM algorithm also estimates the probability of each event type (grad and drop) at each event time for each censored data sample. An example of this approach is developed in detail in Section 10.

This report starts by developing solution methods using the first approach that relies on independent censoring. To this end, the next section (Section 4.1) provides a formal definition of independent censoring, and explains why this assumption enables the use of censored (plus uncensored) data samples to estimate the hazard function. Then, in Section 4.2, we show how to exploit the independent censoring assumption to develop a simple hazard estimation algorithm. We also show how to derive the other prediction functions from the hazard. In Section 4.4 we take a closer look at the censoring mechanism in TTG problem and conclude that it does not conform to the traditional probabilistic censoring model assumed in the survival analysis literature. Instead the TTG problem exhibits a type of censoring that we call *staged censoring*, which satisfies the independent censoring assumption, but admits an alternative synthesis model which plays an important role in the development of advanced estimation algorithms needed to overcome some of the other nontraditional aspects of the TTG problem described later in this report. The development of advanced estimation algorithms begins in Section 5.

4.1 Independent Censoring

Independent censoring is vaguely defined in the literature. Indeed, excerpts from the literature are more likely to describe consequences of the definition rather than provide the definition itself. Most of these descriptions rely on the notion of *at-risk* samples which are defined as follows. A sample (t, c, b) is *at-risk at time* t' if $t \geq t'$. Simply put, an at-risk student is a student that is still active. Three of the better descriptions of independent censoring from the literature are provided below.

- (Singer and Willett [24]) "Independent censoring is the assumption that censoring is unrelated to event occurrence. Under independent censoring, each year's risk set is representative of all students who would be in school in that year; censored individuals do not differ from those who remain. If censoring is not independent, individuals in the risk set differ systematically from censored individuals, and the generalization may be incorrect."

- (John Fox [11]) "By considering only those subjects that are at-risk, unbiased estimates of survival times, survival probabilities, etc., can be made, as long as those at-risk are representative of all subjects. This implies that the censoring mechanism is unrelated to survival time. That is, the distribution of survival times of subjects who are censored at a particular time is no different from that of subjects who are still a-risk at this time. When this is the case, censoring is said to be *noninformative* (i.e. about survival time)."

- (Scott and Kennedy [23]) "We define the *ignorability condition for censoring* as the condition that these two probabilities are the same

$$\frac{\mathsf{Pr}(\text{event } k \text{ at } t)}{\mathsf{Pr}(\text{nothing before } t)} = \frac{\mathsf{Pr}(k \text{ at } t \text{ and not censored by } t)}{\mathsf{Pr}(\text{nothing before } t \text{ and not censored by } t)}$$

  If a particular censoring event satisfies the above, we say that it is *noninformative with respect to event k*. One of the strengths of working with hazard estimates is that, if the ignorability conditions are satisfied, we can ignore censoring, which we define as (1) using data form a

censored subject until the censoring occurs, and then allowing that subject to disappear from subsequent periods, and (2) interpreting hazard estimates in each period just as we would if censoring never occurred."(In this description k is an event type, i.e. in the TTG problem $k = b$.)

These descriptions highlight the most heavily exploited consequence of the independence assumption: namely that the collection of observed at-risk samples (t, c, b) at time t' are representative of the collection of all unobserved samples $(t_e, b) : t_e \geq t'$. This implies that if the (hazard) estimates at time t' are based (exclusively) on the observed at-risk samples at time t' then the censoring mechanism can be ignored because it is irrelevant. Thus, the independence assumption is often referred to as the *ignorability condition* or the *noninformative assumption*.

(Scott and Kennedy [23]) attempt to provide additional insight into the censoring mechanism by describing the following data synthesis process (note that K is the number of event types, so that in the TTG problem $K = |\mathcal{B}| = 2$).

- (Scott and Kennedy [23]) "A common description of such a noninformative censoring process is that it is *independent of events*. If we articulate how we imagine our data to be generated, we can make this criterion precise. One possibility is the following. Imagine that, in each period, two experiments take place: a $(K+1)$-tomous experiment deciding which event occurs to each subject, and a dichotomous experiment determining whether or not the student is censored (K = number of event types). The noncensored students who receive event 0 go to the next stage, where the two experiments occur again, independent of earlier stages. Under this data-generating mechanism, a criterion for censoring to be ignorable is that the censoring experiment and the event experiment are independent at every stage."

None of the descriptions above provide a definition of the independence assumption, but the definition we provide here is quite simple.

Definition 1. Censoring is *independent* if

$$P_{T_e,T_c,B} = P_{T_e,B} P_{T_c}$$

i.e. the censor times are independent of the (event time, event type) values.

This definition implies that data can be synthesized according to the following sample plan.

Sample Plan 2 (independent censoring): Generate N (unobserved) iid samples of the form (t_e, t_c, b_e) where the (t_e, b_e) values are generated according to the distribution $P_{T_e,B}$, and the t_c values are generated according to the distribution P_{T_c}. Then compute the observed samples (t, c, b) using

$$(t, c, b) = \begin{cases} (t_e, 0, b_e), & t_e \leq t_c \\ (t_c, 1, b_e), & t_e > t_c \end{cases} \tag{7}$$

Note that the observed time satisfies $t = \min(t_e, t_c)$. Also, if $c = 1$ then the true value of the event type b is not observed in practice even though it is produced as a part of this synthesis process.

Appendix C shows the equivalence between Sample Plan 2 and the synthesis method described by (Scott and Kennedy [23]) above.

Independent censoring has many consequences, but perhaps the most important is the equivalence between the *true* and *observed* hazard functions established by the theorem below. First note that,

by definition, the marginal distribution P_{T_e} is given by

$$P_{T_e}(t) = \sum_{b \in \mathcal{B}} P_{T_e,B}(t,b)$$

and this allows us to rewrite the hazard function in (3) as follows

$$\begin{aligned} h(t,b) &= \frac{P_{T_e,B}(t,b)}{\sum_{t'=t}^{\mathsf{T}_e} \sum_{b \in \mathcal{B}} P_{T_e,B}(t',b)} \\ &= \frac{P_{T_e,B}(t,b)}{\sum_{t'=t}^{\mathsf{T}_e} P_{T_e}(t')} \\ &= \frac{P_{T_e,B}(t,b)}{P_{T_e}(T_e \geq t)} \end{aligned} \tag{8}$$

Also, let $T_o = \min(T_e, T_c)$ be the observed time random variable, C be the observed censor flag random variable, and $P_{T_o,C,B}$ be the probability distribution of the observed samples. Then the marginal distribution P_{T_o} is given by

$$P_{T_o}(t) = \sum_{c=0}^{1} \sum_{b \in \mathcal{B}} P_{T_o,C,B}(t,c,b)$$

and the *observed* hazard function is defined

$$h_o(t,b) = \frac{P_{T_o,C,B}(t,0,b)}{P_{T_o}(T_o \geq t)} \tag{9}$$

The numerator represents the probability that a student experiences event b at time t and that this event is uncensored. The denominator is the probability that a student is still active at time t and not censored before time t.

Theorem 1. *If censoring is independent then*

$$h_o = h$$

i.e. the true and observed hazard functions are identical.

Proof. See Appendix D. □

Theorem 1 tells us that the following equality holds under the independent censoring assumption.

$$\frac{P_{T_e,B}(t,b)}{P_{T_e}(T_e \geq t)} = \frac{P_{T_o,C,B}(t,0,b)}{P_{T_o}(T_o \geq t)}$$

There is little doubt that (Scott and Kennedy [23]) were referring to this relationship when they said "these two probabilities are the same" in the description above, even though their probability functions were not clearly defined. Once again we emphasize that Theorem 1 is a consequence of Definition 1, not the definition itself.

4.2 The Discrete-time Kaplan-Meier (DKM) Method

Let $D_N = ((t_1, c_1, b_1), (t_2, c_2, b_2), ..., (t_N, c_N, b_N))$ be a collection of N iid samples generated according to Sample Plan 2. If we define

$$\begin{array}{rcl} n(t,c,b) & = & \text{number of samples from } D_N \text{ where } (t_i, c_i, b_i) = (t, c, b) \\ n_{risk}(t) & = & \text{number of samples from } D_N \text{ where } t_i \geq t \end{array} \tag{10}$$

then direct sample estimates of $P_{T_o,C,B}(t, 0, b)$ and $P_{T_o}(T_o \geq t)$ are simply

$$\hat{P}_{T_o,C,B}(t, 0, b) = \frac{n(t, 0, b)}{N}$$

$$\hat{P}_{T_o}(T_o \geq t) = \frac{n_{risk}(t)}{N}$$

Thus, a direct sample estimate of the observed hazard, and by Theorem 1 the true hazard, is simply

$$\hat{h}(t, b) = \frac{n(t, 0, b)}{n_{risk}(t)} \tag{11}$$

This hazard estimate is ubiquitous in the survival analysis literature, e.g. see [1, 6, 15, 25, 23]. Specifically it has been used to analyze the TTG problem in [25, 23]. This hazard estimate is often unnamed, but in this report we will refer to it as the *discrete-time Kaplan-Meier* (DKM) estimate because of its connection to the continuous-time survival analysis method proposed and analyzed by (Kaplan and Meier [16]) in 1958.

4.3 Hazard to Probability

As we have seen, independent censoring allows a simple, direct, and unbiased estimate of the hazard function. But our main goal is to estimate the cessation (or survival) function. This can be accomplished by converting the hazard function estimate into a probability function estimate, and then substituting the probability function estimate into (1) (or (2)). To this end we now develop an expression for the probability function in terms of the hazard function.

First we write the hazard function in (3) as follows

$$h(t, b) = \frac{P_{T_e,B}(t, b)}{P_{T_e}(T_e \geq t)}$$

and then we solve for $P_{T_e,B}(t, b)$ giving

$$P_{T_e,B}(t, b) = h(t, b)P_{T_e}(T_e \geq t) = h(t, b)(1 - P_{T_e}(T_e < t)).$$

Note that we have replaced $P_{T_e}(T_e \geq t)$ by $(1 - P_{T_e}(T_e < t))$ to emphasize that the probability of an event at time t or greater is equivalent to the probability of a non-event at all times less than t. Next we express $(1 - P_{T_e}(T_e < t))$ in terms of the hazard. We start by defining the aggregate hazard function

$$h(t) = \sum_{b \in \mathcal{B}} h(t, b). \tag{12}$$

Since $h(t)$ is the conditioned probability of an event at time t it follows that $1-h(t)$ is the conditioned probability of a non-event at time t. Thus, we can express the probability of a non-event at all times less than t in the form of a product of conditioned non-events at times $1, 2, ..., t-1$ as follows

$$(1 - P_{T_e}(T_e < t)) = \prod_{\acute{t}=1}^{t-1} \left(1 - h(\acute{t})\right)$$

Substituting this into the expression for $P_{T_e,B}(t,b)$ above gives

$$P_{T_e,B}(t,b) = h(t,b)\prod_{\acute{t}=1}^{t-1}\left(1-h(\acute{t})\right)$$

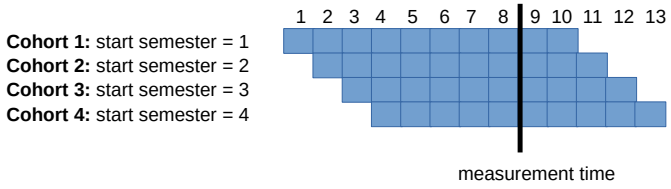
which holds for all $t > 1$. At $t = 1$ we have the identity $h(1,b) = P_{T_e,B}(1,b)$ which follows from the definition of $h(t,b)$ above and the fact that $P_{T_e}(T_e \geq 1) = 1$. Thus the probability function can be expressed in terms of the hazard function as follows

$$P_{T_e,B}(t,b) = \begin{cases} h(t,b), & t = 1 \\ h(t,b)\prod_{t'=1}^{t-1}(1-h(t')), & t > 1 \end{cases} \tag{13}$$

Probability function estimates can now be obtained by substituting hazard function estimates into (13). Then the cessation (or survival) function estimate can be obtained by substituting the probability function estimate into (1) (or (2)).

4.4 Staged Censoring (independent censoring for TTG)

The data synthesis model in **Sample Plan 2** above, where the censor time is modeled as an independent random variable, is consistent with a vast majority of the survival analysis literature, but does not adequately model the type of censoring that occurs in the TTG problem. In the TTG problem the data consists of multiple cohorts of students, each starting at a different absolute semester so they progress through the institution in a staged fashion as illustrated below.



The *measurement time* corresponds to the absolute semester at which the student data is collected. Each cohort is observed for a different length of time L, and censoring occurs for students whose event semester is greater than the cohort observation length. This type of censoring is *independent* because the cohort observation length is independent of the event semester.

The censor mechanism works a little bit differently for the NSS and NSE event times. Let the *event semester* s_e be the semester at which the event occurs. Note that $s_e = t_e$ for NSS, but may differ from t_e for NSE. For both the NSS and NSE, a sample is censored when $s_e > L$. But their censor times are determined differently.

$$NSS:\ t_c = L$$
$$NSE:\ t_c = \text{ number of semesters enrolled over the observation length } L$$

Note that the NSS censor time may correspond to a semester where the student is not enrolled, but this is okay because we know that the true NSS event time must be greater than this value[5].

[5]Note also that the NSS censor time is *deterministic*. However it can be modeled probabilistically using a (trivial) probability model

$$P_{T_c}(t) = \begin{cases} 1, & t = L \\ 0, & otherwise \end{cases}$$

where L is the cohort observation length, i.e. there will be one such probability model for each cohort.

Thus, unlike event times, censor times do not necessarily correspond semesters where the students are enrolled. To implement the NSE censor mechanism we must know more than just the event semester, event time, and observation length, we must also know the student's enrollment status at each semester. Thus, this censor mechanism is not as simple as comparing an event time to an independently generated censor time, and so the traditional data synthesis model in **Sample Plan 2** is inadequate.

We now present a unified staged censoring synthesis model that accommodates both NSS and NSE. Let e_t be a binary-valued variable that denotes a student's enrollment status at semester t according to

$$e_t = \begin{cases} 0, \text{ not enrolled in semester } t \\ 1, \text{ enrolled in semester } t \end{cases}$$

and let $\mathbf{e} = (e_1, e_2, ..., e_{\mathsf{T}_e})$ be the corresponding *enrollment vector* that represents the student's enrollment over all time. Now let $P_{\mathbf{E},B}$ be the probability distribution of the corresponding (enrollment vector, event type) random variable. Let L be the cohort observation length. The event semester s_e for a student with enrollment vector $\mathbf{e}$ is given by

$$s_e \;=\; (\text{largest value of } t \text{ where } e_t = 1) \;=\; \max\{t : e_t = 1\}. \tag{14}$$

Regardless of the type of event time, the student's event will be censored if $s_e > L$. Furthermore, for NSS the event and censor times are given by

$$t_e = s_e, \qquad t_c = L \qquad\qquad (NSS) \tag{15}$$

and for NSE the event and censor times are given by

$$t_e = \sum_{t=1}^{\mathsf{T}_e} e_t, \qquad t_c = \sum_{t=1}^{L} e_t \qquad\qquad (NSE) \tag{16}$$

With this characterization we can now present the unified staged censoring synthesis model in **Sample Plan 3** below.

Remark 4. This unified staged censoring model essentially replaces $P_{T_e,B}$ with $P_{\mathbf{E},B}$ in the synthesis process. But this does not preclude the existence of the probability model $P_{T_e,B}$ for this process. Indeed, the event time random variable T_e is derived from the enrollment vector random variable $\mathbf{E} = (E_1, E_2, ..., E_{\mathsf{T}_e})$ as follows

$$T_e = \begin{cases} \max\{T : E_T = 1\}, & NSS \\ \sum_{T=1}^{\mathsf{T}_e} E_T, & NSE \end{cases}$$

and the corresponding probability model $P_{T_e,B}$ is obtained by transforming $P_{\mathbf{E},B}$ through this process. Thus, the definitions of the hazard, cessation, and survival functions are unchanged, i.e. they continue to be defined in terms of the probability model $P_{T_e,B}$. The difference here is that synthesis cannot be accomplished by simply comparing an event time to an independently generated censor time.

Sample Plan 3 (staged censoring):
- Let $\mathcal{K}$ = number student cohorts, and let $\kappa_1 \leq \kappa_2 \leq ... \leq \kappa_{\mathcal{K}}$ be the ordered cohort labels.
- Let L_{κ_i} be the observation length of cohort κ_i.
- Let N_{κ_i} be the number of students in cohort κ_i. The values of N_{κ_i} may be determined by drawing $\mathcal{K}$ iid samples from a cohort size distribution P_N. If this is the case then we assume that this distribution is independent of all other distributions.

for (each cohort κ_i) **do**
- draw N_{κ_i} iid (enrollment vector, event type) samples $(\mathbf{e}, b_e)$ from distribution $P_{\mathbf{E},B}$
- for each $\mathbf{e}$ compute the event semester $s_e = \max\{t : e_t = 1\}$
- for each sample, compute the (event time, censor time) values (t_e, t_c) as follows

$$t_e = s_e, \qquad t_c = L_{\kappa_i} \qquad (NSS)$$

$$t_e = \sum_{t=1}^{\mathsf{T}_e} e_t, \qquad t_c = \sum_{t=1}^{L_{\kappa_i}} e_t \qquad (NSE)$$

- for each sample (s_e, t_e, t_c, b_e) generate an observed sample of the form (κ, t, c, b) = (cohort label, time value, censor flag, event type) by applying the following operation

$$(\kappa, t, c, b) = \begin{cases} (\kappa_i, t_e, 0, b_e), & s_e \leq L_{\kappa_i} \\ (\kappa_i, t_c, 1, b_e), & s_e > L_{\kappa_i} \end{cases}$$

end for

The total number of observed samples from all cohorts is $N = N_{\kappa_1} + N_{\kappa_2} + ... + N_{\kappa_{\mathcal{K}}}$ and the complete data set is denoted D_N. Note that if $c = 1$ then the true value of the event type b is not observed in practice even though it is produced as a part of this synthesis process.

4.5 TTG V2 Examples

This section shows two examples, one for NSS and one for NSE. These examples use synthetic data generated according to **Sample Plan 3**. Because the data is synthetic the true prediction functions are known, and this allows us to compare true and estimated functions. These examples also introduce a visualization for TTG data called the *staged data table*.

4.5.1 Example with the NSS Option

In this example we generate 5 cohorts of data, each with 800 samples. This data was generated according to **Sample Plan 3** with the NSS event time. A summary of this data is presented in the *staged data table* below.

```
-------------------------------------------------------------
Cohort 1: nsamples  800
   ngrad:      0    0   76  107   65  128   52   22   30    0
   ndrop:     54   52   18   15   17   16   14   41   44   49
   ncensor:    0    0    0    0    0    0    0    0    0    0
   nrisk:    800  746  694  600  478  396  252  186  123   49
Cohort 2: nsamples  800
   ngrad:      0    0   75  127   68  117   53   19    -    -
   ndrop:     66   43   19   13   14   22   11   40    -    -
   ncensor:    0    0    0    0    0    0    0  113    -    -
   nrisk:    800  734  691  597  457  375  236  172    0    0
Cohort 3: nsamples  800
   ngrad:      0    0   52  139   73  139    -    -    -    -
   ndrop:     71   53   12   14    8   15    -    -    -    -
   ncensor:    0    0    0    0    0  224    -    -    -    -
   nrisk:    800  729  676  612  459  378    0    0    0    0
Cohort 4: nsamples  800
   ngrad:      0    0   66  120    -    -    -    -    -    -
   ndrop:     57   52   13   20    -    -    -    -    -    -
   ncensor:    0    0    0  472    -    -    -    -    -    -
   nrisk:    800  743  691  612    0    0    0    0    0    0
Cohort 5: nsamples  800
   ngrad:      0    0    -    -    -    -    -    -    -    -
   ndrop:     67   53    -    -    -    -    -    -    -    -
   ncensor:    0  680    -    -    -    -    -    -    -    -
   nrisk:    800  733    0    0    0    0    0    0    0    0

Semester       1    2    3    4    5    6    7    8    9   10
-------------------------------------------------------------
```

The t^{th} column of the table corresponds to the t^{th} semester of each cohort. In this example each cohort starts 2 semesters after the previous cohort. If the first semester of Cohort 1 is assigned the *absolute* semester number 1 then the *absolute* starting semester numbers for cohorts 1,2,3,4,5 are 1,3,5,7,9 and all cohorts are *measured* after *absolute* semester 10. The event and censor times in the TTG analysis are expressed as *relative* semester numbers which correspond to the column numbers in the table, i.e. the 3rd semester of Cohort 4 (in column 3) corresponds to absolute semester number 9. Each cohort contains four rows of aggregate counts:

- ngrad = number of cohort students that graduate at each semester,
- ndrop = number of cohort students that drop at each semester,
- ncensor = number of censored cohort students at each semester, and
- nrisk = number of at-risk cohort students at each semester.

This table reveals numerous insights into the data.

- Cohort 1 has no censored samples because its last semester is equal to the largest possible event time in the synthesis model ($\mathsf{T}_e = 10$), i.e. Cohort 1 is observed for *all possible event times*. (This is unlikely to be the case with real world data.)
- All censored samples appear in a Cohort's last semester because it is not possible to have censored samples in earlier semesters with the NSS event time.

- The number of censored samples increases as the Cohort observation length decreases (as expected).
- The least likely graduation semesters are 1 and 2 as evidenced by the fact that no students in this data set graduate during these semesters.
- The most likely graduation semesters are 4 and 6, and the most likely drop semesters are 1 and 2.
- It is easy to produce a *cohort-specific* unbiased estimate of probability function values directly from the table entries. For example, from Cohort 3 we can produce the estimates

$$\hat{P}_{T_e,B}(4,\mathsf{G}) = \frac{139}{800} \qquad \hat{P}_{T_e,B}(4,\mathsf{D}) = \frac{14}{800}$$

  It is also easy to produce more accurate estimates by combining the contribution from multiple cohorts. For example we can combine the data from Cohorts 1,2,3 to produce the estimate

$$\hat{P}_{T_e,B}(6,\mathsf{G}) = \frac{128+117+139}{800+800+800}$$

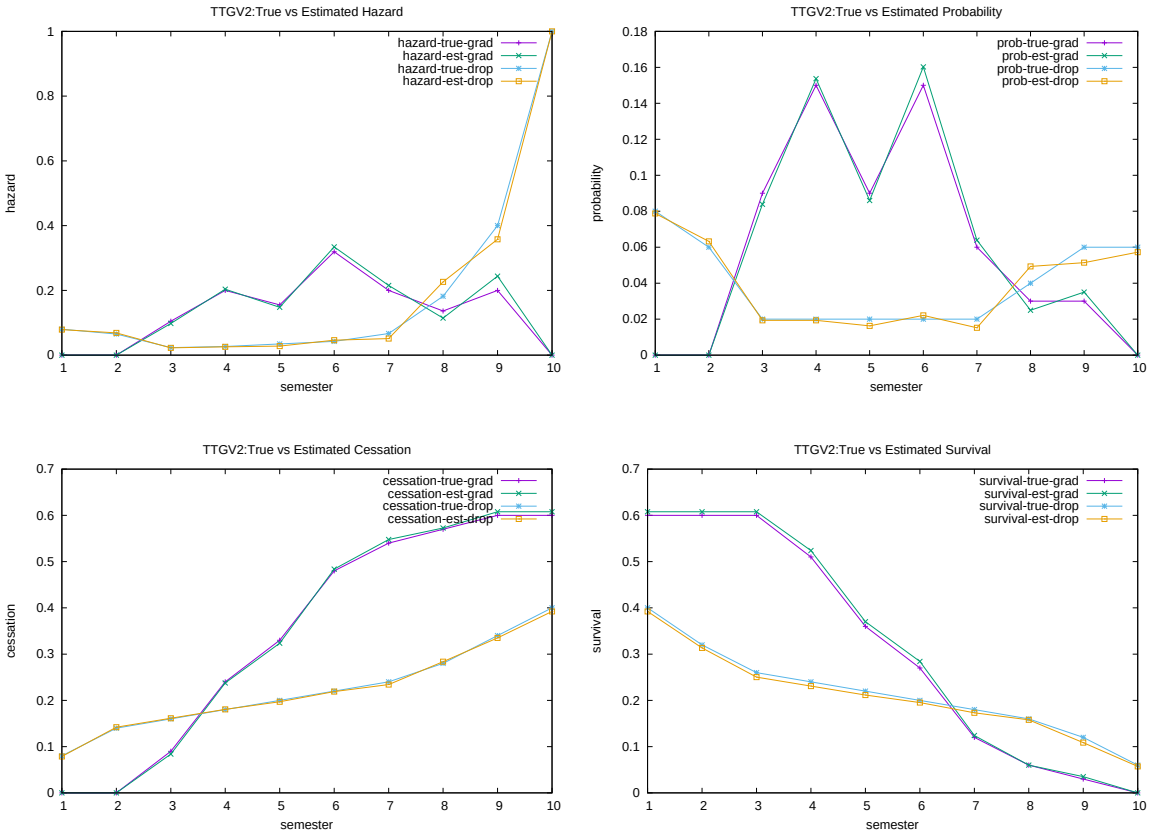
  This type of direct estimate omits the potential contribution of censored samples. In addition it is not valid for NSE event times, as explained in the next section.
- Regardless of how function estimates are produced, it is clear that estimates at later semesters will be derived from fewer samples, and therefore be less accurate.
- The entries in this table correspond to the sample counts in (10) as follows

$$\begin{aligned} n(t,0,\mathsf{G}) &= \sum_{\text{all cohorts}} \mathsf{ngrad}(t) \\ n(t,0,\mathsf{D}) &= \sum_{\text{all cohorts}} \mathsf{ndrop}(t) \\ n_{risk}(t) &= \sum_{\text{all cohorts}} \mathsf{nrisk}(t) \end{aligned}$$

  Thus, it is easy to produce hazard estimates by plugging the table entries directly into (11). For example,

$$\hat{h}(8,\mathsf{G}) = \frac{22+19}{186+172} = \frac{41}{358}$$

Prediction function estimates are obtained using the DKM method in (11) to estimate the hazard, then substituting into (13) to obtain the probability, and finally substituting into (1) and (2) to obtain the cessation and survival function estimates. These estimates, along with the true function values, are shown below. Note that the estimates are very close to the true functions with the largest differences occurring at later semesters.



Remark 5. It might be interesting to compare the DKM probability estimates with the direct probability estimates from the table. The direct estimates are simpler, but the DKM estimates incorporate the censored samples, so it is not clear which would be more accurate. This issue is only relevant when the data is formed with the NSS event time.

4.5.2 Example with the NSE Option

In this example we generate 5 cohorts of data, each with 800 samples. This data was generated according to **Sample Plan 3** with the NSE event time. A summary of this data is presented in the *staged data table* below.

```
-------------------------------------------------------------
Cohort 1: nsamples  800
   ngrad:     29   29   84  106   71   85   37   28   13    0
   ndrop:     87   31   22   18   18   11   16   38   47   30
   ncensor:    0    0    0    0    0    0    0    0    0    0
   nrisk:    800  684  624  518  394  305  209  156   90   30
Cohort 2: nsamples  800
   ngrad:     19   27   80  110   73   83   29   18    0    0
   ndrop:     98   31   15   21   17   22   15   21    0    0
   ncensor:    0    0    0    2    7   12   26   74    0    0
   nrisk:    800  683  625  530  397  300  183  113    0    0
Cohort 3: nsamples  800
   ngrad:     22   36   93  102   73   66    0    0    0    0
   ndrop:     80   40   15   20    7    7    0    0    0    0
   ncensor:    1    3   17   11   39  168    0    0    0    0
   nrisk:    800  697  618  493  360  241    0    0    0    0
Cohort 4: nsamples  800
   ngrad:      9   28   80   77    0    0    0    0    0    0
   ndrop:     93   35   12   10    0    0    0    0    0    0
   ncensor:   13   22   62  359    0    0    0    0    0    0
   nrisk:    800  685  600  446    0    0    0    0    0    0
Cohort 5: nsamples  800
   ngrad:      0    0    0    0    0    0    0    0    0    0
   ndrop:     75   30    0    0    0    0    0    0    0    0
   ncensor:   81  614    0    0    0    0    0    0    0    0
   nrisk:    800  644    0    0    0    0    0    0    0    0

Semester       1    2    3    4    5    6    7    8    9   10
-------------------------------------------------------------
```

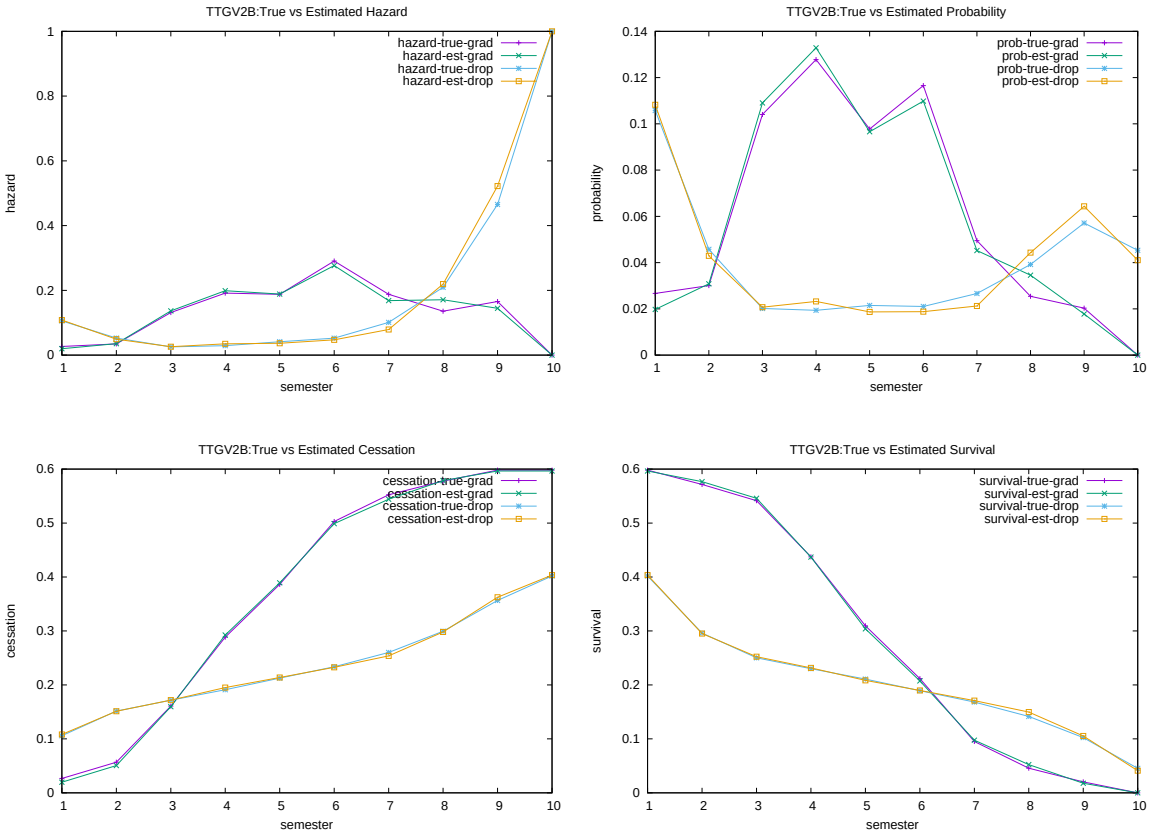
Numerous insights can be gleaned from this table.

- Cohort 1 has no censored samples because it is observed for *all possible event times*. (This is unlikely to be the case with real world data.)
- The censored samples appear across all semesters because the *number of semesters enrolled* is less than the *number of semesters since the start* for some students.
- The appearance of censored samples across all semesters prevents us from forming direct estimates of the probability function as we did in the previous section. For example, the direct estimate
$$\hat{P}_{T_e,B}(6, \mathsf{G}) = \frac{85 + 83 + 66}{800 + 800 + 800}$$
is *biased low* because the numerator is missing some counts which are currently "hidden" in the censored sample counts from earlier semesters.
- The number of censored samples increases as the Cohort observation length decreases (as expected).
- The least likely graduation semesters are at the beginning and end, while the most likely graduation semesters are in the middle.
- The most likely drop semesters are at the beginning and the end, while the least likely drop semesters are in the middle.

- Regardless of how function estimates are produced, it is clear that estimates at later semesters will be derived from fewer samples, and therefore be less accurate.
- The entries in this table correspond to the sample counts in (10) in the same way as the previous section. Thus, it is easy to produce hazard estimates by plugging the table entries directly into (11). For example,

$$\hat{h}(8, \mathsf{G}) = \frac{28 + 18}{156 + 113} = \frac{41}{269}$$

Prediction function estimates are obtained from this data using the DKM method in (11) to estimate the hazard, then substituting into (13) to obtain the probability, and finally substituting into (1) and (2) to obtain the cessation and survival function estimates. These estimates, along with the true function values, are shown below. Note that the estimates are very close to the true functions with the largest differences occurring at later semesters.



5 TTG Version 3 (drop estimation and labeling errors)

In this next version of the TTG problem we address *labeling errors* that occur because of the ambiguity in determining the dropout label. We start with the following definitions.

- **Stopout:** A *stopout* is a disruption in enrollment, i.e. one or more semesters of non-enrollment followed by one or more semesters of enrollment.

- **Dropout:** A *dropout* is a termination of enrollment[6].

Note that it is impossible to distinguish between stopouts and dropouts with perfect accuracy because we never know if a student will enroll again in the future. Thus a dropout estimation procedure is required, and knowledge of its accuracy it is essential if we wish to assess the accuracy of inferences drawn from this data. The following is a rather obvious dropout estimation procedure proposed by (Scott and Kennedy [23]).

Dropout Estimation Procedure: A student is labeled as *drop* if the student has not graduated and not enrolled in the most recent K semesters.

This procedure introduces a K semester *delay* in the determination of the *drop* label relative to the *grad* label. Indeed, while the *grad* label can be correctly assigned for all semesters up to the most recent, the *drop* label will be unknown for the past K semesters. The choice of K involves a trade-off between the amount of label error and the amount of data loss, and is discussed in more detail in Section 5.2.

The procedure above produces the following two types of labeling errors.

1. **drop-to-censor:** In this case a student who has already *dropped* is mislabeled as *censored* because the student has not yet met the criterion of non-enrollment for K consecutive semesters. This leads to an under-representation of dropouts, i.e. a dropout bias, that is described in detail in Section 5.1 below. Sections 5.1.2-5.1.4 describe methods for mitigating this bias.

2. **censor-to-drop:** In this case a student has failed to enroll over the past K (or more) semesters but enrolls in a future semester and so is mislabeled as *drop*, but should be labeled as *censor*. This student will eventually *drop* or *grad*, but we won't know which until later. It turns out that most students that exhibit this type of behavior will eventually drop, so the *drop* label is usually correct, but the drop time may be premature resulting in a bias towards earlier drop times. This type of labeling error is analyzed in detail in Section 5.2.

The synthesis model for data produced with this estimation procedure differs from the previous staged censoring model (i.e. **Sample Plan 3**) in two ways. First, the effective observation time for *drop* samples is now $L - K$ instead of L, but it remains at L for *grad* samples. Second, the determination of observed samples is complicated by the fact that any student that hasn't graduated and experiences non-enrollment over the past K semester will be assigned the *drop* label, regardless of their actual status. Let $\mathbf{e} = [e_1, e_2, ..., e_{\mathsf{T}_e}]$ be a student enrollment vector, and let

$$d = \begin{cases} 1, & \text{if } e_t = 0 \text{ for all } t \text{ satisfying } L - K < t \leq L \\ 0, & \text{otherwise} \end{cases}$$

be a flag that detects the assigned drop situation. If the student's event semester s_e exceeds the effective observation length and $d = 0$ then the censor times are the same as before

$$t_c = L, \qquad (NSS)$$

$$t_c = \sum_{t=1}^{L} e_t, \qquad (NSE)$$

[6]Dropouts typically occur when a student decides to abandon the quest for a degree, at least for the time being. In this report however, dropouts also include cases where a student transfers to another institution (prior to completing a degree at the current institution).

but if $d = 1$ then the assigned drop times will be

$$t_d = \text{ most recent enrolled semester } = \max\{t : e_t = 1, t \le L\}, \qquad (NSS)$$

$$t_d = \sum_{t=1}^{L} e_t, \qquad (NSE)$$

Note that $t_c = t_d$ for the NSE option, but t_c is not necessarily equal to t_d for the NSS option because a drop event time must correspond to an enrolled semester. With these definitions an observed sample (t, c, b) is produced by one of the following six cases.

$$(t, c, b) = \begin{cases} (t_e, 0, b_e), & (b_e = \mathsf{G}) \text{ and } (s_e \le L) & (a) \\ (t_c, 1, b_e), & (b_e = \mathsf{G}) \text{ and } (s_e > L) \text{ and } (d = 0) & (b) \\ (t_d, 0, \mathsf{D}), & (b_e = \mathsf{G}) \text{ and } (s_e > L) \text{ and } (d = 1) & (c) \\ (t_e, 0, b_e), & (b_e = \mathsf{D}) \text{ and } (s_e \le L - K) & (d) \\ (t_c, 1, b_e), & (b_e = \mathsf{D}) \text{ and } (s_e > L - K) \text{ and } (d = 0) & (e) \\ (t_d, 0, \mathsf{D}), & (b_e = \mathsf{D}) \text{ and } (s_e > L - K) \text{ and } (d = 1) & (f) \end{cases} \qquad (17)$$

Cases (a) and (d) correspond to situations where the true event is observed. Note that the value of d in case (a) is irrelevant since the student has already graduated, and the value of d in case (d) is always equal to 1 because the student has dropped prior to the last K semesters. Cases (b) and (e) correspond to situations where the student is censored because the event semester exceeds the effective observation length. Case (e) produces to a **drop-to-censor** error if the event semester satisfies $L - K < s_e \le L$. Cases (c) and (f) correspond to situations where the assigned drop always produces a **censor-to-drop** error. Specifically, case (c) erroneously assigns a *drop* label instead of a *censor* label to some students whose true label is *grad* and whose event semester satisfies $s_e > L$, and case (f) assigns an erroneous (premature) time value for students whose event semester satisfies $s_e > L$ [7].

If we define the effective observation time $\tilde{L}$ to be

$$\tilde{L} = \begin{cases} L, & b_e = \mathsf{G} \\ L - K, & b_e = \mathsf{D} \end{cases} \qquad (18)$$

then the six cases above can be simplified to the three cases below.

$$(t, c, b) = \begin{cases} (t_e, 0, b_e), & (s_e \le \tilde{L}) \\ (t_c, 1, b_e), & (s_e > \tilde{L}) \text{ and } (d = 0) \\ (t_d, 0, \mathsf{D}), & (s_e > \tilde{L}) \text{ and } (d = 1) \end{cases}$$

With this, the complete synthesis model for data produced with the dropout estimation procedure is shown below.

[7] Note that $s_e > L$ for all samples in case (f) because $d = 1$ implies that $e_t = 0$ for $L - K < t \le L$.

Sample Plan 4 (staged censoring with dropout estimation):
- Let $\mathcal{K}$ = number student cohorts, and let $\kappa_1 \leq \kappa_2 \leq ... \leq \kappa_{\mathcal{K}}$ be the ordered cohort labels.
- Let L_{κ_i} be the observation length of cohort κ_i.
- Let N_{κ_i} be the number of students in cohort κ_i. The values of N_{κ_i} may be determined by drawing $\mathcal{K}$ iid samples from a cohort size distribution P_N. If this is the case then we assume that this distribution is independent of all other distributions.

for (each cohort κ_i) **do**
- draw N_{κ_i} iid (enrollment vector, event type) samples $(\mathbf{e}, b_e)$ from distribution $P_{\mathbf{E},B}$
- for each $\mathbf{e}$ compute the event semester $s_e = \max\{t : e_t = 1\}$
- for each sample, compute the (event time, censor time) values (t_e, t_c) as follows

$$t_e = s_e, \qquad t_c = L_{\kappa_i}, \qquad t_d = \max\{t : e_t = 1, t \leq L_{\kappa_i}\} \qquad (NSS)$$

$$t_e = \sum_{t=1}^{\mathsf{T}_e} e_t, \qquad t_c = \sum_{t=1}^{L_{\kappa_i}} e_t, \qquad t_d = \sum_{t=1}^{L_{\kappa_i}} e_t \qquad (NSE)$$

- for each $\mathbf{e}$ compute the drop estimation flag

$$d = \begin{cases} 1, & \text{if } e_t = 0 \text{ for all } t \text{ satisfying } L_{\kappa_i} - K < t \leq L_{\kappa_i} \\ 0, & \text{otherwise} \end{cases}$$

- for each sample $(s_e, t_e, t_c, t_d, b_e, d)$ generate an observed sample of the form (κ, t, c, b) = (cohort label, time value, censor flag, event type) by applying the following operation

$$(\kappa, t, c, b) = \begin{cases} (\kappa_i, t_e, 0, b_e), & (s_e \leq \tilde{L}_{\kappa_i}) \\ (\kappa_i, t_c, 1, b_e), & (s_e > \tilde{L}_{\kappa_i}) \text{ and } (d = 0) \\ (\kappa_i, t_d, 0, \mathsf{D}), & (s_e > \tilde{L}_{\kappa_i}) \text{ and } (d = 1) \end{cases}$$

where $\tilde{L}_{\kappa_i}$ given by (18).

end for

The total number of observed samples from all cohorts is $N = N_{\kappa_1} + N_{\kappa_2} + ... + N_{\kappa_{\mathcal{K}}}$ and the complete data set is denoted D_N. Note that if $c = 1$ then the true value of the event type b is not observed in practice even though it is produced as a part of this synthesis process.

The formal problem statement for the TTG problem with staged censoring and dropout estimation is as follows.

TTG-V3: *Let $D_N = ((\kappa_1, t_1, c_1, b_1), (\kappa_2, t_2, c_2, b_2), ..., (\kappa_N, t_N, c_N, b_N))$ be a collection of N samples that represent (cohort label, time, censor flag, event type) values for students from a particular institution. Assume that these samples are generated according to Sample Plan 4 with an unknown distribution $P_{\mathbf{E},B}$. Given D_N the goal is to estimate the prediction functions (hazard, cessation, survival, and probability) for the institution.*

5.1 Drop-to-Censor Error and Drop Label Delay Bias

The dropout estimation procedure above introduces a *delay* in the determination of the *drop* label relative to the *grad* label. This means that dropout samples will be *under-represented* relative to grad samples. Indeed, dropout samples will be completely missing from the last K semesters of

each cohort.

The staged data table below shows UNM FTFT Cohorts 2012 through 2017 with the NSS option and with drop samples estimated using $K = 2$ semesters. The missing drop values in the last $K = 2$ semesters of each cohort are denoted by the * symbol. It is obvious that some of the censored samples should be drop samples instead. For example, (Cohort 2012, semester 9) and (Cohort 2013, semester 9) have 57 and 48 drop samples respectively, and therefore we expect a similar number from Cohort 2014[8], that is we expect approximately 50 of the 644 censored samples from (Cohort 2014, semester 9) to be drops. This data set is clearly biased in favor of grad over drop.

```
-------------------------------------------------------------------------------
UNM FTFT Data (Option NSS,K=2)
Cohort Fall 2012:  nsamples 3424
  ngrad:      0    0    0    3    6   41   53  644  351  379   96  126   36
  ndrop:    260  392  169  191  123   98   56   59   57   50   29    *    *
  ncensor:    0    0    0    0    0    0    0    0    0    0    0   36  169
  nrisk:   3424 3164 2772 2603 2409 2280 2141 2032 1329  921  492  367  205
Cohort Fall 2013:  nsamples 3518
  ngrad:      0    0    0    0    6   48  106  886  313  340   86    -    -
  ndrop:    293  347  187  189   87  102   60   58   48    *    *    -    -
  ncensor:    0    0    0    0    0    0    0    0    0   48  314    -    -
  nrisk:   3518 3225 2878 2691 2502 2409 2259 2093 1149  788  400    -    -
Cohort Fall 2014:  nsamples 3132
  ngrad:      0    0    0    1    2   64  121  891  220    -    -    -    -
  ndrop:    272  278  152  167  105   86   56    *    *    -    -    -    -
  ncensor:    0    0    0    0    0    0    0   73  644    -    -    -    -
  nrisk:   3132 2860 2582 2430 2262 2155 2005 1828  864    -    -    -    -
Cohort Fall 2015:  nsamples 3327
  ngrad:      0    0    0    4    7   92  143    -    -    -    -    -    -
  ndrop:    258  342  204  178  130    *    *    -    -    -    -    -    -
  ncensor:    0    0    0    0    0   97 1872    -    -    -    -    -    -
  nrisk:   3327 3069 2727 2523 2341 2204 2015    -    -    -    -    -    -
Cohort Fall 2016:  nsamples 3402
  ngrad:      0    0    0    9   13    -    -    -    -    -    -    -    -
  ndrop:    290  404  233    *    *    -    -    -    -    -    -    -    -
  ncensor:    0    0    0  207 2246    -    -    -    -    -    -    -    -
  nrisk:   3402 3112 2708 2475 2259    -    -    -    -    -    -    -    -
Cohort Fall 2017:  nsamples 3219
  ngrad:      0    0    0    -    -    -    -    -    -    -    -    -    -
  ndrop:    386    *    *    -    -    -    -    -    -    -    -    -    -
  ncensor:    0  437 2396    -    -    -    -    -    -    -    -    -    -
  nrisk:   3219 2833 2396    -    -    -    -    -    -    -    -    -    -

Semester      1    2    3    4    5    6    7    8    9   10   11   12   13
-------------------------------------------------------------------------------
```

We now explore various bias mitigation strategies. First we show how a simple modification to the DKM method can produce unbiased estimates from the biased representation. Then we present three different methods that modify the data in an attempt to create unbiased representations that can be incorporated into any inference method.

[8] Actually we expect a slightly smaller number of drops because Cohort 2014 starts with slightly fewer students.

5.1.1 Bias Mitigation via Modified DKM Method

Consider the staged data table above. Note that the bias in this representation is a by-product of the errors in the ndrop and ncensor counts in the last $K = 2$ time slots of each cohort. Recall that the DKM hazard estimates can be formed by plugging the staged data table entries directly into (11). For example, DKM estimates of h_{grad} at $t = 8, 10, 12$ can be formed from the staged data table above as follows

$$\hat{h}_{grad}(12) = \frac{n(12, 0, \mathsf{G})}{n_{risk}(12)} = \frac{126}{367}$$
$$\hat{h}_{grad}(10) = \frac{n(10, 0, \mathsf{G})}{n_{risk}(10)} = \frac{379 + 340}{921 + 788}$$
$$\hat{h}_{grad}(8) = \frac{n(8, 0, \mathsf{G})}{n_{risk}(8)} = \frac{644 + 886 + 891}{2032 + 2093 + 1828}$$

Note that these estimates are independent of the ndrop and ncensor counts, i.e. they depend only on the ngrad and nrisk counts which are uncorrupted. Thus, these estimates are unbiased. Indeed, all such estimates of h_{grad} from this table are unbiased, accurate, and able to exploit all available data.

Note also that there is unbiased evidence for h_{drop} at nearly every semester. For example, even though the Cohort 2013 drop values are missing at $t = 11$, an unbiased estimate of the drop hazard value for this semester can be obtained by applying the DKM method to Cohort 2012 data and is given by $h_{drop}(11) \approx 29/492 = .06$. This estimate is unbiased because the ndrop value for Cohort 2012 is uncorrupted. Similarly, we can form unbiased estimates at $t = 7, 9$ by using uncorrupted cohort data and ignoring corrupted cohort data as follows

$$\hat{h}_{drop}(9) = \frac{57 + 48}{1329 + 1149}$$
$$\hat{h}_{drop}(7) = \frac{53 + 106 + 121}{2141 + 2259 + 2005}$$

This approach can be used to produce unbiased estimates of h_{drop} at all semesters except the very last $K = 2$ semesters on the far right, i.e. $t = 12, 13$, because these semesters contain no uncorrupted drop data.

This modified DKM method is extremely simple, exploits all available ground truth information, and produces a nearly complete set of unbiased hazard estimates. However, this approach does not produce an unbiased representation, it simply manipulates the biased representation, and therefore it is not compatible with most other estimation methods. Therefore, the next three sections present different methods that modify the individual data samples in an attempt to create unbiased representations that can be incorporated into any inference method.

5.1.2 Bias Mitigation via Sample Deletion

In this section we describe a bias mitigation strategy proposed by (Scott and Kennedy [23]) who where the first to draw attention to the dropout delay bias issue. Here is an excerpt from their paper that describes their strategy. (Note that they have chosen $K = 4$ semesters, i.e. 2 years.)

> (Scott and Kennedy [23]) "We make an important point here: given our dropout standard, it is impossible for us to detect a dropout semester during any student's last 2 years of data (i.e., during the last four calendar-time semesters for which data are available). Accordingly, while we can use these last 2 years to identify dropouts, we must disregard, for example, degrees attained in these years."

This excerpt suggests discarding all students who have graduated in the last K semesters so that the remaining data contains students whose *grad* and the *drop* labels are determined over the same time intervals. While this may sound like a reasonable approach, it has several disadvantages.

- First, it results in a loss of good data. Consider discarding all students who have graduated in the last $K = 2$ semesters from the UNM data above. This results in the loss of a significant fraction of the data set.
- Second, it has its own particular type of bias.
- Third, it is unclear what to do with the censored samples in the last K semesters.

We address the second and third issues simultaneously. First let us assume that, prior to the sample discard, the last K semesters contain both grad and censored samples. Then deleting the grad samples and keeping the censored samples will result in a biased representation because it suggests none of the samples in the last K semesters are grad samples, and we know this is not true. For example, if we restrict to a single cohort of data, the DKM estimate of h_{grad} would be zero for the last K semesters, and we know that this is not correct. Now let us assume that, prior to the sample discard, the last K semesters contain only censored samples, some of which are actually mislabeled drop samples. Then simply keeping these censored samples will also result in a biased representation because it suggests that there are no dropouts in the last K semesters, and we know this is not true. For example, if we restrict to a single cohort of data, the DKM estimate of h_{drop} would be zero for the last K semesters, and we know that this is not correct.

Another option would be to delete both the grad and censored samples from the last K semesters. This may be what (Scott & Kennedy) intended. The UNM FTFT Cohort staged data table for this case is shown below, where deleted samples are indicated by the # symbol. A comparison with the previous table reveals that the total number of deleted samples is quite large. Unfortunately even this approach provides a biased representation because the number of at-risk samples is smaller than it should be at every semester, causing the hazard estimates to be biased high. In addition, under the NSS option this approach removes *all* censored samples from the data set, and since we know there are students with censored event times this representation is clearly incorrect. Next we explore an alternative bias mitigation strategy that, instead of deleting samples, reassigns their label and time values.

```
-------------------------------------------------------------------------
UNM FTFT Data (Option NSS,K=2)
Cohort Fall 2012:  nsamples 3424
  ngrad:      0    0    0    3    6   41   53  644  350  380   96    #    #
  ndrop:    260  392  169  191  123   98   56   59   57   50   29    *    *
  ncensor:    0    0    0    0    0    0    0    0    0    0    0    #    #
Cohort Fall 2013:  nsamples 3518
  ngrad:      0    0    0    0    6   48  106  886  313    #    #    -    -
  ndrop:    293  347  187  189   87  102   60   58   48    *    *    -    -
  ncensor:    0    0    0    0    0    0    0    0    0    #    #    0    0
Cohort Fall 2014:  nsamples 3132
  ngrad:      0    0    0    1    2   64  121    #    #   -    -    -    -
  ndrop:    272  278  152  167  105   86   56    *    *    -    -    -    -
  ncensor:    0    0    0    0    0    0    0    #    #    0    0    0    0
Cohort Fall 2015:  nsamples 3327
  ngrad:      0    0    0    4    7    #    #   -    -    -    -    -    -
  ndrop:    258  342  204  178  130    *    *    -    -    -    -    -    -
  ncensor:    0    0    0    0    0    #    #    0    0    0    0    0    0
Cohort Fall 2016:  nsamples 3402
  ngrad:      0    0    0    #    #   -    -    -    -    -    -    -    -
  ndrop:    290  404  233    *    *    -    -    -    -    -    -    -    -
  ncensor:    0    0    0    #    #    0    0    0    0    0    0    0    0
Cohort Fall 2017:  nsamples 3219
  ngrad:      0    #    #   -    -    -    -    -    -    -    -    -    -
  ndrop:    386    *    *    -    -    -    -    -    -    -    -    -    -
  ncensor:    0    #    #    0    0    0    0    0    0    0    0    0    0

Semester      1    2    3    4    5    6    7    8    9   10   11   12   13
-------------------------------------------------------------------------
```

5.1.3 Bias Mitigation via Sample Reassignment

The main idea here is to process the data so that all samples have the same *effective* observation length. This means artificially shortening the observation length from L to $L-K$ for samples that were originally assigned the grad and censor labels. This has the following effect.

- For the NSS option all the grad and censored samples in the last K semesters are moved to semester $(L-K)$, and they are all now labeled as censored samples. This means that their time values are changed to $L-K$ and their censor flags are set to $c=1$.
- For the NSE option all grad and censor samples from the last K semesters are reassigned by processing their enrollment vectors $\mathbf{e}$ using observation length $L-K$ instead of L. This means that their censor flags will be set to $c=1$ so they are all now labeled as censored samples, and their new time values are computed using

$$t_c = \sum_{t=1}^{L-K} e_t$$

Application of this method to UNM FTFT Cohorts with the NSS option produces the staged data table below, where the original location of reassigned samples is shown with the r character. This table differs from the previous staged data table produced using the sample deletion method because of the ncensor values. This method gives an unbiased representation with a balanced depiction of

grad and drop samples and correct censor sample placement throughout. But it provides no data for the last K semesters (i.e. semesters 12 and 13 in the table below). It also reduces the number of samples that contribute to the hazard estimates at later times.

```
-------------------------------------------------------------------------
UNM FTFT Data (Option NSS,K=2)
Cohort Fall 2012:  nsamples 3424
  ngrad:      0    0    0    3    6   41   53  644  350  380   96    r    r
  ndrop:    260  392  169  191  123   98   56   59   57   50   29    *    *
  ncensor:    0    0    0    0    0    0    0    0    0    0  367    r    r
Cohort Fall 2013:  nsamples 3518
  ngrad:      0    0    0    0    6   48  106  886  313    r    r    -    -
  ndrop:    293  347  187  189   87  102   60   58   48    *    *    -    -
  ncensor:    0    0    0    0    0    0    0    0  788    r    r    0    0
Cohort Fall 2014:  nsamples 3132
  ngrad:      0    0    0    1    2   64  121    r    r    -    -    -    -
  ndrop:    272  278  152  167  105   86   56    *    *    -    -    -    -
  ncensor:    0    0    0    0    0    0 1828    r    r    0    0    0    0
Cohort Fall 2015:  nsamples 3327
  ngrad:      0    0    0    4    7    r    r    -    -    -    -    -    -
  ndrop:    258  342  204  178  130    *    *    -    -    -    -    -    -
  ncensor:    0    0    0    0 2204    r    r    0    0    0    0    0    0
Cohort Fall 2016:  nsamples 3402
  ngrad:      0    0    0    r    r    -    -    -    -    -    -    -    -
  ndrop:    290  404  233    *    *    -    -    -    -    -    -    -    -
  ncensor:    0    0 2475    r    r    0    0    0    0    0    0    0    0
Cohort Fall 2017:  nsamples 3219
  ngrad:      0    r    r    -    -    -    -    -    -    -    -    -    -
  ndrop:    386    *    *    -    -    -    -    -    -    -    -    -    -
  ncensor: 2833    r    r    0    0    0    0    0    0    0    0    0    0

Semester      1    2    3    4    5    6    7    8    9   10   11   12   13
-------------------------------------------------------------------------
```

5.1.4 Bias Mitigation via Data Weighting

This section develops a method for producing an unbiased representation without discarding any information from the original data, e.g. no samples are deleted or reassigned. Instead, it compensates for missing drop samples by increasing the contribution of non-missing drop samples from previous cohorts. This method has two key ingredients.

- To compensate for missing *drop* data in a given cohort the weight of non-missing *drop* data from previous cohorts is increased, and
- to compensate for the inflated number of *censored* samples in a given cohort (because of the **drop-to-censor** error) the weight of *censored* samples for that cohort is decreased.

We refer to this as the *weighted data* (WD) method. The intuition and reasoning behind the specifics of the WD method can be developed by focusing on the highlighted data in semester 8 of the staged data table below.

Cohort 2012:	nsamples 3424												
ngrad:	0	0	0	3	6	41	53	644	350	380	96	126	36
ndrop:	260	392	169	191	123	98	56	59	57	50	29	*	*
ncensor:	0	0	0	0	0	0	0	0	0	0	0	36	169
nrisk:	3424	3164	2772	2603	2409	2280	2141	2032	1329	922	492	367	205
Cohort 2013:	nsamples 3518												
ngrad:	0	0	0	0	6	48	106	886	313	340	86	-	-
ndrop:	293	347	187	189	87	102	60	58	48	*	*	-	-
ncensor:	0	0	0	0	0	0	0	0	0	48	314	0	0
nrisk:	3518	3225	2878	2691	2502	2409	2259	2093	1149	788	400	0	0
Cohort 2014:	nsamples 3132												
ngrad:	0	0	0	1	2	64	121	891	220	-	-	-	-
ndrop:	272	278	152	167	105	86	56	*	*	-	-	-	-
ncensor:	0	0	0	0	0	0	0	73	644	0	0	0	0
nrisk:	3132	2860	2582	2430	2262	2155	2005	1828	864	0	0	0	0
Cohort 2015:	nsamples 3327												
ngrad:	0	0	0	4	7	92	143	-	-	-	-	-	-
ndrop:	258	342	204	178	130	*	*	-	-	-	-	-	-
ncensor:	0	0	0	0	0	97	1872	0	0	0	0	0	0
nrisk:	3327	3069	2727	2523	2341	2204	2015	0	0	0	0	0	0
Cohort 2016:	nsamples 3402												
ngrad:	0	0	0	9	13	-	-	-	-	-	-	-	-
ndrop:	290	404	233	*	*	-	-	-	-	-	-	-	-
ncensor:	0	0	0	207	2246	0	0	0	0	0	0	0	0
nrisk:	3402	3112	2708	2475	2259	0	0	0	0	0	0	0	0
Cohort 2017:	nsamples 3219												
ngrad:	0	0	0	-	-	-	-	-	-	-	-	-	-
ndrop:	386	*	*	-	-	-	-	-	-	-	-	-	-
ncensor:	0	437	2396	0	0	0	0	0	0	0	0	0	0
nrisk:	3219	2833	2396	0	0	0	0	0	0	0	0	0	0
Semester	1	2	3	4	5	6	7	8	9	10	11	12	13

The method in Section 5.1.1 can be used to produce the following unbiased estimate of h_{drop} at semester $t = 8$

$$\hat{h}_{drop}(8) = \frac{\mathsf{ndrop}(2012, 8) + \mathsf{ndrop}(2013, 8)}{\mathsf{nrisk}(2012, 8) + \mathsf{nrisk}(2013, 8)} = \frac{117}{4125} \approx .028$$

This estimate can be used to predict that the missing ndrop value for (cohort 2014, semester 8) as follows

$$\mathsf{ndrop}_{impute}(2014, 8) = \hat{h}_{drop}(8) \cdot \mathsf{nrisk}(2014, 8) = (.028)(1828) \approx 51$$

and therefore the true ncensor value for (cohort 2014, semester 8) is estimated to be

$$\mathsf{ncensor}_{true}(2014, 8) = 73 - 51 = 22$$

To generalize this analysis let κ represent the cohort label and let

$\mathsf{K}_m(t) =$ the set of labels for cohorts whose drop values are *m*issing at time t
$\mathsf{K}_u(t) =$ the set of labels for cohorts whose drop values are *u*ncorrupted at time t

Then, in the general case, the missing drop value for every missing drop location $(\kappa, t) : \kappa \in \mathsf{K}_m(t)$

in the table is estimated to be

$$\begin{aligned}
\mathsf{ndrop}_{impute}(\kappa,t) &= \hat{h}_{drop}(t)\cdot \mathsf{nrisk}(\kappa,t) \\
&= \left(\frac{\sum_{\kappa'\in \mathsf{K}_u(t)} \mathsf{ndrop}(\kappa',t)}{\sum_{\kappa'\in \mathsf{K}_u(t)} \mathsf{nrisk}(\kappa',t)}\right)\cdot \mathsf{nrisk}(\kappa,t) \\
&= \left(\frac{\mathsf{nrisk}(\kappa,t)}{\sum_{\kappa'\in \mathsf{K}_u(t)} \mathsf{nrisk}(\kappa',t)}\right)\cdot \sum_{\kappa'\in \mathsf{K}_u(t)} \mathsf{ndrop}(\kappa',t) \\
&= w_{impute}(\kappa,t)\cdot \sum_{\kappa'\in \mathsf{K}_u(t)} \mathsf{ndrop}(\kappa',t)
\end{aligned}$$

where

$$w_{impute}(\kappa,t) = \frac{\mathsf{nrisk}(\kappa,t)}{\sum_{\kappa'\in \mathsf{K}_u(t)} \mathsf{nrisk}(\kappa',t)}$$

is the imputation weight. This result tells us that an unbiased estimate of the missing ndrop value at location (κ,t) can be obtained by applying the weight $w_{impute}(\kappa,t)$ to all (uncorrupted) drop samples in table locations $(\kappa',t):\kappa'\in \mathsf{K}_u(t)$ and then summing these sample weights. Given the imputed ndrop value, an estimate of the true ncensor count at table location $(\kappa,t):\kappa\in \mathsf{K}_m(t)$ is given by

$$\mathsf{ncensor}_{true}(\kappa,t) = \mathsf{ncensor}(\kappa,t) - \mathsf{ndrop}_{impute}(\kappa,t)$$

In the example above, only one cohort contributes missing drop values at any given time t, but in general there can be multiple cohorts with missing drop values at a given time t. For example, if we change the drop estimation parameter to $K=4$ then we obtain the staged data table below which has two cohorts that contribute missing drop values for most of the times. In cases like these, where $|K_m(t)|>1$, the formulas above can be used to compute values for ndrop_{impute}, w_{impute}, and $\mathsf{ncensor}_{true}$ for all table locations $(\kappa,t):\kappa\in \mathsf{K}_m(t)$.

At this point we could form a complete staged data table by replacing the missing ndrop values with imputed values, and replacing the corresponding ncensor values with the "true" values. Then we could apply the (original unmodified) DKM method directly to the completed staged data table to produce unbiased hazard estimates. It turns out that this would give the same result as the method in Section 5.1.1. However, our goal here is to produce a sample data set that is unbiased. So instead of imputing the missing *ndrop* values, we use sample weights to increase the contribution of *drop* samples from the uncorrupted cohorts so that the net effect is the same. In addition, instead of decreasing the *ncensor* counts of the current cohort we use sample weights to decrease the contribution of the *censor* samples from this cohort.

To implement this method we must know the cohort membership for each data sample, so the input data samples now take the form (κ,t,c,b) (instead of (t,c,b)). Our plan is to convert each original data sample (κ_i,t_i,c_i,b_i) into a weighted data sample (t_i,c_i,b_i,w_i) where the results above are used to determine the weight w_i according to the procedure below. Then, hazard functions can be estimated using any procedure that accepts weighted data.

```
-------------------------------------------------------------------------------
UNM FTFT Data (Option NSS,K=4)
Cohort Fall 2012:  nsamples 3424
  ngrad:      0    0    0    4    7   47   76  656  372  365   97   95   26   14
  ndrop:    260  407  193  205  126   95   52   64   48   31    *    *    *    *
  ncensor:    0    1    0    3    7    9   10   17    6   18   14   33   19   28
   nrisk:  3424 3164 2756 2563 2351 2211 2060 1922 1185  759  345  216   87   42
Cohort Fall 2013:  nsamples 3518
  ngrad:      0    0    2    3    9   54  110  892  337  319   80   52    -    -
  ndrop:    293  364  209  197   90   89   56   45    *    *    *    *    -    -
  ncensor:    0    5    2    5    7    8   11   20   30   50   64   82    0    0
   nrisk:  3518 3225 2856 2643 2438 2332 2181 2004 1047  647  278  134    -    -
Cohort Fall 2014:  nsamples 3132
  ngrad:      0    0    0    3    4   67  134  886  227  228    -    -    -    -
  ndrop:    272  293  179  169   98   77    *    *    *    *    -    -    -    -
  ncensor:    0    3    6    8   12   14   27   74   74  232    0    0    0    0
   nrisk:  3132 2860 2564 2379 2199 2085 1927 1726  761  460    -    -    -    -
Cohort Fall 2015:  nsamples 3327
  ngrad:      0    0    0    6   10   92  139  808    -    -    -    -    -    -
  ndrop:    258  353  219  177    *    *    *    *    -    -    -    -    -    -
  ncensor:    0    0    6   17   31  128  143  820    0    0    0    0    0    0
   nrisk:  3327 3069 2716 2491 2291 2136 1910 1628    -    -    -    -    -    -
Cohort Fall 2016:  nsamples 3402
  ngrad:      0    0    0    9   15   43    -    -    -    -    -    -    -    -
  ndrop:    290  412    *    *    *    *    -    -    -    -    -    -    -    -
  ncensor:    0    2   23  237  195 1940    0    0    0    0    0    0    0    0
   nrisk:  3402 3112 2698 2443 2193 1983    -    -    -    -    -    -    -    -

Semester      1    2    3    4    5    6    7    8    9   10   11   12   13   14
-------------------------------------------------------------------------------
```

The procedure for determining the data weights is specified in the three steps below.

- All *grad* samples, i.e. all samples where $(c_i, b_i) = (0, \mathsf{G})$, are assigned a weight $w_i = 1$.

- All *drop* samples, i.e. all samples where $(c_i, b_i) = (0, \mathsf{D})$, are assigned a weight value based on their time location in the staged data table. Specifically, the weight value for all drop samples with $t_i = t$ is

$$w_{drop}(t) = 1.0 + \sum_{\kappa \in \mathsf{K}_m(t)} w_{impute}(\kappa, t)$$

  That is, the initial weight of 1.0 is increased by a value w_{impute} for each cohort with missing drop samples at time t. Note that this weight is the same for all cohorts $\kappa \in \mathsf{K}_u(t)$ with uncorrupted drop samples at time t.

- All *censored* samples, i.e. all samples where $c_i = 1$, are assigned a weight value based on their (cohort, time) location in the staged data table. Specifically, the weight value for censor samples with $(\kappa_i, t_i) = (\kappa, t)$ is

$$w_{censor}(\kappa, t) = \frac{\mathsf{ncensor}(\kappa, t) - \mathsf{ndrop}_{impute}(\kappa, t)}{\mathsf{ncensor}(\kappa, t)}$$

  i.e. we decrease their weight value so that the sum of censored sample weights in location (κ, t) is equal to the estimated number of true censor samples in this location. Note, as a

practical matter in the rare case where this computation produces a negative value we set the weight value to 0.

In summary, the weight values for individual samples $(\kappa_i, t_i, c_i, b_i)$ are determined as follows

$$w_i = \begin{cases} w_{drop}(t), & (t_i, c_i, b_i) = (t, 0, \mathsf{D}) \\ w_{censor}(\kappa, t), & (t_i, c_i) = (t, 1) \text{ and } \kappa_i = \kappa \\ 1.0, & otherwise \end{cases}$$

An example of the drop and censor weights for the $K = 2$ staged data table above is shown below. In the table $\mathsf{dwt} = w_{drop}$ and $\mathsf{cwt} = w_{censor}$.

```
-------------------------------------------------------------------------
UNM FTFT Data (Option NSS,K=2)
Cohort Fall 2012:  nsamples 3424
  ngrad:      0    0    0    3    6   41   53  644  350  380   96  126   36
  ndrop:    260  392  169  191  123   98   56   59   57   50   29    *    *
   dwt:    1.00 1.18 1.18 1.24 1.24 1.32 1.31 1.44 1.35 1.85 1.81    -    -
  ncensor:    0    0    0    0    0    0    0    0    0    0    0   36  169
   cwt:    1.00 1.00 1.00 1.00 1.00 1.00 1.00 1.00 1.00 1.00 1.00 1.00 1.00
Cohort Fall 2013:  nsamples 3518
  ngrad:      0    0    0    0    6   48  106  886  313  340   86    -    -
  ndrop:    293  347  187  189   87  102   60   58   48    *    *    -    -
   dwt:    1.00 1.18 1.18 1.24 1.24 1.32 1.31 1.44 1.35    -    -    -    -
  ncensor:    0    0    0    0    0    0    0    0    0   48  314    0    0
   cwt:    1.00 1.00 1.00 1.00 1.00 1.00 1.00 1.00 1.00 0.11 0.92    -    -
Cohort Fall 2014:  nsamples 3132
  ngrad:      0    0    0    1    2   64  121  891  220    -    -    -    -
  ndrop:    272  278  152  167  105   86   56    *    *    -    -    -    -
   dwt:    1.00 1.18 1.18 1.24 1.24 1.32 1.31    -    -    -    -    -    -
  ncensor:    0    0    0    0    0    0    0   73  644    0    0    0    0
   cwt:    1.00 1.00 1.00 1.00 1.00 1.00 1.00 0.29 0.94    -    -    -    -
Cohort Fall 2015:  nsamples 3327
  ngrad:      0    0    0    4    7   92  143    -    -    -    -    -    -
  ndrop:    258  342  204  178  130    *    *    -    -    -    -    -    -
   dwt:    1.00 1.18 1.18 1.24 1.24    -    -    -    -    -    -    -    -
  ncensor:    0    0    0    0    0   97 1872    0    0    0    0    0    0
   cwt:    1.00 1.00 1.00 1.00 1.00 0.05 0.97    -    -    -    -    -    -
Cohort Fall 2016:  nsamples 3402
  ngrad:      0    0    0    9   13    -    -    -    -    -    -    -    -
  ndrop:    290  404  233    *    *    -    -    -    -    -    -    -    -
   dwt:    1.00 1.18 1.18    -    -    -    -    -    -    -    -    -    -
  ncensor:    0    0    0  207 2246    0    0    0    0    0    0    0    0
   cwt:    1.00 1.00 1.00 0.15 0.95    -    -    -    -    -    -    -    -
Cohort Fall 2017:  nsamples 3219
  ngrad:      0    0    0    -    -    -    -    -    -    -    -    -    -
  ndrop:    386    *    *    -    -    -    -    -    -    -    -    -    -
   dwt:    1.00    -    -    -    -    -    -    -    -    -    -    -    -
  ncensor:    0  437 2396    0    0    0    0    0    0    0    0    0    0
   cwt:    1.00 0.26 0.93    -    -    -    -    -    -    -    -    -    -

Semester      1    2    3    4    5    6    7    8    9   10   11   12   13
-------------------------------------------------------------------------
```

Note that this method compensates for all missing drop values except the last two semester drop

counts in the first cohort 2012 (because there is no previous cohort data to draw from). Note also that the censor sample weights in the next-to-last semester for each cohort are near 0. These censor samples represent students who did not enroll in the most recent semester, but we are unable to assign a drop label until we know their enrollment status for the upcoming semester. Based on the evidence from previous cohorts these students are likely to drop, so reducing the censor sample weight to near 0 makes sense. On the other hand the censor sample weights in the last semester for each cohort are typically close to 1. These censor samples represent students who did enroll in the most recent semester, and are therefore more likely to continue than to drop.

Remark 6. Note that the WD method assumes that all cohort data are generated according to the same distribution which is the same assumption made by virtually all survival analysis methods.

Remark 7. In Section 8 we add covariates to the model so that we can make predictions based on student attributes, i.e. we build models whose predictions depend on the covariate value. In this case the potential variability of the drop distribution across covariate values will have a direct affect on the censor weights w_{censor}, and so these weights must depend on the covariate value. On the other hand, the drop weight w_{drop} relies only on the relative number of samples between cohorts and is unaffected by this variability. Therefore, the drop weight w_{drop} does not depend on the covariate value.

5.1.5 The DKM Method for Weighted Data

The DKM method in Section 4.2 is easily extended to accommodate weighted data by replacing "sample counts" by "sample weight sums" as follows. The new hazard function estimate is given by

$$\hat{h}(t,b) = \frac{\omega(t,0,b)}{\omega_{risk}(t)},$$

where

$$\omega(t,c,b) = \sum_{i:(t_i,c_i,b_i)=(t,c,b)} w_i$$
$$\omega_{risk}(t) = \sum_{i:t_i \geq t} w_i$$

It is easy to see that this reverts to the traditional DKM estimate when the sample weights are all $w_i = 1$. Once the hazard function is determined, the other prediction functions are computed in the same way as before.

5.1.6 Experimental Comparison of Drop Label Delay Bias Mitigation Methods

This section provides an experimental comparison of the drop delay bias mitigation methods described in previous sections. Synthetic data is used so ground truth comparisons can be made. The data set is generated using the synthesis model in **Sample Plan 4** with the following parameters.

- Event and censor times are determined using the NSS option.
- The student enrollment vectors $\mathbf{e}$ contain no stopouts. Therefore the data contains no *censor-to-drop* errors. It contains only *drop-to-censor* errors that produce the bias we're trying to correct. This makes it easier to isolate the efficacy of the bias mitigation methods.
- The maximum data synthesis time is $\mathsf{T}_e = 10$ semesters.
- The data consists of four cohorts that start at absolute times 1, 3, 5, and 7. A total of 1000 samples are synthesized for each of the four cohorts.

- The measurement time is $\mathsf{T} = 8$ semesters. With this choice, all four cohorts will have censored data.
- Two versions of the synthetic data are generated.
  - **ideal:** In this version all grad, drop, and censored samples are labeled correctly as if there was no drop delay issue. This data is used to obtain the *ideal* results below.
  - **realistic:** In this version drop samples are determined using the drop estimation procedure with $K = 2$ semesters as they would be in the real world. The staged data table for this data set is shown below.

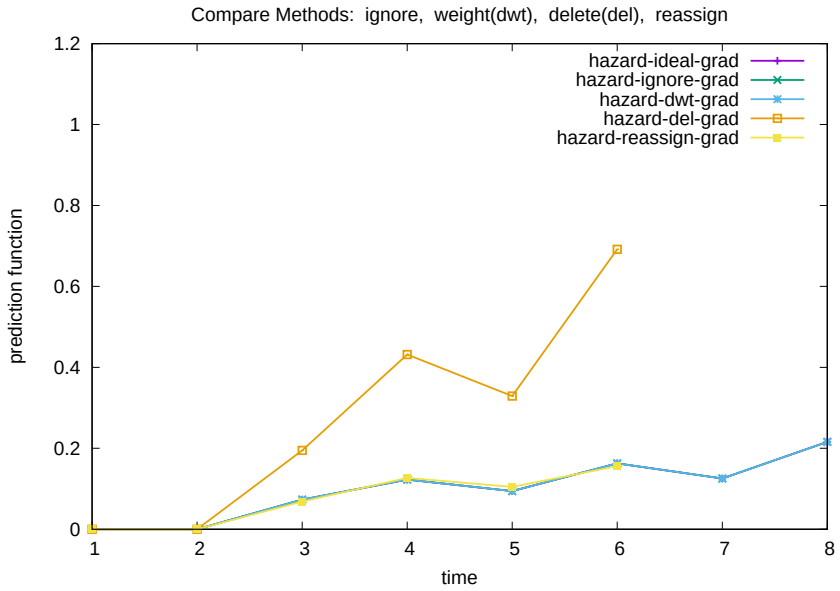
```
--------------------------------------------------
Cohort 1:  start time 1, nsamples  1000
   ngrad:      0    0   65  112   77  101   63   90
   ndrop:     17   30   15   25   11   45   22   46
   ncensor:    0    0    0    0    0    0    0  281
   nrisk:   1000  983  953  873  736  648  502  417
Cohort 2:  start time 3, nsamples  1000
   ngrad:      0    0   65  109   61  110    0    0
   ndrop:     13   42   10   32   19   36    0    0
   ncensor:    0    0    0    0    0  503    0    0
   nrisk:   1000  987  945  870  729  649    0    0
Cohort 3:  start time 5, nsamples  1000
   ngrad:      0    0   79   98    0    0    0    0
   ndrop:     21   27   17   28    0    0    0    0
   ncensor:    0    0    0  730    0    0    0    0
   nrisk:   1000  979  952  856    0    0    0    0
Cohort 4:  start time 7, nsamples  1000
   ngrad:      0    0    0    0    0    0    0    0
   ndrop:     16   30    0    0    0    0    0    0
   ncensor:    0  954    0    0    0    0    0    0
   nrisk:   1000  984    0    0    0    0    0    0

Semester       1    2    3    4    5    6    7    8
--------------------------------------------------
```

We compare the prediction functions produced by the following five methods.

1. **ideal:** The DKM method is applied to the *ideal* data to provide an unbiased hazard estimate that represents represents a "best possible" empirical result.
2. **ignore:** The DKM method is applied to the (unmodified) *realistic* data to show what happens if we simply ignore the drop delay bias.
3. **delete:** The DKM method is applied to the *realistic* data set that has been modified by *deleting* samples as described in Section 5.1.2.
4. **reassign:** The DKM method is applied to the *realistic* data set that has been modified by *reassigning* samples as described in Section 5.1.3.
5. **weight:** The (weighted) DKM method is applied to the *realistic* data set that has been modified by adding sample *weights* as described in Section 5.1.4.

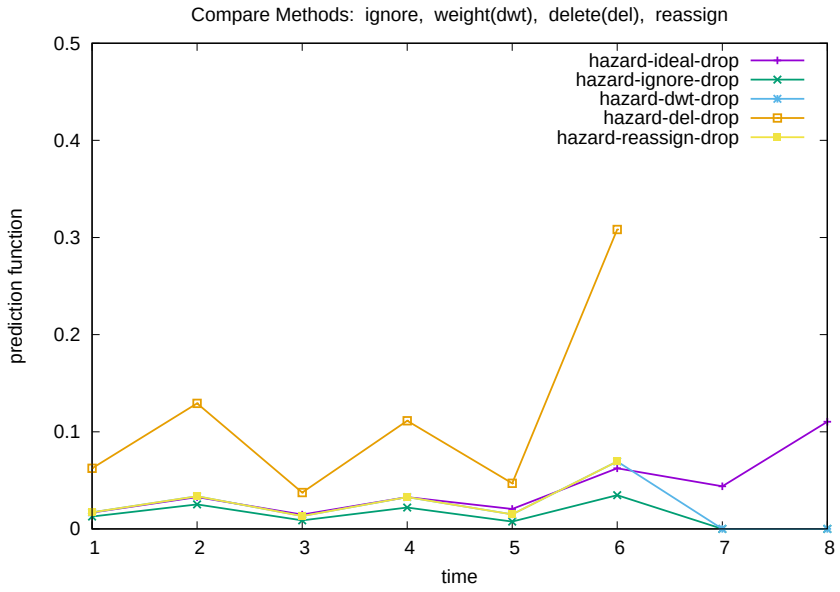
The estimates of h_{grad} are shown in the plot below.



We make the following observations.

- The *ideal*, *ignore*, and *dwt* estimates are identical, i.e. the *ignore* and *dwt* methods exhibit no bias when estimating h_{grad}.
- The *reassign* estimate is almost identical to *ideal* for semesters 1-6, i.e. it produces an unbiased estimate until the last $K = 2$ semesters. No estimate is provided for the last $K = 2$ semesters because this method leaves no data for these semesters.
- The *delete* estimate is biased high for semesters 3-6. This happens because the nrisk table values are deflated due to sample deletion. No estimate is provided for the last $K = 2$ semesters because this method leaves no data for these semesters.

The estimates of h_{drop} are shown in the plot below.



We make the following observations.

- The *ideal*, *dwt*, and *reassign* estimates are nearly identical for semesters 1-6, i.e. the *dwt* and *reassign* methods produce unbiased estimates until the last $K = 2$ semesters. No drop data is provided for the last K semesters so the h_{drop} estimates are 0.
- The *ignore* estimate is biased low because of the missing drop samples in last $K = 2$ semesters.
- The *delete* estimate is biased high for semesters 1-6. This happens because the nrisk table values are deflated due to sample deletion. No estimate is provided for the last $K = 2$ semesters because this method leaves no data for these semesters.

We draw the following conclusions from this experiment.

- The *ignore* and *delete* estimates will produce biased results for real world data.
- The *dwt* and *reassign* methods provide unbiased estimates up to the last K semesters, with the *dwt* method also producing an unbiased estimate of h_{grad} for the last K semesters.
- No method can produce reliable estimates of the last K semesters of h_{drop} due to the complete lack of drop data for this period.

5.2 Censor-to-Drop Error Analysis

Recall that the drop estimation procedure produces a *censor-to-drop* error when a student has failed to enroll over the past K (or more) semesters but decides to enroll again in a future semester. In this case the student is mislabeled as *drop*, but should be labeled as *censor*. This type of error can be reduced by increasing the value of K, but increasing K also increases the information loss. In particular, there is a *complete* loss of information about student drop behavior for the last K semesters, so increasing K increases this loss. Thus, the choice of K involves a trade-off between the amount of error and the amount of information loss. A good practical approach might be to choose the value of K where the *censor-to-drop* error rate curve begins to levels off. This choice will be data dependent, and therefore may vary from one institution to the next. In this section we use the methods described below to estimate the *censor-to-drop* error rate for UNM data.

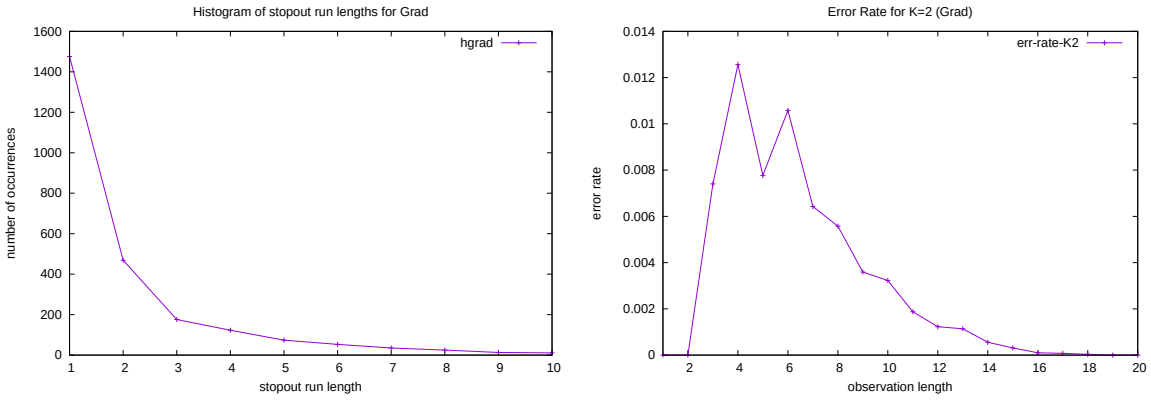
We treat the two types of *censor-to-drop* errors separately. The first type erroneously assigns a *drop* label instead of a *censor* label to some students whose true label is later determined to be *grad*. This type of error is represented by case (c) of Equation (17). The second type erroneously assigns a *drop* label instead of a *censor* label to some students whose true label is later determined to be *drop*. This is an error because the assigned time corresponds to a censor time instead of the true drop time. This type of error is represented by case (f) of Equation (17).

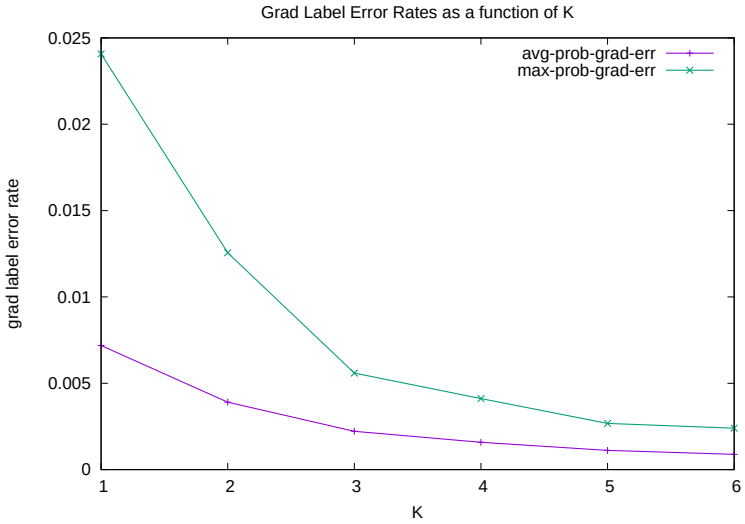
5.2.1 Analysis of Type 1 Errors

To analyze the first *censor-to-drop* error type we extract a group of UNM students that have graduated, i.e. students whose event type label is known with certainty. We analyze the stopout statistics of this group to produce an estimate of the *censor-to-drop* error rate as a function of K. This is accomplished by artificially varying the observation length from 1 to t_e (i.e. from 1 to the *grad* semester) for each student, and tracking *censor-to-drop* errors that would be made by the drop estimation procedure. Accumulating this information over all students allows us to produce a *censor-to-drop* error rate for each semester. Then we retain the average and worst error rates

across the semesters. This entire procedure is repeated for $K = 1, 2, 3, ...$ to provide average and worst case error rates as a function of K.

The data used in this analysis was gathered during the Fall 2020 semester. It consists of graduating students from UNM cohorts from fall 2012 through fall 2016, including the spring cohorts inbetween. All types of students were included, e.g. FTFT, part-time, and transfer students. This group was comprised of 15227 students. A total of 2149 (14 %) experienced at least one stopout semester before graduating. The left plot below shows a histogram of the stopout run lengths for this group. A vast majority of the stopouts are only 1 semester long, so choosing $K \geq 2$ would prevent most of these stopouts from becoming *censor-to-drop* errors. The right plot below shows the *censor-to-drop* error rate for this group as a function of observation length when $K = 2$. The average (nonzero) error rate is 0.4% and the maximum error rate is 1.25%. The next plot below shows the average and maximum error rates as a function of K. These curves suggest that $K = 3$ might be a good choice, and that this choice yields a *censor-to-drop* error rate well below 1%.



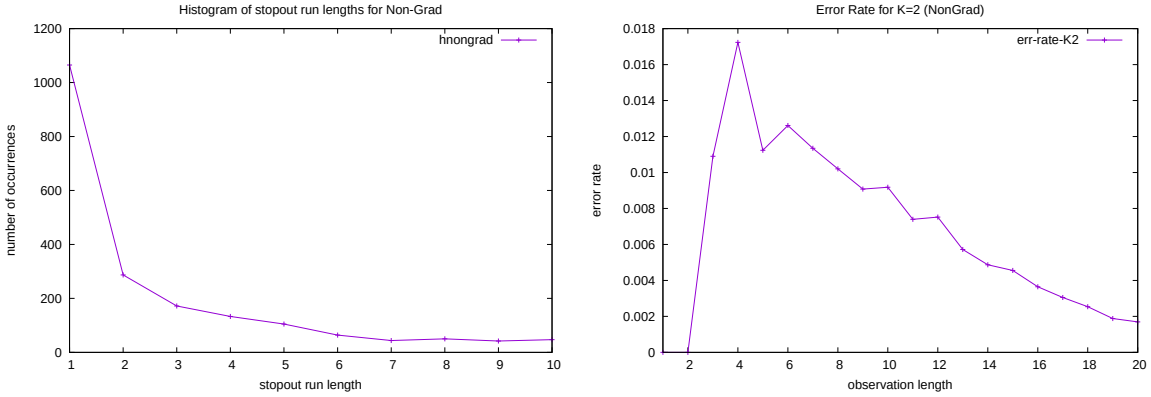


5.2.2 Analysis of Type 2 Errors

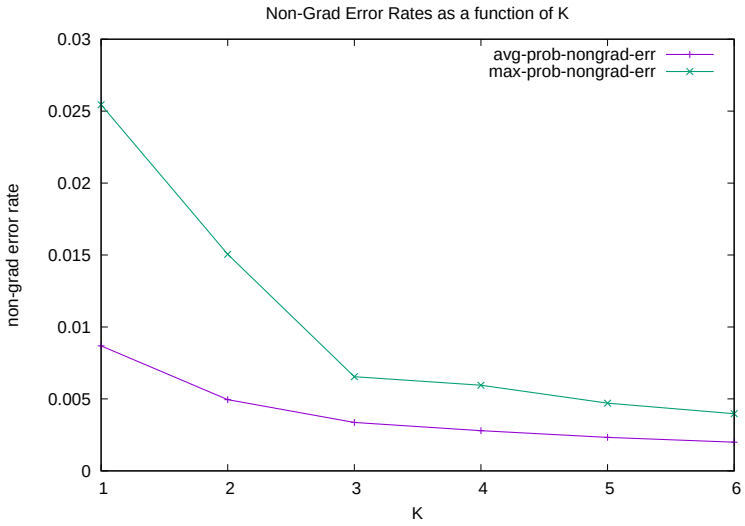
The second type of *censor-to-drop* error occurs when a *drop* label is assigned at time t_d to a student whose true event type is *drop*, but whose true event time is greater than t_d. To analyze this second error type we extract a group of UNM students who started over ten years ago, but still

have not graduated[9]. Because of the extended observation length the true event type for a vast majority of these students is likely to be *drop* (with most of them having dropped prior to the data collection time). Thus, the *censor-to-drop* errors in this group could be viewed as correctly labeled samples with erroneous (premature) event times, instead of incorrectly labeled samples with correct censored time values.

The data used in this analysis was gathered during the Fall 2020 semester. It consists of students from UNM cohorts fall 2008, spring 2009, and fall 2009 that have not graduated by the Fall 2020 semester. All types of students were included, e.g. FTFT, part-time, and transfer students. This group was comprised of 7343 students. A total of 1676 (23 %) experienced at least one stopout semester. The same stopout analysis described in the previous section was used to obtain a *censor-to-drop* error rate for each semester, and to produce average and worst error rates as a function of K. The left plot below shows a histogram of the stopout run lengths for this group. It is very similar to the histogram for graduating students in the previous section. A vast majority of the stopouts are only 1 semester long, so choosing $K \geq 2$ would prevent most of these stopouts from becoming *censor-to-drop* errors. The right plot below shows the *censor-to-drop* error rate for this group as a function of observation length when $K = 2$. This plot is also similar to the plot for graduating students in the previous section except that it has larger error rates at the longer observation lengths, which simply reflects the fact that students who drop experience more stopouts in later semesters than students who graduate. The average (nonzero) error rate is 0.6% and the maximum error rate is 1.72%. The next plot below shows the average and maximum error rates as a function of K. Once again these curves suggest that $K = 3$ might be a good choice, and that this choice yields a *censor-to-drop* error rate well below 1%.



[9]This is the complement set to the students in the previous section.



5.2.3 Summary of Censor-to-Drop Errors

Choosing $K = 2$ or 3 provides a good trade-off between the amount of censor-to-drop error and the amount of information loss for UNM data. Both choices yield an expected censor-to-drop error rate well below 1%. Approximately half of these errors (i.e. the type 2 errors) can be viewed as event time errors instead of label errors. Indeed, all *censor-to-drop* errors assign a premature event time that will bias inferences towards slightly lower event times. But this bias will be small because of the extremely low error rates. Finally, censor-to-drop errors are most likely to occur during observation lengths 3-6, suggesting (slightly) more accurate results when the data consists entirely of student cohorts that started more than three years before the data was gathered.

5.3 Grad Label Delay

Sections 5.1 and 5.2 described a *delay* in the determination of the *drop* label relative to the *grad* label. In this section we describe a delay in the opposite direction, i.e. a delay in the determination of the *grad* label relative to the *drop* label. This delay stems from the fact that a student's enrollment status for a given semester is known well before a student's graduation status for that same semester.

The existence of this delay depends on the measurement time, and one of the most likely measurement times is right after the first six weeks of the semester when the enrollment data for the current semester is known, and the graduation data for the previous semester is known. In this case there is a one semester difference between the *latest semester for which graduation results are valid* and the *latest semester for which enrollment results are valid*. Thus, relative to the current semester there is a one semester delay in determining the *grad* label, and a K semester delay in determining the *drop* label. So the only label available for students enrolled in the current semester is *censor*. But, as we have discussed previously, labeling current semester times as *censor* when some of these samples should be labeled *drop* or *grad* results in a biased representation.

This issue is resolved as follows. The current semester enrollment data is used to determine *drop* labels and the previous semester graduation data is used to determine the *grad* labels. Then all samples are processed as if the data was gather during the previous semester, i.e. measurement time = previous semester, and the effective value of K is reduced by 1. In particular, observation lengths L for all cohorts are based on the number of semesters up to and including the previous

semester. This means that there will be no samples with event times that correspond to the current semester. An example of UNM FTFT cohorts that have been processed in this way is shown in the a staged data table below. This data was collected during the Spring 2019 semester, but the last semester in the table now corresponds to the previous semester, i.e. Fall 2018.

```
-------------------------------------------------------------------------------
UNM FTFT Data (Option NSS,K=2)
Cohort Fall 2012:  nsamples 3424
  ngrad:      0    0    0    3    6   41   53  644  351  379   96  126   36
  ndrop:    260  392  169  191  123   98   56   59   57   50   29   36    *
  ncensor:    0    0    0    0    0    0    0    0    0    0    0    0  169
Cohort Fall 2013:  nsamples 3518
  ngrad:      0    0    0    0    6   48  106  886  313  340   86    -    -
  ndrop:    293  347  187  189   87  102   60   58   48   48    *    -    -
  ncensor:    0    0    0    0    0    0    0    0    0    0  314    -    -
Cohort Fall 2014:  nsamples 3132
  ngrad:      0    0    0    1    2   64  121  891  220    -    -    -    -
  ndrop:    272  278  152  167  105   86   56   73    *    -    -    -    -
  ncensor:    0    0    0    0    0    0    0    0  644    -    -    -    -
Cohort Fall 2015:  nsamples 3327
  ngrad:      0    0    0    4    7   92  143    -    -    -    -    -    -
  ndrop:    258  342  204  178  130   97    *    -    -    -    -    -    -
  ncensor:    0    0    0    0    0    0 1872    -    -    -    -    -    -
Cohort Fall 2016:  nsamples 3402
  ngrad:      0    0    0    9   13    -    -    -    -    -    -    -    -
  ndrop:    290  404  233  207    *    -    -    -    -    -    -    -    -
  ncensor:    0    0    0    0 2246    -    -    -    -    -    -    -    -
Cohort Fall 2017:  nsamples 3219
  ngrad:      0    0    0    -    -    -    -    -    -    -    -    -    -
  ndrop:    386  437    *    -    -    -    -    -    -    -    -    -    -
  ncensor:    0    0 2396    -    -    -    -    -    -    -    -    -    -

Semester      1    2    3    4    5    6    7    8    9   10   11   12   13
-------------------------------------------------------------------------------
```

6 TTG Version 4 (modifying the start and end times)

In this version of the TTG problem we add the option to change the prediction function *start* and *end* times. First we consider changing the *start* time.

Up to now the prediction functions start at the very first semester $t = 1$. These functions are useful for making predictions about students who are just now starting their degree program. But the time-to-event statistics will be different for students who have already completed $\tau_s > 1$ semesters, and we would like to estimate prediction functions for these students as well, i.e. we would like to estimate prediction functions that start at a time $\tau_s > 1$. The new prediction functions are characterized by the distribution $P_{T_e,T_c,B|T_e\geq\tau_s}$ (instead of $P_{T_e,T_c,B}$), and the independent censoring assumption now implies that

$$P_{T_e,T_c,B|T_e\geq\tau_s} = P_{T_e,B|T_e\geq\tau_s} P_{T_c}. \qquad \text{(independent censoring)}$$

The new hazard function is defined

$$h(t,b) = \frac{P_{T_e,B|T_e\geq\tau_s}(t,b)}{P_{T_e|T_e\geq\tau_s}(T_e \geq t)}, \qquad t \geq \tau_s$$

Note that this function is only defined for times $t \geq \tau_s$. Now since

$$P_{T_e,B|T_e\geq\tau_s}(t,b) = \frac{P_{T_e,B}(t,b)}{P_{T_e}(T_e \geq \tau_s)} \quad \text{and} \quad P_{T_e|T_e\geq\tau_s}(T_e \geq t) = \frac{P_{T_e}(T_e \geq t)}{P_{T_e}(T_e \geq \tau_s)}$$

the hazard becomes

$$h(t,b) = \frac{P_{T_e,B}(t,b)/P_{T_e}(T_e \geq \tau_s)}{P_{T_e}(T_e \geq t)/P_{T_e}(T_e \geq \tau_s)} = \frac{P_{T_e,B}(t,b)}{P_{T_e}(T_e \geq t)} \qquad t \geq \tau_s$$

which is the same as before, except that it is only defined for $t \geq \tau_s$. This means that we can use the exact same formula in (11) to produce a DKM estimate this "new" hazard function. Once we have this hazard function, the probability function can be computed as follows,

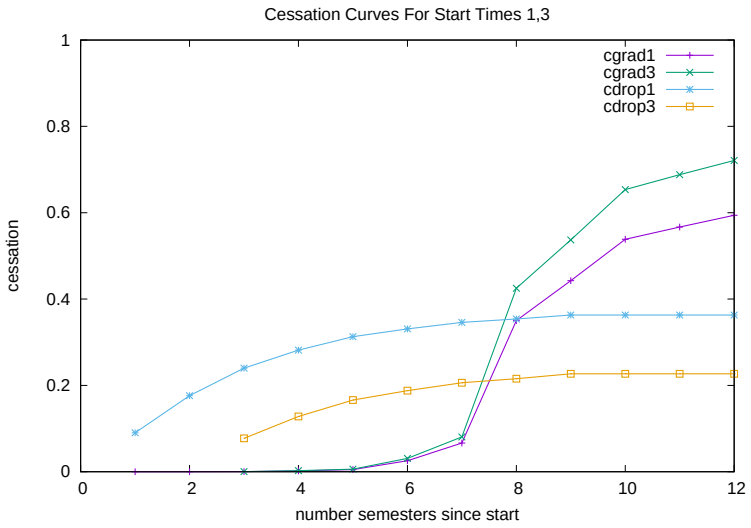
$$P_{T_e,B|T_e\geq\tau_s}(t,b) = \begin{cases} h(t,b), & t = \tau_s \\ h(t,b)\prod_{t'=1}^{t-1}\left(1-h(t')\right), & t > \tau_s \end{cases}$$

where $h(t') = h(t,\mathsf{G}) + h(t,\mathsf{D})$. Also, the cessation and survival functions can be computed using

$$C(t,b) = \sum_{t'=\tau_s}^{t} P_{T_e,B|T_e\geq\tau_s}(t',b) \qquad t \geq \tau_s$$

$$S(t,b) = \sum_{t'=t}^{\mathtt{T}_e} P_{T_e,B|T_e\geq\tau_s}(t',b) \qquad t \geq \tau_s$$

We illustrate this method with the following example. The data for this example consists of UNM FTFT students from the fall cohorts 2013 - 2017. This data was collected in the summer of 2019, so that ground truth labels for both *grad* and *drop* students are known through the end of the Spring 2019 semester. The plot below compares the cessation functions for the original group of students that are active in semester 1, to the subset of students that are still active in semester 3, i.e. the subset that make it past their first year. This result shows a substantial increase in graduation rate, and corresponding decrease in drop rate, for the students who make it past the first year.



Now we consider changing the *end* time. In practice the prediction functions are typically estimated out to the largest time in the current data set, but this time may vary from one data set to the next, and so it might be useful to allow the user to specify a common end time τ_e that is less than the largest time in a data set. In addition, the largest time slots tend to have very few samples, so the prediction function estimates are less accurate for these times. For this reason it might be useful to specify an end time τ_e that guarantees a sufficient number of at-risk samples for all time slots $t \leq \tau_e$.

Implementation of this option is straightforward and can be accomplished by decreasing the effective observation length for selected cohorts, similar to bias mitigation method described in Section 5.1.3. For the NSS option this is accomplished by converting all data samples with event or censor time $t_i > \tau_e$ to censored samples with time $t_i = \tau_e$, and then proceeding with the standard estimation method(s). For the NSE option, all data samples with event or censor time $t_i > \tau_e$ are converted to censored samples whose censor times are determined by counting the number of enrolled semesters that would be observed with an earlier measurement time that is chosen so that all adjusted data times satisfy $t_i \leq \tau_e$. This typically means retreating the measurement time by $(\max\{t_i\} - \tau_e)$ semesters.

7 TTG Version 5 (extending the prediction functions)

Truncating the end time (as described in the previous section) produces an *incomplete* probability model where

$$\sum_{t=1}^{\tau_e} \sum_{b \in \mathcal{B}} \hat{P}_{T_e,B}(t,b) < 1$$

Even when the end time is not truncated, most modeling methods (including DKM and the multinomial logistic regression model) produce an *incomplete* probability model. There are several reasons for this.

- First recall from Remark 3 that the true end time T_e for the distribution $P_{T_e,B}$ may never actually be known. Although there is a practical limit on the number of semesters that a student can remain in a program, this limit is not always clearly defined. Furthermore it is often difficult to infer a hard limit from empirical data because there always seems to be a few students that remain in the system longer than the maximum time allotted by the monitoring system.

- In practice, when the end time is not truncated, prediction functions are typically estimated out to the largest time in the current data set. But most data sets contain censored samples in the last time slot, which indicates that T_e is larger than the largest time in the data set. Most modeling methods, including DKM, will produce an *incomplete* probability model in this case.

- Even when the data contains no censored samples in the last time slot, many popular survival methods will produce incomplete probability estimates simply because of the model structure. To see this, first note that the aggregate hazard definition in (12) can be expressed as

  $$h(t) = \frac{P_{T_e}(t)}{P_{T_e}(T_e \geq t)}$$

  and this implies that

  $$h(\mathsf{T}_e) = 1.$$

Thus, if T is the largest (finite) time for which a model is defined and if $h(\mathsf{T}) < 1$ then the corresponding probability model will be incomplete. Consider one of the most popular hazard models for multiple event types, the multinomial logistic regression model below (Scott and Kennedy [23])[10]

$$h_{\mathbf{x}}(t,b) = \frac{\exp\left[(\alpha_{t,b} + \boldsymbol{\beta}_b \cdot \mathbf{x})\right]}{1 + \sum_{b\in\mathcal{B}} \exp\left[(\alpha_{t,b} + \boldsymbol{\beta}_b \cdot \mathbf{x})\right]} \tag{19}$$

where $\mathbf{x}$ is a real-valued covariate vector[11]. The corresponding aggregate hazard function takes the form

$$h_{\mathbf{x}}(t) = \sum_{b\in\mathcal{B}} h_{\mathbf{x}}(t,b) = \frac{\sum_{b\in\mathcal{B}} \exp\left[(\alpha_{t,b} + \boldsymbol{\beta}_b \cdot \mathbf{x})\right]}{1 + \sum_{b\in\mathcal{B}} \exp\left[(\alpha_{t,b} + \boldsymbol{\beta}_b \cdot \mathbf{x})\right]}$$

and as long as the model parameters $\{\alpha_{t,b}, \boldsymbol{\beta}_b\}$ are finite this hazard function will satisfy $h_{\mathbf{x}}(\mathsf{T}) < 1$, and the corresponding probability model will be incomplete.

Incomplete function estimates are not necessarily inaccurate at their estimated times, they simply provide no estimate for the extended time slots beyond the last time T. In fact, incomplete function estimates may be acceptable in some cases. For example, if our main goal is to estimate the 4-year and 6-year graduation rates then estimates beyond $t = 12$ may not be important. On the other hand, if we want to make predictions about future enrollments (as described in Section 9) then an incomplete survival function estimate will produce biased enrollment predictions, which is undesirable.

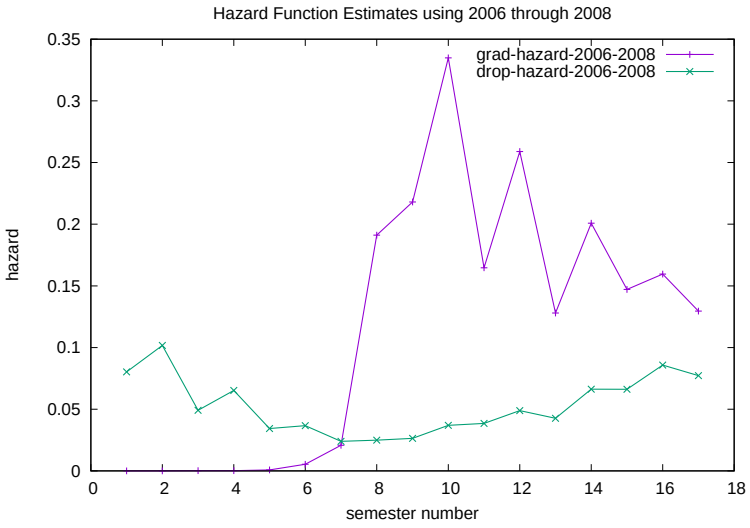
Our goal in this section is to *extend* the prediction function estimates beyond the previously estimated times so that they form a *complete* probability model that satisfies

$$\sum_{t=1}^{\hat{\mathsf{T}}_e} \sum_{b\in\mathcal{B}} \hat{P}_{T_e,B}(t,b) \approx 1$$

where $\hat{\mathsf{T}}_e$ is the estimated value of T_e. Our approach is to use a simple model that exploits the characteristics of a typical TTG hazard function to extrapolate the tail values. Consider the group of FTFT students with entry semesters Fall 2006, Fall 2007, and Fall 2008 whose status was collected more than 11 years later in the summer of 2019 so that the data contains no missing drop or censored samples. We will refer to the hazard values after semester 10 (year 5) as the *tail* values. Over half of the students have either graduated or dropped by semester 10.

[10] In the single event case this model reduces to the logistic regression model first proposed in (Cox [5]) and developed extensively in (Singer and Willett [25]).

[11] The incorporation of covariates is described in more detail in Section 8.



We seek to exploit the following characteristics of these functions.

- *Tail Trends:* The tail of h_{grad} trends *downward* and the tail of h_{drop} trends *upward*. This suggests that after 10 semesters students become less and less likely to graduate, and more and more likely to drop, even though at any given semester they are more likely to graduate than drop, at least up to semester 17.

- *Sawtooth Pattern:* The tail of both hazard functions exhibits a sawtooth pattern with sawtooth peaks at *even numbered semesters*. This suggests that students are more likely to graduate or drop in the spring semester of a given year rather than the fall semester. This effect is more pronounced in the h_{grad} function.

- *Tooth Size:* In the tail region, the *tooth size* of the h_{grad} sawtooth *decreases* over time, while the *tooth size* of the h_{drop} sawtooth increases over time.

We propose two TTG hazard extrapolation models which extrapolate the first part of the tail to the extended part of the tail. Let $\mathcal{T}_{first}$ be the time values associated with the first part of the tail where the hazard functions have already been reliably estimated, and let $\mathcal{T}_{ext}$ be the time values associated with the extended part of the tail. For example $\mathcal{T}_{first} = \{10, 11, 12, 13\}$ and $\mathcal{T}_{ext} = \{14, 15, 16, ..., \hat{\mathsf{T}}_e\}$. Also, let $\mathcal{T}^{odd}$ and $\mathcal{T}^{even}$ be the set of odd and even values from $\mathcal{T}$ respectively.

1. **Linear:** This approach fits the trends in the first part of the tail with a *linear* model, and uses a *proportional offset* parameter to create the teeth. Specifically, the tail estimates at even times are computed starting at $t = \min\{t' : t' \in \mathcal{T}_{ext}^{even}\}$ as follows,

$$h(t, b) = [\ h(t-2, b) + \delta_b\ ]_{0:1}\,, \qquad t \in \mathcal{T}_{ext}^{even}$$

where $[\ \cdot\ ]_{0:1}$ is a clipping function

$$[\ \alpha\ ]_{0:1} = \begin{cases} 0, & \alpha < 0 \\ \alpha, & 0 \le \alpha \le 1 \\ 1, & \alpha > 1 \end{cases}$$

that restricts the estimates to the range $[0, 1]$. Once the even-time tail estimates have been computed the odd-time tail estimates are computed starting at $t = \min\{t' : t' \in \mathcal{T}_{ext}^{odd}\}$ as follows,

$$h(t, b) = [\ \rho_b h(t-1, b)\ ]_{0:1}\,, \qquad t \in \mathcal{T}_{ext}^{odd}$$

The parameters δ_b and ρ_b are estimated as follows

$$\delta_b = \frac{1}{(|\mathcal{T}_{first}| - 2)} \sum_{(t-2)\in\mathcal{T}_{first}} (h(t, b) - h(t-2, b))$$

$$\rho_b = \frac{1}{|\mathcal{T}_{first}^{odd}|} \sum_{t\in\mathcal{T}_{first}^{odd}} \frac{h(t, b)}{h(t-1, b)}$$

Note that the linear slope δ_b is estimated from both the even time hazard trend and the odd time hazard trend in the first part of the tail, and the proportional offset parameter ρ_b is estimated from the adjacent "odd time over even time" hazard values in the first part of the tail.

2. **Exponential:** This approach models the two transitions from *even-to-odd* and *odd-to-even* semesters as proportional changes. Specifically, the tail estimates are computed as follows.

$t \leftarrow \min\{t' : t' \in \mathcal{T}_{ext}\}$
while $(t \in \mathcal{T}_{ext})$ **do**
  $h(t, b) \leftarrow [\ \rho_{b,1} h(t-1, b)\ ]_{0:1}$
  if $((t+1) \in \mathcal{T}_{ext})$ **then**
    $h(t+1, b) \leftarrow [\ \rho_{b,2} h(t, b)\ ]_{0:1}$
  end if
  $t \leftarrow t + 2$
end while

The proportionality parameters $\rho_{b,1}$ and $\rho_{b,2}$ are estimated using the hazard values in the first part of the tail as follows. Let $t^* = \max\{t' : t' \in \mathcal{T}_{first}\}$ be the largest time in $\mathcal{T}_{first}$, and let

$$\mathcal{T} = \{t^*, t^* - 2, t^* - 4, ...\} \cap \mathcal{T}_{first}$$

be every other time in $\mathcal{T}_{first}$ starting with t^* and counting backwards. Then

$$\rho_{b,1} = \frac{1}{n_1} \sum_{(t-2)\in\mathcal{T}} \frac{h(t-1, b)}{h(t-2, b)}, \qquad \text{where } n_1 = \left\lfloor \frac{|\mathcal{T}_{first}| - 1}{2} \right\rfloor$$

$$\rho_{b,2} = \frac{1}{n_2} \sum_{(t-1)\in\mathcal{T}} \frac{h(t, b)}{h(t-1, b)}, \qquad \text{where } n_2 = \left\lfloor \frac{|\mathcal{T}_{first}|}{2} \right\rfloor$$

With this method the estimate at any extended time t can be expressed as

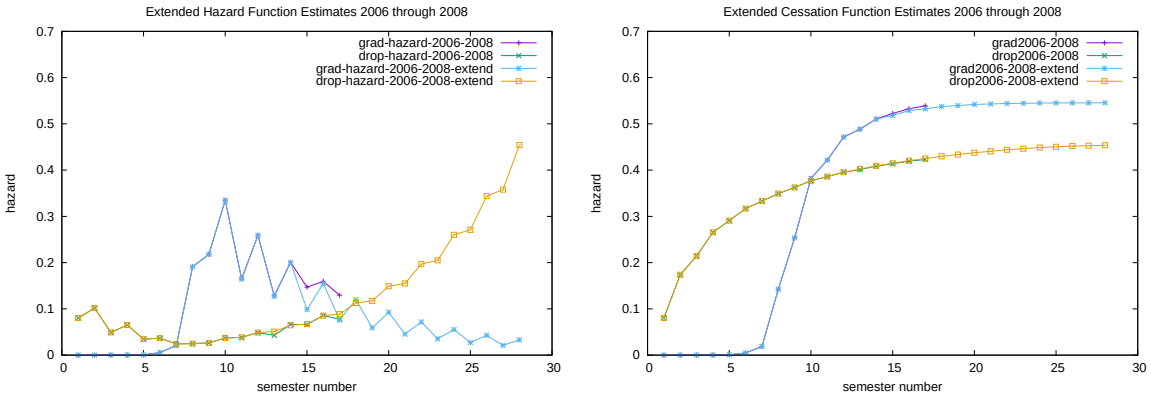
$$h(t, b) = \rho_{b,1}\rho_{b,2} h(t-2, b) = \rho_b h(t-2, b).$$

Thus if $\rho_b > 1$ then h is exponentially increasing and if $\rho_b < 1$ then h is exponentially decreasing.

The size of the extension, i.e. the estimated value of $\hat{\mathsf{T}}_e$, is determined by extrapolating the hazard tail values to larger and larger times until the corresponding probability function satisfies $\sum_{t=1}^{\hat{\mathsf{T}}_e} \sum_{b\in\mathcal{B}} \hat{P}_{T_e,B}(t,b) \approx 1$. The details are shown in the algorithm below.

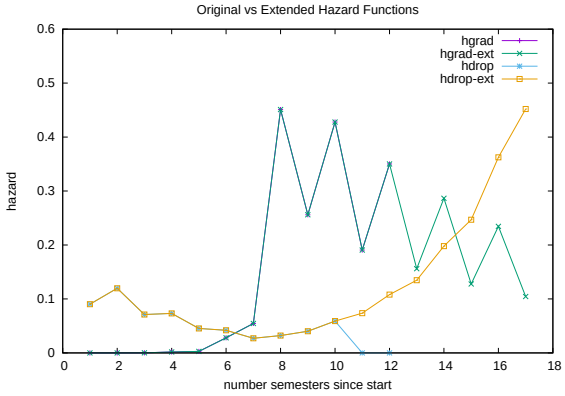
$t \leftarrow \max\{t' : t' \in \mathcal{T}_{first}\}$
$\hat{\mathsf{T}}_e \leftarrow t+1$
$\mathcal{T}_{ext} = \{\hat{\mathsf{T}}_e\}$ {(initialize the extended time set)}
$\hat{h} \leftarrow$ hazard estimate extended to the times in $\mathcal{T}_{ext}$ using linear or exponential method
$\hat{P}_{T_e,B} \leftarrow$ probability function estimate using (13)
$Sum = \sum_{t=1}^{\hat{\mathsf{T}}_e} \sum_{b\in\mathcal{B}} \hat{P}_{T_e,B}(t,b)$
while $(Sum < 1)$ **do**
  $\hat{\mathsf{T}}_e \leftarrow \hat{\mathsf{T}}_e + 1$
  $\mathcal{T}_{ext} \leftarrow \mathcal{T}_{ext} \cup \{\hat{\mathsf{T}}_e\}$ {(extend by one)}
  $\hat{h} \leftarrow$ hazard estimate extended to the times in $\mathcal{T}_{ext}$
  $\hat{P}_{T_e,B} \leftarrow$ probability function estimate using (13)
  $Sum = \sum_{t=1}^{\hat{\mathsf{T}}_e} \sum_{b\in\mathcal{B}} \hat{P}_{T_e,B}(t,b)$
end while
Return$(\hat{h}, \hat{\mathsf{T}}_e)$

Consider the following example where we compare extended hazard values with the DKM hazard estimates from UNM FTFT 2006-2008 data shown above. Specifically, we estimate the parameters for the *exponential* model using the DKM hazard estimates at $\mathcal{T}_{first} = \{10, 11, 12\}$. Then we use these parameters to estimate the extended tail values, including extended estimates for $t = 13-17$ which can be compared with the original DKM estimates. The results for the hazard estimates, and the corresponding cessation estimates, are shown in the plots below.



Note that the extended h_{grad} estimates at $t = 13, 14, 16$ are nearly identical to the previous DKM estimates. But the extended h_{grad} estimates at $t = 15, 17$ are different from the previous DKM estimates because of the slight change in the behavior of the odd time DKM estimates. The extended h_{drop} values are very similar to the previous DKM estimates at all times $t = 13-17$. The extended hazard estimates beyond $t = 17$ are needed to form a *complete* model, and their behavior is consistent with expectations. In addition, the extended cessation functions at $t = 13-17$ are nearly identical to the previous DKM estimates, and beyond $t = 17$ their behavior is consistent with expectations.

In the next example we extend a hazard function whose largest time value is limited because the data is so recent. The data consists of UNM FTFT fall cohorts for 2013-2017 where only 12 semesters of data are available. In addition drop labels are estimated using $K = 2$ semesters. Thus, the DKM method is used to produce a hazard estimate of h_{grad} for t =1-12 and h_{drop} for t =1-10. Parameters for the *exponential* models are estimated using the DKM hazard estimates at $\mathcal{T}_{first} = \{8, 9, 10, 11, 12\}$. Parameters for the h_{drop} extension use only h_{drop} values at times 8-10 (since the values at t =11-12 are missing). The plot below shows the DKM hazard estimates and their extensions. Note that the first extension of h_{grad} is at $t = 13$ and the first extension of h_{drop} is at $t = 11$. This example shows how the extension method can be used to compensate for the missing drop values, as well extend the hazard values beyond $t = 12$. Both extensions are consistent with expectations.



8 TTG Version 6 (adding covariates)

In this section we add covariates to the model so that the time-to-event predictions are based on student attributes such as gender, ethnicity, HS GPA, and financial aid. Student attributes will be represented by a d-tuple $\mathbf{x} = (x_1, x_2, ..., x_d)$ where each component x_i represents a different student attribute. In practice most of these attributes are *discrete* and *finite* valued, e.g. gender, residency, ethnicity, and chosen major. However some attributes, such as HS GPA, are *continuous* valued. Nevertheless, the *continuous* valued attributes can often be quantized into a finite number of discrete bins without losing their dominant influence on the timing of events. Thus, the inference methods developed in this section are designed exclusively for *discrete* and *finite* valued covariates.

Prior to censoring, the ground truth data associated with this problem is a collection of samples, one for each student, that take the form of a (covariate, event time, event type) tuple $(\mathbf{x}, t_e, b)$ which we model as a specific realization of the random variable (X, T_e, B). Thus, the probability distribution that defines the covariate-dependent prediction functions is now $P_{X,T_e,B}$ (instead of $P_{T_e,B}$). In the unified staged censor model the initial probability distribution that defines the covariate-dependent prediction functions is $P_{X,\mathbf{E},B}$, where $\mathbf{E}$ is the the enrollment vector random variable. Application of the staged sensor mechanism to samples $(\mathbf{x}, \mathbf{e}, b)$ from $P_{X,\mathbf{E},B}$ produces samples $(\mathbf{x}, t_e, b)$ distributed according to a distribution $P_{X,T_e,B}$. The joint distribution can be written

$$P_{X,T_e,B} = P_{T_e,B|X} P_X$$

and we will use the short-hand notation

$$P_{T_e,B|\mathbf{x}} = P_{T_e,B|X=\mathbf{x}}$$

for the distribution given a specific covariate value $\mathbf{x}$.

In the survival analysis literature it is conventional to define *conditional* prediction functions based on the *conditional* distribution $P_{T_e,B|X}$ instead of the joint distribution $P_{X,T_e,B}$. This means that there is little interest in modeling the the contribution or influence of P_X, and it implies that the prediction functions treat all values of $\mathbf{x}$ equally, regardless of their likelihood. We will adopt this convention in most of our work, although some of the methods described later will produce estimates of P_X in addition to $P_{T_e,B|X}$. Also, the data synthesis model will require a mechanism for generating the covariates and we will incorporate a distribution P_X for this purpose.

The covariate dependent prediction functions are defined as follows.

- **Probability:** The core probability function is $P_{T_e,B|X}$.
- **Hazard:**

$$\begin{aligned} h_{grad,\mathbf{x}}(t) = h_{\mathbf{x}}(t,\mathsf{G}) &= \frac{P_{T_e,B|\mathbf{x}}(t,\mathsf{G})}{\sum_{t'=t}^{\mathtt{T}_e}\left(P_{T_e,B|\mathbf{x}}(t',\mathsf{G}) + P_{T_e,B|\mathbf{x}}(t',\mathsf{D})\right)} \\ h_{drop,\mathbf{x}}(t) = h_{\mathbf{x}}(t,\mathsf{D}) &= \frac{P_{T_e,B|\mathbf{x}}(t,0)}{\sum_{t'=t}^{\mathtt{T}_e}\left(P_{T_e,B|\mathbf{x}}(t',\mathsf{G}) + P_{T_e,B|\mathbf{x}}(t',\mathsf{D})\right)} \end{aligned} \tag{20}$$

- **Cessation:**

$$\begin{aligned} C_{grad,\mathbf{x}}(t) = C_{\mathbf{x}}(t,\mathsf{G}) &= \sum_{t'=1}^{t} P_{T_e,B|\mathbf{x}}(t',\mathsf{G}) \\ C_{drop,\mathbf{x}}(t) = C_{\mathbf{x}}(t,\mathsf{D}) &= \sum_{t'=1}^{t} P_{T_e,B|\mathbf{x}}(t',\mathsf{D}) \end{aligned} \tag{21}$$

- **Survival:**

$$\begin{aligned} S_{grad,\mathbf{x}}(t) = S_{\mathbf{x}}(t,\mathsf{G}) &= \sum_{t'=t}^{\mathtt{T}_e} P_{T_e,B|\mathbf{x}}(t',\mathsf{G}) \\ S_{drop,\mathbf{x}}(t) = S_{\mathbf{x}}(t,\mathsf{D}) &= \sum_{t'=t}^{\mathtt{T}_e} P_{T_e,B|\mathbf{x}}(t',\mathsf{D}) \end{aligned} \tag{22}$$

The probability function can be expressed in terms of the hazard as follows (similar to (13))

$$P_{T_e,B|\mathbf{x}}(t,b) = \begin{cases} h_{\mathbf{x}}(t,b), & t = 1 \\ h_{\mathbf{x}}(t,b)\prod_{t'=1}^{t-1}\left(1 - h_{\mathbf{x}}(t')\right), & t > 1 \end{cases} \tag{23}$$

where

$$h_{\mathbf{x}}(t) = h_{\mathbf{x}}(t,\mathsf{G}) + h_{\mathbf{x}}(t,\mathsf{D}) \tag{24}$$

We can view our inference task as follows: for each distinct value of $\mathbf{x}$ we seek to estimate the same functions as before.

The covariate dependent censoring and synthesis models take the following form.

- **Random Censoring:** In the case of random censoring the joint probability distribution that defines the data synthesis process is $P_{X,T_e,T_c,B}$ which, under the independent censoring assumption satisfies

$$P_{X,T_e,T_c,B} = P_{X,T_e,B} P_{T_c}$$

Unobserved (ground truth) samples of the form $(\mathbf{x}, t_e, t_c, b_e)$ are generated as follows:

- generate t_c according to P_{T_c}
- generate $\mathbf{x}$ according to P_X
- generate (t_e, b_e) according to $P_{T_e,B|\mathbf{x}}$

Then the observed samples are obtained using

$$(\mathbf{x}, t, c, b) = \begin{cases} (\mathbf{x}, t_e, 0, b_e), & t_e \leq t_c \\ (\mathbf{x}, t_c, 1, b_e), & t_e > t_c \end{cases}$$

- **Staged Censoring:** In the case of staged censoring, samples are first generated according the joint probability distribution $P_{X,\mathbf{E},B}$ and then filtered by the staged censoring mechanism (same as before). The "ideal" *covariate-dependent staged censoring model* that generates samples with error free values of (t, c, b) is shown in **Sample Plan 5** below. Similarly, the *covariate-dependent staged censoring with dropout estimation model* is shown in **Sample Plan 6** below.

Sample Plan 5 (covariate-dependent staged censoring):
- Let $\mathcal{K}$ = number student cohorts, and let $\kappa_1 \leq \kappa_2 \leq ... \leq \kappa_{\mathcal{K}}$ be the ordered cohort labels.
- Let L_{κ_i} be the observation length of cohort κ_i.
- Let N_{κ_i} be the number of students in cohort κ_i. The values of N_{κ_i} may be determined by drawing $\mathcal{K}$ iid samples from a cohort size distribution P_N. If this is the case then we assume that this distribution is independent of all other distributions.

for (each cohort κ_i) **do**
- draw N_{κ_i} iid covariate samples $\mathbf{x}_e$ from the distribution P_X
- foreach $\mathbf{x}_e$ draw an iid (enrollment vector, event type) sample $(\mathbf{e}, b_e)$ from $P_{\mathbf{E},B|\mathbf{x}_e}$
- for each $\mathbf{e}$ compute the event semester $s_e = \max\{t : e_t = 1\}$
- for each sample, compute the (event time, censor time) values (t_e, t_c) as follows

$$t_e = s_e, \qquad t_c = L_{\kappa_i} \qquad (NSS)$$

$$t_e = \sum_{t=1}^{\mathsf{T}_e} e_t, \qquad t_c = \sum_{t=1}^{L_{\kappa_i}} e_t \qquad (NSE)$$

- for each sample $(\mathbf{x}_e, s_e, t_e, t_c, b_e)$ generate an observed sample of the form $(\kappa, \mathbf{x}, t, c, b)$ = (cohort label, covariate value, time value, censor flag, event type) by applying the following operation

$$(\kappa, \mathbf{x}, t, c, b) = \begin{cases} (\kappa_i, \mathbf{x}_e, t_e, 0, b_e), & s_e \leq L_{\kappa_i} \\ (\kappa_i, \mathbf{x}_e, t_c, 1, b_e), & s_e > L_{\kappa_i} \end{cases}$$

end for

The total number of observed samples from all cohorts is $N = N_{\kappa_1} + N_{\kappa_2} + ... + N_{\kappa_{\mathcal{K}}}$ and the complete data set is denoted D_N. Note that if $c = 1$ then the true value of the event type b is not observed in practice even though it is produced as a part of this synthesis process.

Sample Plan 6 (covariate-dependent staged censoring with dropout estimation):
- Let $\mathcal{K}$ = number student cohorts, and let $\kappa_1 \leq \kappa_2 \leq ... \leq \kappa_{\mathcal{K}}$ be the ordered cohort labels.
- Let L_{κ_i} be the observation length of cohort κ_i.
- Let N_{κ_i} be the number of students in cohort κ_i. The values of N_{κ_i} may be determined by drawing $\mathcal{K}$ iid samples from a cohort size distribution P_N. If this is the case then we assume that this distribution is independent of all other distributions.

for (each cohort κ_i) **do**
- draw N_{κ_i} iid covariate samples $\mathbf{x}_e$ from the distribution P_X
- foreach $\mathbf{x}_e$ draw an iid (enrollment vector, event type) sample $(\mathbf{e}, b_e)$ from $P_{\mathbf{E},B|\mathbf{x}_e}$
- for each $\mathbf{e}$ compute the event semester $s_e = \max\{t : e_t = 1\}$
- for each sample, compute the (event time, censor time) values (t_e, t_c) as follows

$$t_e = s_e, \qquad t_c = L_{\kappa_i}, \qquad t_d = \max\{t : e_t = 1, t \leq L_{\kappa_i}\} \qquad (NSS)$$

$$t_e = \sum_{t=1}^{\mathsf{T}_e} e_t, \qquad t_c = \sum_{t=1}^{L_{\kappa_i}} e_t, \qquad t_d = \sum_{t=1}^{L_{\kappa_i}} e_t \qquad (NSE)$$

- for each $\mathbf{e}$ compute the drop estimation flag

$$d = \begin{cases} 1, & \text{if } e_t = 0 \text{ for all } t \text{ satisfying } L - K < t \leq L \\ 0, & \text{otherwise} \end{cases}$$

- for each sample $(\mathbf{x}_e, s_e, t_e, t_c, t_d, b_e, d)$ generate an observed sample of the form $(\kappa, \mathbf{x}, t, c, b)$ = (cohort label, covariate value, time value, censor flag, event type) as follows

$$(\kappa, \mathbf{x}, t, c, b) = \begin{cases} (\kappa_i, \mathbf{x}_e, t_e, 0, b_e), & (s_e \leq \tilde{L}_{\kappa_i}) \\ (\kappa_i, \mathbf{x}_e, t_c, 1, b_e), & (s_e > \tilde{L}_{\kappa_i}) \text{ and } (d = 0) \\ (\kappa_i, \mathbf{x}_e, t_d, 0, \mathsf{D}), & (s_e > \tilde{L}_{\kappa_i}) \text{ and } (d = 1) \end{cases}$$

where $\tilde{L}_{\kappa_i}$ given by (18).

end for

The total number of observed samples from all cohorts is $N = N_{\kappa_1} + N_{\kappa_2} + ... + N_{\kappa_{\mathcal{K}}}$ and the complete data set is denoted D_N. Note that if $c = 1$ then the true value of the event type b is not observed in practice even though it is produced as a part of this synthesis process.

The formal problem statement for the covariate-dependent TTG problem with staged censoring and dropout estimation is as follows.

TTG-V6: *Let $D_N = ((\kappa_1, \mathbf{x}_1, t_1, c_1, b_1), (\kappa_2, \mathbf{x}_2, t_2, c_2, b_2), ..., (\kappa_N, \mathbf{x}_N, t_N, c_N, b_N))$ be a collection of N samples that represent (cohort label, covariate value, time value, censor flag, event type) values for students from a particular institution. Assume that these samples are generated according to Sample Plan 6 with an unknown distribution $P_{X,\mathbf{E},B}$. Given D_N the goal is to estimate the prediction functions (hazard, cessation, survival, and probability) for the institution.*

The consequence of independent censoring for the covariate-dependent case is similar to before. In particular, if we let $T_o = \min(T_e, T_c)$ be the observed time random variable, C be the observed censor flag random variable, and $P_{T_o,C,B|\mathbf{x}}$ be the probability distribution of the observed samples,

then the marginal distribution $P_{T_o|\mathbf{x}}$ is given by

$$P_{T_o|\mathbf{x}}(t) = \sum_{c=0}^{1} \sum_{b \in \mathcal{B}} P_{T_o,C,B|\mathbf{x}}(t,c,b)$$

and the *observed* hazard function is defined

$$h_{o,\mathbf{x}}(t,b) = \frac{P_{T_o,C,B|\mathbf{x}}(t,0,b)}{P_{T_o|\mathbf{x}}(T_o \geq t)}$$

Then independent censoring implies that $h_{o,\mathbf{x}} = h_{\mathbf{x}}$, i.e.

$$\frac{P_{T_e,B|\mathbf{x}}(t,b)}{P_{T_e|\mathbf{x}}(T_e \geq t)} = \frac{P_{T_o,C,B|\mathbf{x}}(t,0,b)}{P_{T_o|\mathbf{x}}(T_o \geq t)}$$

This allows the same type of hazard estimation algorithms as before. Indeed, possible methods for estimating the covariate-dependent hazard functions include the following.

1. **Distinct Covariate Groups:** Partition the data into a separate group of samples for each distinct value of $\mathbf{x}$, and apply the DKM method (or any other covariate-free method) to each group separately to estimate the covariate-dependent hazard functions.

2. **Local Interpolation:** Form the hazard estimate at each distinct $\mathbf{x}$ by including the data samples from the nearest neighbors of $\mathbf{x}$ in the DKM estimate.

3. **Global Interpolation:** Fit the full data set to a parametric (or semi-parametric) model which takes on a mathematical form that interpolates across values of $\mathbf{x}$. Possible models include the multinomial logistic regression model (for continuous- and discrete-valued covariates), and the multinomial probability model (for discrete-valued covariates) [2, 4, 23, 27].

This report explores the first two methods above, starting with an example of the *distinct covariate groups* method in the next section.

Remark 8. Note that regardless of the hazard estimation method, if weighted data are used to mitigate the drop delay bias then the weights must be determined for each distinct value of $\mathbf{x}$ separately (perhaps using information from the nearest neighbors of $\mathbf{x}$).

8.1 Distinct Covariate Groups Example

This section produces hazard and cessation estimates for UNM students as a function of HS-GPA using the *distinct covariate groups* method. We also seek to compare the cessation functions of students from the following two demi-decades.

- *UNM FTFT fall cohorts 2006-2011:* These students sought degrees before UNM required all four-year programs across campus to reduce the total number of credit hours to 120.

- *UNM FTFT fall cohorts 2013-2017:* These students sought degrees after UNM required all four-year programs across campus to reduce the total number of credit hours to 120.

Specifically, the HS-GPA values are mapped into the four groups in the table below. This table also shows the population size for each group for each of the two demi-decades.

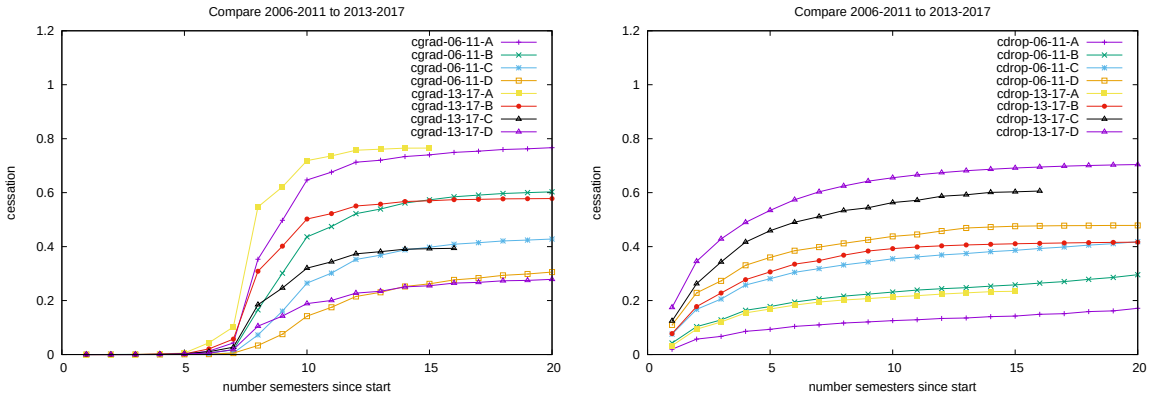
Group Label	HS-GPA Range	2006-2011 population	2013-2017 population
A	3.75-4.50	2601	1608
B	3.25-3.75	6332	4933
C	2.75-3.23	6783	6078
D	0.00-2.75	3788	3977

Prediction functions are produced for each group using data from the students that populate the group. Missing HS-GPA values are imputed as follows. One of the other performance scores (ACT, SAT, or Units-GPA) is mapped to an imputed HS-GPA value using a linear predictor that is fit to historical data. The imputation is performed with the first non-missing score, starting with ACT, then SAT, and finally Units-GPA.

A DKM model is built for each of the four HS-GPA groups in each of the two demi-decade categories yielding a total of 8 models. Each of the 8 models contains *hazard* and *cessation* functions for both the *grad* and *drop* event types. These models are built using the following parameters.

- Event times are determined using the NSS option.
- Drop labels are estimated using $K = 2$ semesters.
- The *weighted data* (WD) method is used to mitigate the drop delay bias.
- The *exponential* model is used to extend each hazard model. The plots below only show the extended values out to 20 semesters.

The first two plots show the 8 *grad* cessation functions C_{grad} on the left and the 8 *drop* cessation functions C_{drop} on the right.

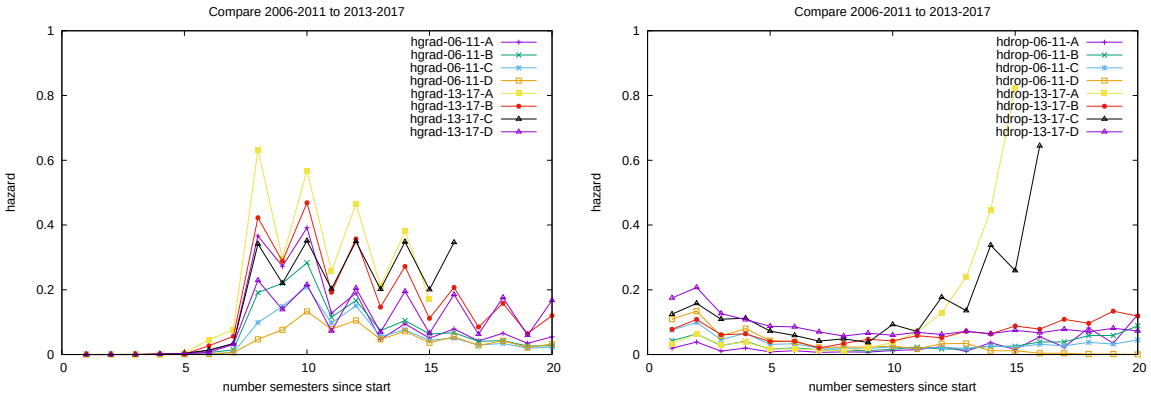


We make the following observations.

- There is a *substantial* difference in graduation timing and success for different HS-GPA categories. Indeed, the long term graduation rate differs approximately 50% between the lowest and highest HS-GPA categories. In addition, students in higher HS-GPA categories tend to graduate sooner.
- There is also a *substantial* difference in dropout behavior for different HS-GPA categories.
- The 2013-2017 cohort graduates sooner than the 2006-2011 cohort, regardless of HS-GPA.

- The 2006-2011 cohort in HS-GPA categories B-D appear to have a slightly better chance of eventually graduating than the 2013-2017 cohort (according to the current extrapolation model).
- The largest dropout probability increases occur in the first few semesters.
- Dropout rates are much higher for the 2013-2017 cohort than the 2006-2011 cohort.

The two plots below show the 8 *grad* hazard functions h_{grad} on the left and the 8 *drop* hazard functions h_{drop} on the right. Note that hazard function extrapolation begins after semester 12 for the 2013-2017 cohort.



8.2 Local Interpolation with Nearest Neighbor DKM (NN-DKM)

This section develops the *Nearest Neighbor DKM* (NN-DKM) method which forms a DKM hazard estimate at each distinct $\mathbf{x}$ by including data samples from the nearest neighbors of $\mathbf{x}$. Because each covariate attribute takes on a finite number of values, the total number of distinct $\mathbf{x}$ values is finite.

We start by defining the dictionary of distinct covariate values as follows. Let d be the number of covariate components (i.e. the dimension of the covariate vector). Let $\mathcal{X}_m$ be the set of distinct values for the m^{th} component of $\mathbf{x}$. Then the dictionary of distinct $\mathbf{x}$ values is given by $\mathcal{X} = \mathcal{X}_1 \times \mathcal{X}_2 \times ... \times \mathcal{X}_d$. We will use the notation $\boldsymbol{\xi}$ to refer to distinct members of $\mathcal{X}$, and retain the notation $\mathbf{x}$ for covariate samples. Both $\boldsymbol{\xi}$ and $\mathbf{x}$ are members of $\mathcal{X}$. Let $D_{\boldsymbol{\xi}}$ be the set of individual data samples $(\kappa, \mathbf{x}, t, c, b)$ whose covariate value satisfies $\mathbf{x} = \boldsymbol{\xi}$.

Nearest neighbors are determined as follows. Let $\Delta(\cdot, \cdot)$ be a function such that $\Delta(\boldsymbol{\xi}_i, \boldsymbol{\xi}_j)$ gives the *distance* between any two values $\boldsymbol{\xi}_i$ and $\boldsymbol{\xi}_j$ from $\mathcal{X}$. Specific examples of Δ are described below. Then the neighbors of any $\boldsymbol{\xi} \in \mathcal{X}$ can be ordered by their distance to $\boldsymbol{\xi}$. Let $\mathcal{N}_{\boldsymbol{\xi},k} = \{\boldsymbol{\xi}_1, \boldsymbol{\xi}_2, ..., \boldsymbol{\xi}_k\}$ be the k nearest neighbors of $\boldsymbol{\xi}$ from $\mathcal{X}$, and let $D_{\mathcal{N}_{\boldsymbol{\xi},k}} = D_{\boldsymbol{\xi}} \cup D_{\boldsymbol{\xi}_1} \cup D_{\boldsymbol{\xi}_2} \cup ... \cup D_{\boldsymbol{\xi}_k}$ be the collection of data samples corresponding to $\boldsymbol{\xi}$ and its nearest neighbors.

The NN-DKM method computes h_{grad} and h_{drop} hazard functions for each $\boldsymbol{\xi} \in \mathcal{X}$ by applying the DKM method to the data in $D_{\mathcal{N}_{\boldsymbol{\xi},k}}$[12]. Then the other prediction functions are derived from the hazard in the same way as before. To determine the value of k for the NN-DKM method we choose k to be the smallest value such that

- the number of individual samples in $D_{\mathcal{N}_{\boldsymbol{\xi},k}}$ is at least N_{min}, and

[12]Note that the *weighted data* method must be applied separately to the data at each value of $\boldsymbol{\xi}$.

- all neighbors at distance $\Delta(\boldsymbol{\xi}, \boldsymbol{\xi}_k)$ are included in $\mathcal{N}_{\boldsymbol{\xi},k}$.

This guarantees that the DKM estimates at each value of $\boldsymbol{\xi}$ are formed with at least N_{min} samples, and that all covariates not in $\mathcal{N}_{\boldsymbol{\xi},k}$ are strictly further away from $\boldsymbol{\xi}$ than the covariates in $\mathcal{N}_{\boldsymbol{\xi},k}$.

To complete the NN-DKM method we must specify a distance function $\Delta(\cdot)$. Many conventional distance functions are designed for continuous data (e.g. Euclidean distance, Mahalanobis distance, Hausdorff distance, ...) and are not the best choice in our case. If all the individual covariates were *binary* then the *Hamming distance* would be a suitable choice. The distance function we choose can be viewed as an extension of the Hamming distance. Let $\xi_{i,m}$ be the m^{th} component of $\boldsymbol{\xi}_i$. We define the distance function

$$\Delta(\boldsymbol{\xi}_i, \boldsymbol{\xi}_j) = \sum_{m=1}^{d} \delta_m(\xi_{i,m}, \xi_{j,m})$$

where $\delta_m(\cdot, \cdot)$ is a distance between any two values ξ_1 and ξ_2 from $\mathcal{X}_m$. One possibility is to choose $\delta_m(\cdot, \cdot)$ to be

$$\delta_m(\xi_1, \xi_2) = \begin{cases} 0, & \xi_1 = \xi_2 \\ 1, & \xi_1 \neq \xi_2 \end{cases}$$

for every $m = 1, 2, ..., d$. This gives the Hamming distance for binary components, and a *same/not-same* distance for the others. This is a special case of the generalized covariate distance δ_m defined next.

Without loss of generality, let each component dictionary take the form $\mathcal{X}_m = \{1, 2, ...|\mathcal{X}_m|\}$, i.e. integer values starting at 1. Then δ_m can be defined by an $|\mathcal{X}_m|$ by $|\mathcal{X}_m|$ matrix where $\delta_m(\xi_1, \xi_2)$ is simply the $(\xi_1, \xi_2)^{th}$ component of the matrix. For example, if $|\mathcal{X}_m| = 3$ then the matrix

$$\delta_m = \begin{bmatrix} 0 & 1 & 1 \\ 1 & 0 & 1 \\ 1 & 1 & 0 \end{bmatrix}$$

will implement the *same/not-same* distance function above. When the covariates have a natural order then δ_m can be specified in a way that reflects this order. For example, if the m^{th} component corresponds to grades from $\{A, B, C, D\}$ and $\mathcal{X}_m = \{1, 2, 3, 4\}$ (so that $A = 1$, $B = 2$, $C = 3$ and $D = 4$), then the matrix might take the form

$$\delta_m = \begin{bmatrix} 0 & 1 & 2 & 3 \\ 1 & 0 & 1 & 2 \\ 2 & 1 & 0 & 1 \\ 3 & 2 & 1 & 0 \end{bmatrix}$$

Although the δ_m matrix can be specified in any way that is appropriate for the end application, the following restrictions will ensure that δ_m (and Δ) are proper distance functions (i.e. *metrics*).

- all entries in the δ_m matrix are nonnegative
- the diagonal entries in the δ_m matrix are 0
- the δ_m matrix is symmetric
- δ_m satisfies $\delta_m(\xi_1, \xi_2) \leq \delta_m(\xi_1, \xi_3) + \delta_m(\xi_3, \xi_2)$ for all ξ_1, ξ_2, ξ_3.

8.2.1 TTG at UNM as a function of ETHNICITY

This example uses the NN-DKM method to estimate prediction functions for UNM students as a function of ETHNICITY. The data consists of UNM FTFT fall cohorts 2013-2017, and the data was collected during the summer of 2019. The nine ETHNICITY categories tracked at UNM are listed in the table below, along with the population size for each category.

Group Label	ETHNICITY	FTFT 2013-2017 population
0	Hispanic	8593
1	American-Indian	609
2	Asian	676
3	Black	391
4	Hawaiian	31
5	White	5260
6	Two-or-more-races	655
7	Unknown-race	128
8	Non-res-alien	261

Although there is no natural order to these categories, we impose an order based on population size. The motivation for this is two-fold. First we hypothesize that categories with similar population sizes are (slightly) more likely to have a common experience at UNM. Second, this type of ordering ensures that the closest neighbors have a similar size so that the nearest neighbor data set $D_{\mathcal{N}_{\boldsymbol{\xi},k}}$ for a smaller category is not overwhelmed by samples from a larger category. Specifically, the distances between categories take values $0, 1, ..., 8$ based on population size difference, and are determined as follows. For each category x_i

- $a_{i,j}$ = absolute population size difference between category x_i and each x_j
- order the absolute distances $a_{i,j}, j = 0, 1, ..., 8$ from smallest to largest
- the distance $\delta(x_i, x_j)$ is equal to the location of $a_{i,j}$ in this ordered list

The resulting distance matrix for the UNM data above is

$$\begin{array}{ccccccccc}
0 & 4 & 2 & 5 & 8 & 1 & 3 & 7 & 6 \\
8 & 0 & 2 & 3 & 6 & 7 & 1 & 5 & 4 \\
8 & 2 & 0 & 3 & 6 & 7 & 1 & 5 & 4 \\
8 & 2 & 5 & 0 & 6 & 7 & 4 & 3 & 1 \\
8 & 4 & 6 & 3 & 0 & 7 & 5 & 1 & 2 \\
1 & 4 & 2 & 5 & 8 & 0 & 3 & 7 & 6 \\
8 & 2 & 1 & 3 & 6 & 7 & 0 & 5 & 4 \\
8 & 4 & 6 & 3 & 1 & 7 & 5 & 0 & 2 \\
8 & 4 & 6 & 1 & 3 & 7 & 5 & 2 & 0
\end{array}$$

The minimum number of neighborhood samples is set to $N_{min} = 390$ so that only three of the nine categories require neighborhood sizes $k > 0$ to achieve $|D_{\mathcal{N}_{\boldsymbol{\xi},k}}| \geq N_{min}$. These categories, their neighbors, and their neighborhood populations are shown in the table below.

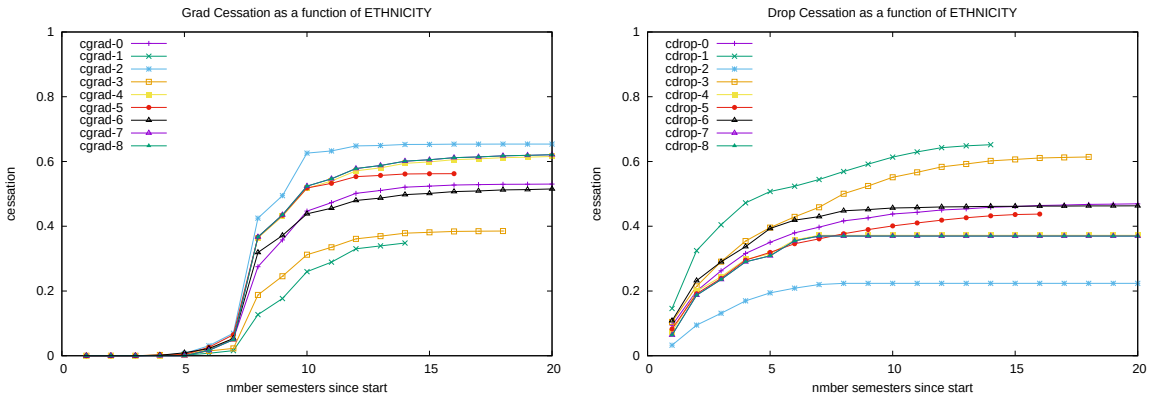
Category	Nearest Neighbor Categories	Neighborhood Population
Hawaiian	Unknown-race, Non-res-alien	31+128+261 = 421
Unknown-race	Non-res-alien	128+261 = 390
Non-res-alien	Unknown-race	261+128 = 390

Note that the last two categories end up with the exact same neighborhood samples, so the prediction estimates for these two categories will be identical.

A DKM model is built for each of the 9 categories using nearest neighbor data determined with the distance matrix above. Each of the 9 models contains *hazard* and *cessation* functions for both the *grad* and *drop* event types. These models are built using the following parameters.

- Event times are determined using the NSS option.
- Drop labels are estimated using $K = 2$ semesters.
- The *weighted data* (WD) method is used to mitigate the drop delay bias.
- The *exponential* model is used to extend each hazard model. The plots below only show the extended values out to 20 semesters.

The plots below show the 9 *grad* cessation functions C_{grad} on the left and the 9 *drop* cessation functions C_{drop} on the right.



We make the following observations.

- There is a substantial difference in both event timing and success rate for the different ETHNICITIES.
- The ranking from most successful to least successful category is as follows: Asian, Non-res-alien & unknown-race (tied), Hawaiian, White, Hispanic, Two-or-more-races, Black, American-Indian

8.2.2 TTG at UNM as a function of (GENDER, HS-GPA, ETHNICITY)

This example uses the NN-DKM method to estimate prediction functions for UNM students as a function of covariates (GENDER, HS-GPA, ETHNICITY). The data consists of UNM FTFT fall cohorts 2013-2017, and the data was collected during the summer of 2019.

The ETHNICITY covariate consists of the same nine categories with the same distance matrix described in Section 8.2.1. The HS-GPA covariate is processed and quantized into the same four categories as in Section 8.1 with the following distance matrix

$$\begin{bmatrix} 0 & 1 & 2 & 3 \\ 1 & 0 & 1 & 2 \\ 2 & 1 & 0 & 1 \\ 3 & 2 & 1 & 0 \end{bmatrix}$$

The GENDER covariate takes on two values, Male and Female, with distance matrix

$$\begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}$$

The total number of distinct covariate values is $2 \times 4 \times 9 = 72$ and the 2013-2017 data set maps into these categories according to the following statistics.

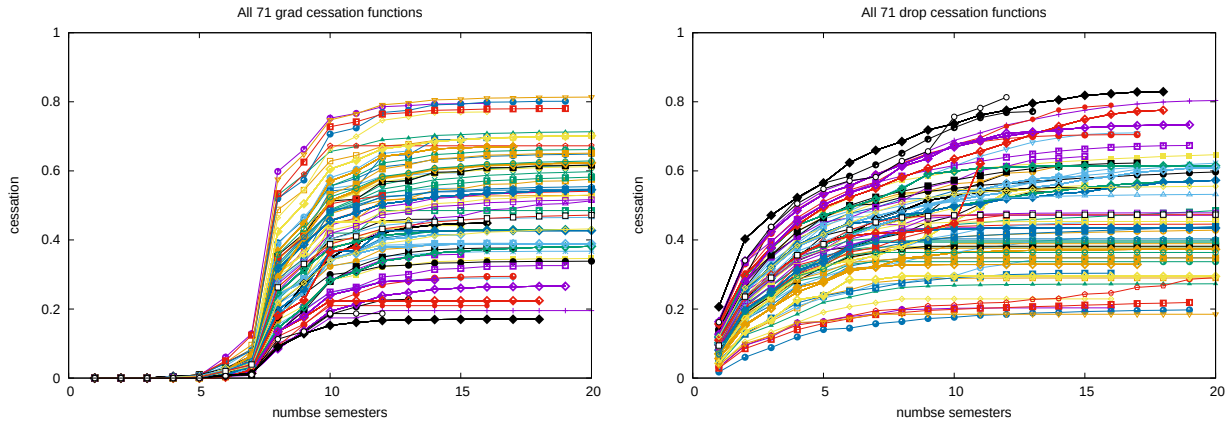
- 71 (out of 72) covariate values are witnessed by at least 1 data sample.
- 35 (out of 72) covariate values are witnessed by < 50 data samples.
- 24 (out of 72) covariate values are witnessed by > 100 data samples.

A DKM model is built for each of the 71 categories using nearest neighbor samples to ensure that each model is built with at least $N_{min} = 300$ samples. These models are built using the following parameters.

- Event times are determined using the NSS option.
- Drop labels are estimated using $K = 2$ semesters.
- The *weighted data* (WD) method is used to mitigate the drop delay bias.
- The *exponential* model is used to extend each hazard model. The plots below only show the extended values out to 20 semesters.

Each of the 71 models contains *hazard* and *cessation* functions for both the *grad* and *drop* event types.

The first two plots show the 71 *grad* cessation functions C_{grad} on the left and the 71 *drop* cessation functions C_{drop} on the right. These plots show a *significant* difference in event timing and success rate for the different covariate values. A more detailed analysis of these results is presented below, but first we explore the inner workings of the nearest neighbor method to develop a better understanding of this local interpolation method.



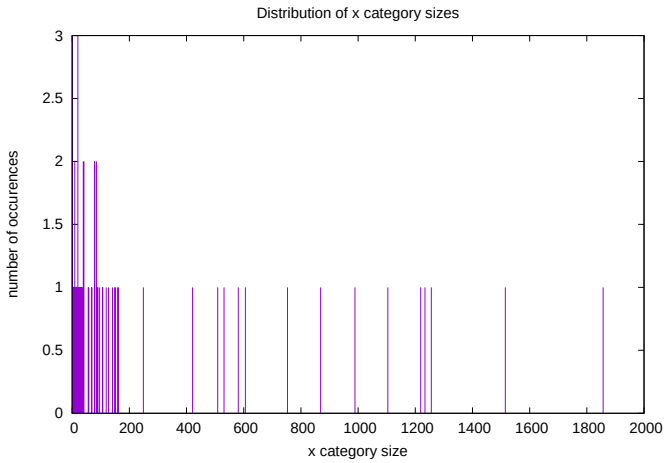
First we investigate the the relative influence of *covariate samples* versus *neighbor samples* on the individual prediction functions. This influence will depend on

1. the distance to the furthest neighbor,

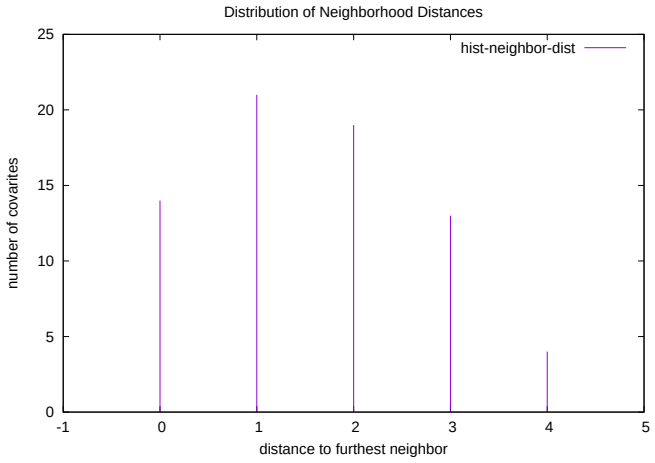
2. the number of neighbors, and

3. the relative number of *covariate samples* to *neighbor samples*

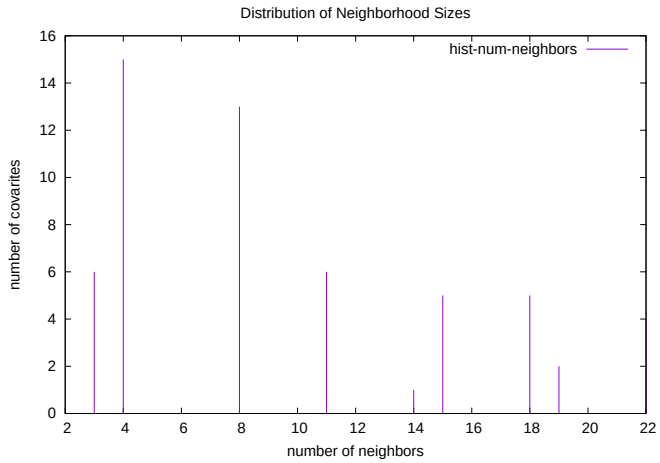
The plot below shows the number of samples associated with each distinct covariate value. Neighbor samples are used only when the number of covariate samples is less than $N_{min} = 300$, which is the case for a majority of 71 covariate values.



A histogram of the distance to the furthest neighbor is shown below. Only covariates witnessed by a very small number of samples have furthest neighbor distances greater than 2.



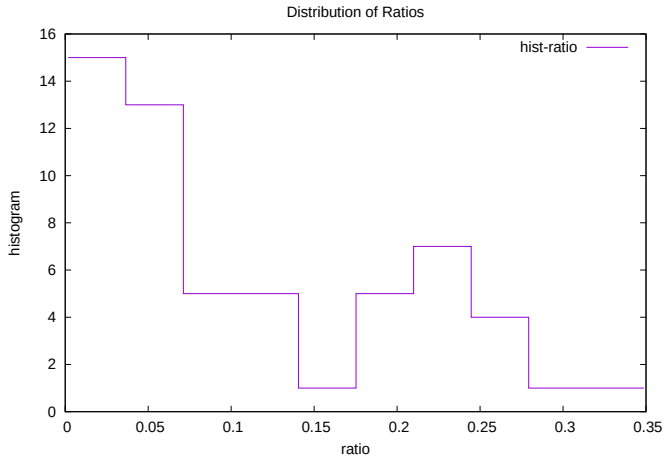
A histogram of the *number of neighbors* k is shown below. Covariate values with a large number of neighbors have a small number of witness samples, and are surrounded by neighbors with a small number of witness samples, while covariates with a small number of neighbors have a large number of witness samples, or are surrounded by neighbors with a large number of witness samples.



A histogram of the ratio

$$\frac{\text{number of covariate samples}}{\text{number of neighbor samples}}$$

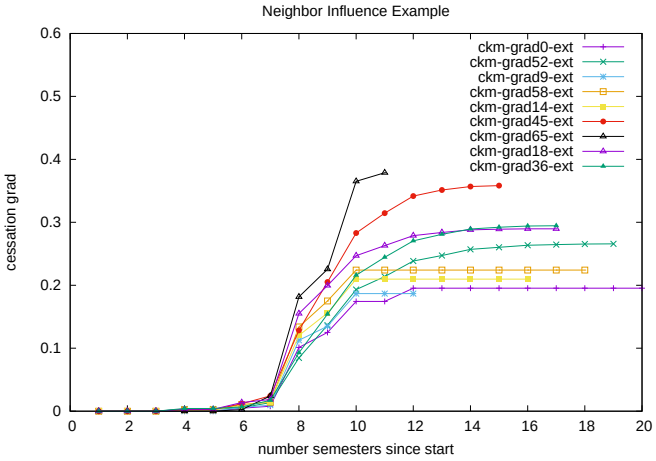
is shown below. Smaller ratios imply that the neighbors have a larger influence on the prediction functions.



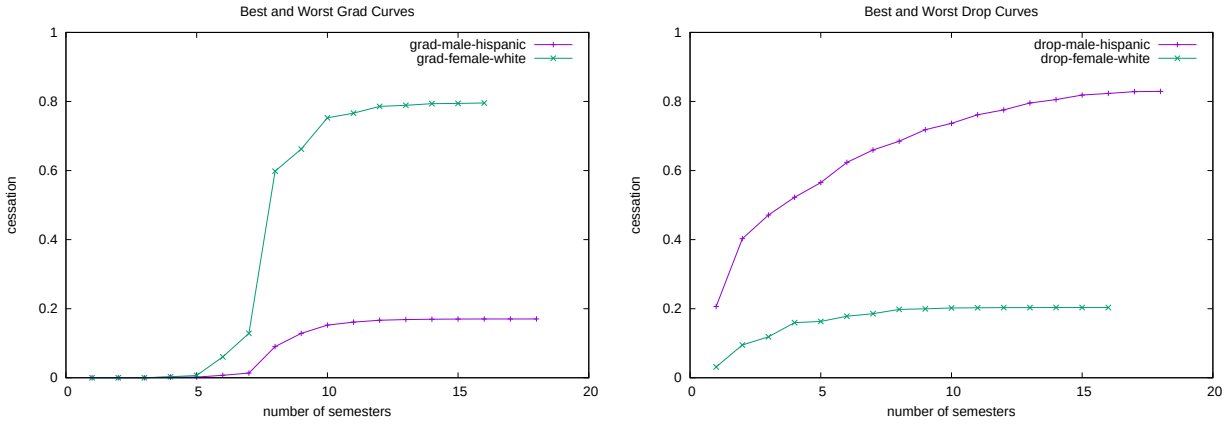
Consider the covariate value (Male, American-Indian, gpa=0.0-2.75) which is witnessed by only 34 samples in the data set. The nearest neighbors for this covariate are as follows.

```
(ID=0) Male, American-Indian, gpa=0-2.75
Number of Samples: 34
Nearest Neighbors: (8)
    (dist=1.00,ns=  27) : (ID=52) Female American-Indian gpa=0-2.75
    (dist=1.00,ns=  88) : (ID= 9) Male American-Indian gpa=2.75-3.25
    (dist=1.00,ns=  41) : (ID=58) Male Black gpa=0-2.75
    (dist=2.00,ns=  29) : (ID=14) Female Black gpa=0-2.75
    (dist=2.00,ns=  78) : (ID=45) Male American-Indian gpa=3.25-3.75
    (dist=2.00,ns=  15) : (ID=65) Male Asian gpa=0-2.75
    (dist=2.00,ns=  85) : (ID=18) Male Black gpa=2.75-3.25
    (dist=2.00,ns= 127) : (ID=36) Female American-Indian gpa=2.75-3.25
```

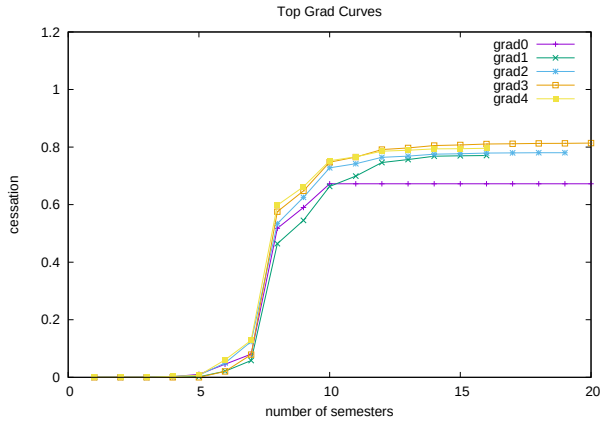
A plot of the *grad* cessation function estimates C_{grad} for this covariate and its nearest neighbors is shown below. This results suggests that the covariate (Male, American-Indian, gpa=0.0-2.75) has a nontrivial influence on the prediction function even though it is only witnessed by 34 samples.

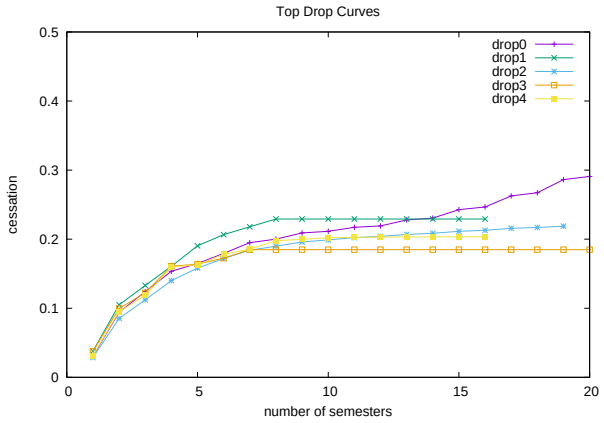


Now we present more detailed results of the cessation modeling results for UNM FTFT fall cohorts 2013-2017. For convenience (and accuracy) we restrict to covariates that are witnessed by at least 100 samples. The next few plots show cessation estimates for (a) the best and worst cases, (b) the top 5 cases, (c) the bottom 7 cases, and (d) the middle 10 cases.



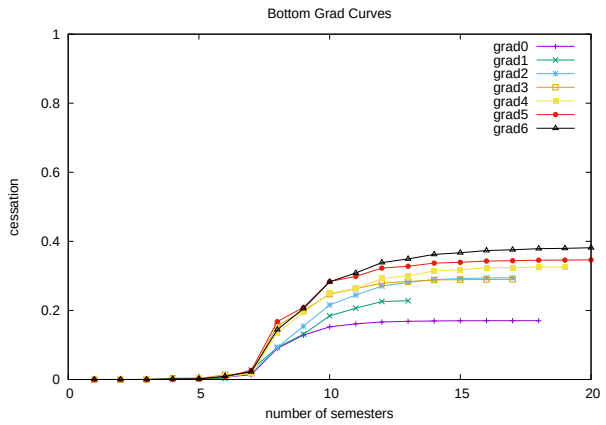
1. **Best Case:** Female, White, gpa=3.75-4.5
2. **Worst Case:** Male, Hispanic, gpa=0-2.75

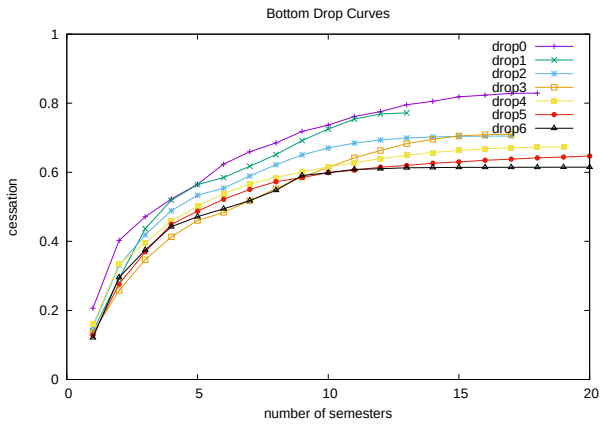




Top 5: In Order of Increasing Performance:

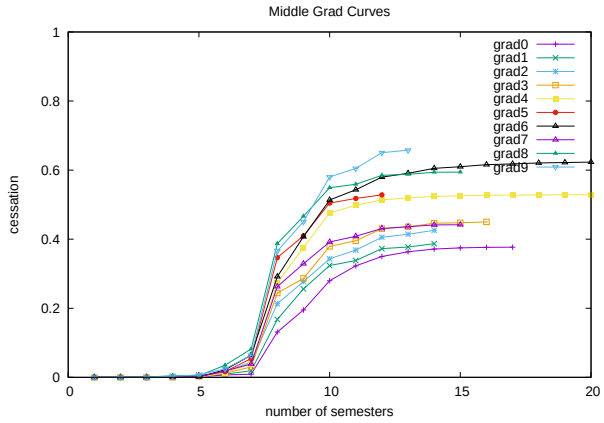
0. Female Asian gpa=3.75-4.5
1. Male Hispanic gpa=3.75-4.5
2. Female Hispanic gpa=3.75-4.5
3. Male White gpa=3.75-4.5
4. Female White gpa=3.75-4.5

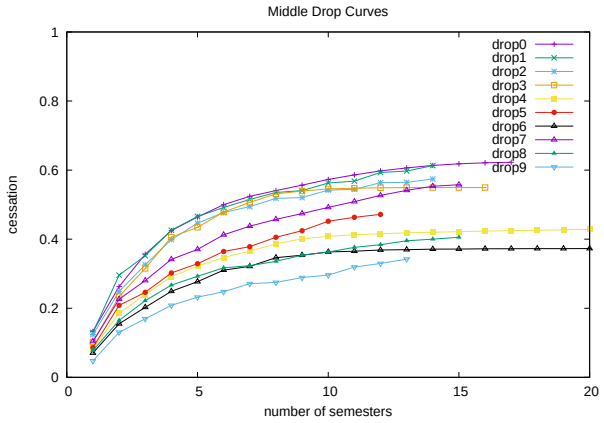




Bottom 7: In Order of Increasing Performance:

0. Male Hispanic gpa=0-2.75
1. Female Hispanic gpa=0-2.75
2. Female American-Indian gpa=2.75-3.25
3. Male Black gpa=2.75-3.25
4. Male White gpa=0-2.75
5. Female White gpa=0-2.75
6. Female American-Indian gpa=3.25-3.75





Middle 10: In Order of Increasing Performance:

0. Male Hispanic gpa=2.75-3.25
1. Male White gpa=2.75-3.25
2. Female Hispanic gpa=2.75-3.25
3. Female White gpa=2.75-3.25
4. Male Hispanic gpa=3.25-3.75
5. Male White gpa=3.25-3.75
6. Female Hispanic gpa=3.25-3.75
7. Male Asian gpa=3.25-3.75
8. Female White gpa=3.25-3.75
9. Female Asian gpa=3.25-3.75

9 Predicting Future Enrollments

In this section we show how prediction functions can be used to estimate future enrollments. One advantage of this approach is that it allows the user to investigate the impact of various system changes on future enrollments. Examples include

- changes in the incoming student cohort size,
- changes in the 4-year and/or 6-year graduation rates, and
- changes in the 3rd semester retention rate.

We start by defining an *absolute* staged data table whose columns are aligned by absolute time instead of relative time, e.g. the first semester of cohort 2014 is aligned with the fourth semester of cohort 2012. For example, a portion of the absolute staged data table of n_{risk} values for UNM FTFT students in fall cohorts 2006-2018 is shown below.

Absolute Semester:	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25
Stage 1 (Fall 2006): nrisk:	766	624	435	359	271	220	167	126	81	64	0	0	0	0	0
Stage 3 (Fall 2007): nrisk:	1476	1107	665	528	374	315	223	175	131	110	79	58	0	0	0
Stage 5 (Fall 2008): nrisk:	2191	2084	1601	1209	779	609	410	337	247	188	142	113	77	53	0
Stage 7 (Fall 2009): nrisk:	2470	2371	2264	2162	1641	1196	748	582	393	326	250	207	158	133	89
Stage 9 (Fall 2010): nrisk:	2814	2657	2441	2362	2244	2120	1515	1089	694	542	357	284	203	143	96
Stage 11 (Fall 2011): nrisk:	3341	3043	2666	2525	2330	2241	2140	2029	1427	1017	579	458	299	234	152
Stage 13 (Fall 2012): nrisk:	0	0	3424	3164	2772	2603	2409	2280	2141	2032	1329	921	492	367	205
Stage 15 (Fall 2013): nrisk:	0	0	0	0	3518	3225	2878	2691	2502	2409	2259	2093	1149	788	400
Stage 17 (Fall 2014): nrisk:	0	0	0	0	0	0	3132	2860	2582	2430	2262	2155	2005	1828	864
Stage 19 (Fall 2015): nrisk:	0	0	0	0	0	0	0	0	3327	3069	2727	2523	2341	2204	2015
Stage 21 (Fall 2016): nrisk:	0	0	0	0	0	0	0	0	0	0	3402	3112	2708	2475	2259
Stage 23 (Fall 2017): nrisk:	0	0	0	0	0	0	0	0	0	0	0	0	3219	2833	2396
Stage 25 (Fall 2018): nrisk:	0	0	0	0	0	0	0	0	0	0	0	0	0	0	2653

sum this column
to get NSum(21)

Let

$$N_{sum}(t_a) = \sum_{\text{all past cohorts}} \mathsf{nrisk}(t_a)$$

be the sum of nrisk values from all cohorts with entry times in the past (i.e. all entry times prior to and including t_a). If there were no stopouts then $N_{sum}(t_a)$ would be equal to the total enrollment at semester t_a. Let ρ be the fraction of at-risk students that are *stopped out* at a given semester. If ρ is the same for all semesters, then an estimate of the total enrollment at semester t_a is

$$N_{enroll}(t_a) = (1-\rho)N_{sum}(t_a)$$

The Survival function $S(t) = S_{grad}(t) + S_{drop}(t)$ gives the probability of *not* experiencing an event (*grad* or *drop*) until after semester t, i.e. the probability of still being active at semester t. This function can be used to make predictions about future enrollments. Given a cohort of N students that all start at the same semester, the expected n_{risk} value at a future semester t is given by $N \cdot S(t)$. This formula can be used to extend the n_{risk} values beyond the current semester for cohorts that have already started and still have active students. It can also be used to extend the n_{risk} values for future cohorts once a starting population size is given. Thus, future enrollment estimates can be obtained using the following steps.

1. Choose starting population sizes for future cohorts. Possible choices include the largest, smallest, or average over the most recent cohorts.

2. For all cohorts, past and future, use the survival function S and the formula $N \cdot S(t)$ to extend the n_{risk} values into future semesters.

3. Compute N_{sum} values for future semesters.

4. Compute $N_{enroll} = (1-\rho)N_{sum}$ for future semesters.

An example of an absolute staged data table resulting from this method is shown below. This table includes data from UNM FTFT fall cohorts 2006-2018 from the past, and imputes future cohorts 2019-2023 using starting populations that are the average of past starting populations. Note that this method requires that S be a *complete* function, i.e. it must be estimated out to the final event

time $\hat{T}_e$ where the corresponding probability function satisfies

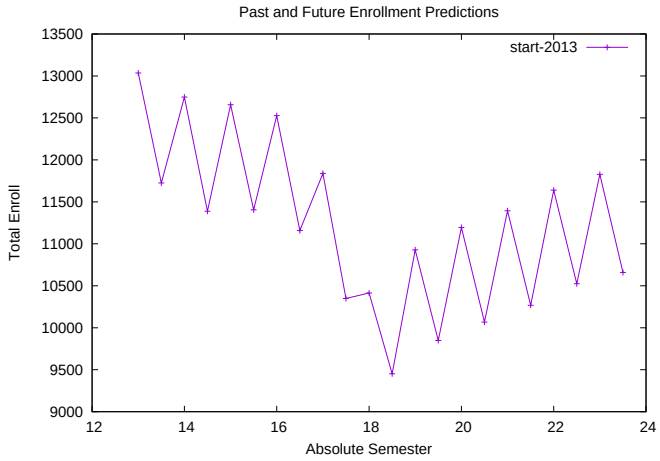
$$\sum_{t=1}^{\hat{\mathsf{T}}_e} \sum_{b \in \mathcal{B}} \hat{P}_{T_e,B}(t,b) \approx 1$$

If S is incomplete then this method will fail to impute some of the trailing n_{risk} values for each cohort and the corresponding enrollment estimates will be biased low.

Absolute Semester:	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36
Stage 1 (Fall 2006): nrisk:	81	64	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Stage 3 (Fall 2007): nrisk:	131	110	79	58	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Stage 5 (Fall 2008): nrisk:	247	188	142	113	77	53	0	0	0	0	0	0	0	0	0	0	0	0
Stage 7 (Fall 2009): nrisk:	393	326	250	207	158	133	89	63	0	0	0	0	0	0	0	0	0	0
Stage 9 (Fall 2010): nrisk:	694	542	357	284	203	143	96	77	52	37	0	0	0	0	0	0	0	0
Stage 11 (Fall 2011): nrisk:	1427	1017	579	458	299	234	152	118	88	71	48	34	0	0	0	0	0	0
Stage 13 (Fall 2012): nrisk:	2141	2032	1329	921	492	367	205	166	121	94	70	56	38	27	0	0	0	0
Stage 15 (Fall 2013): nrisk:	2502	2409	2259	2093	1149	788	400	312	208	169	123	96	71	57	38	27	0	0
Stage 17 (Fall 2014): nrisk:	2582	2430	2262	2155	2005	1828	864	628	377	295	196	160	116	90	67	54	36	26
Stage 19 (Fall 2015): nrisk:	3327	3069	2727	2523	2341	2204	2015	1895	1310	952	572	447	298	242	176	137	102	82
Stage 21 (Fall 2016): nrisk:	0	0	3402	3112	2708	2475	2259	2164	2046	1925	1330	967	581	454	302	246	178	139
Stage 23 (Fall 2017): nrisk:	0	0	0	0	3219	2833	2396	2252	2088	2001	1892	1780	1230	894	537	420	280	227
Stage 25 (Fall 2018): nrisk:	0	0	0	0	0	0	2653	2425	2142	2014	1867	1789	1692	1591	1100	799	480	375
Stage 27 (Fall 2019): nrisk:	0	0	0	0	0	0	0	0	3245	2966	2620	2463	2284	2188	2069	1946	1345	978
Stage 29 (Fall 2020): nrisk:	0	0	0	0	0	0	0	0	0	0	3245	2966	2620	2463	2284	2188	2069	1946
Stage 31 (Fall 2021): nrisk:	0	0	0	0	0	0	0	0	0	0	0	0	3245	2966	2620	2463	2284	2188
Stage 33 (Fall 2022): nrisk:	0	0	0	0	0	0	0	0	0	0	0	0	0	3245	2966	2620	2463	
Stage 35 (Fall 2023): nrisk:	0	0	0	0	0	0	0	0	0	0	0	0	0	0	3245	2966		

sum these columns to get predicted future enrollments

Summing the columns in the table above and then multiplying these sums by $(1 - \rho)$ yields the enrollment values in the plot below. This plot shows a slight decline in enrollment for recent cohorts, with a substantial dip in 2018. It also shows an enrollment recovery after 2018 as long as the future starting populations are equal to the historical average.



By changing the future starting populations the user can easily investigate the impact of changes in the incoming student cohort size on future enrollments. Similarly, by adjusting the *grad* and *drop* probability functions the user can explore the impact of changes in graduation and retention rates on future enrollments. One of the simplest ways to do this is by shifting some of the probability mass from one probability function to the other while maintaining the shape of both.

10 Semi-supervised Learning

This section takes a very different approach to the estimation of TTG prediction functions. This approach produces an estimate of $P_{T_e,B|X}$ using the maximum likelihood (ML) method with a *semi-supervised* likelihood function[13]. This approach does not require independent censoring, but typically does require a parametric model for $P_{T_e,B|X}$ (or $P_{X,T_e,B}$). The censored data samples represent a type of missing information that can be inferred through the use of the Expectation-Maximization (EM) algorithm to optimize the likelihood function. Indeed, in addition to estimating the parameters of $P_{T_e,B|X}$, the EM algorithm also estimates the probability of each event type (*grad* and *drop*) at each event time for each censored data sample. In addition, unlike the methods in the previous sections this approach produces an estimate of P_X. Thus, it provides a more complete assessment of the data, even though P_X may not be used in the TTG problem. Furthermore, since this approach does not require independent censoring it estimates the probability function directly, and then infers the hazard, cessation, and survival functions using (20), (21), and (22).

Using the product rule of probability we can write

$$\begin{aligned} P_{T_e,B|X} &= \frac{P_{X,T_e,B}}{P_X} \\ &= \frac{P_{T_e,B}P_{X|T_e,B}}{P_X} \\ &= \frac{P_{T_e,B}P_{X|T_e,B}}{\sum_{(T_e,B)\in\mathcal{T}\times\mathcal{B}} P_{T_e,B}P_{X|T_e,B}} \end{aligned} \tag{25}$$

We will use the ML method to estimate $P_{T_e,B}$ and $P_{X|T_e,B}$, and then substitute these estimates into (25) to obtain $P_{T_e,B|X}$. Note that $P_X = \sum_{(T_e,B)\in\mathcal{T}\times\mathcal{B}} P_{T_e,B}P_{X|T_e,B}$ can be viewed as a *mixture* distribution with $|\mathcal{T}\times\mathcal{B}|$ mixture components. It is analogous to a multi-class classification distribution where the number of classes is $|\mathcal{T}\times\mathcal{B}|$, the individual class labels take the form (T_e, B), and the individual class distributions are $P_{X|T_e,B}$.

Covariates are treated in the same way as Section 8. Covariate samples will be represented by a d-tuple $\mathbf{x} = (x_1, x_2, ..., x_d)$ where each component x_i represents a different student attribute. We restrict to case where the covariate vectors are *discrete* and *finite* valued and let $\mathcal{X}$ be the dictionary of distinct $\mathbf{x}$ values. We will use the notation $\boldsymbol{\xi}$ to refer to distinct members of $\mathcal{X}$, and retain the notation $\mathbf{x}$ for covariate samples. Both $\boldsymbol{\xi}$ and $\mathbf{x}$ are members of $\mathcal{X}$.

Note that both $P_{T_e,B}$ and $P_{X|T_e,B}$ are discrete distributions with a finite number of values. Thus, if we know all these values then we know the distributions. If we let

$$\begin{aligned} \pi_{t_e,b} &= P_{T_e,B}(t_e, b) \\ \pi_{\boldsymbol{\xi}|t_e,b} &= P_{X|T_e=t_e,B=b}(\boldsymbol{\xi}) \end{aligned} \tag{26}$$

then our task is to estimate the complete set of parameters[14]

$$\phi = \left\{\pi_{t_e,b}, \pi_{\boldsymbol{\xi}|t_e,b}\right\} \quad \text{for all } \boldsymbol{\xi}\in\mathcal{X} \text{ and } (t_e,b)\in\mathcal{T}\times\mathcal{B}$$

[13]The term *semi-supervised learning* refers to a data driven modeling method that exploits two types of data samples, generally referred to as *labeled* and *unlabeled*. For example, in a pattern classification problem the *labeled* and *unlabeled* data samples take the form (covariate, class label) and (covariate) respectively. In the TTG problem the *uncensored* and *censored* samples play the role of *labeled* and *unlabeled* samples respectively. There are numerous approaches to *semi-supervised learning*, but the one we pursue in this section builds a probability model using the maximum-likelihood (ML) method. We design a likelihood function specifically for the TTG data, and since it is similar to likelihood functions used in *missing data* problems, the TTG likelihood function lends itself to optimization via a so-called *expectation maximization* (EM) algorithm.

[14]This could be viewed as a *nonparametric* probability model because it has a separate parameter for every argument

These parameters can be displayed in matrix form as follows

$$\begin{array}{ccccc}
\pi_{1,\mathsf{G}} & \pi_{\boldsymbol{\xi}_1|1,\mathsf{G}} & \pi_{\boldsymbol{\xi}_2|1,\mathsf{G}} & \cdots & \pi_{\boldsymbol{\xi}_M|1,\mathsf{G}} \\
\pi_{2,\mathsf{G}} & \pi_{\boldsymbol{\xi}_1|2,\mathsf{G}} & \pi_{\boldsymbol{\xi}_2|2,\mathsf{G}} & \cdots & \pi_{\boldsymbol{\xi}_M|2,\mathsf{G}} \\
\cdot & \cdot & \cdot & \cdots & \cdot \\
\pi_{\mathsf{T}_e,\mathsf{G}} & \pi_{\boldsymbol{\xi}_1|\mathsf{T}_e,\mathsf{G}} & \pi_{\boldsymbol{\xi}_2|\mathsf{T}_e,\mathsf{G}} & \cdots & \pi_{\boldsymbol{\xi}_M|\mathsf{T}_e,\mathsf{G}}
\end{array}$$

$$\begin{array}{ccccc}
\pi_{1,\mathsf{D}} & \pi_{\boldsymbol{\xi}_1|1,\mathsf{D}} & \pi_{\boldsymbol{\xi}_2|1,\mathsf{D}} & \cdots & \pi_{\boldsymbol{\xi}_M|1,\mathsf{D}} \\
\pi_{2,\mathsf{D}} & \pi_{\boldsymbol{\xi}_1|2,\mathsf{D}} & \pi_{\boldsymbol{\xi}_2|2,\mathsf{D}} & \cdots & \pi_{\boldsymbol{\xi}_M|2,\mathsf{D}} \\
\cdot & \cdot & \cdot & \cdots & \cdot \\
\pi_{\mathsf{T}_e,\mathsf{D}} & \pi_{\boldsymbol{\xi}_1|\mathsf{T}_e,\mathsf{D}} & \pi_{\boldsymbol{\xi}_2|\mathsf{T}_e,\mathsf{D}} & \cdots & \pi_{\boldsymbol{\xi}_M|\mathsf{T}_e,\mathsf{D}}
\end{array}$$

where $M = |\mathcal{X}|$.

Using (25) the hazard, cessation, and survival functions can be expressed as a function of these parameters as follows. The cessation function is given by

$$\begin{aligned}
C_{\mathbf{x}}(t,b) &= \sum_{t'=1}^{t} P_{T_e,B|\mathbf{x}}(t',b) \\
&= \sum_{t'=1}^{t} \left(\frac{P_{T_e,B}(t',b) P_{X|t,b}(\mathbf{x})}{\sum_{(\tau,\beta)\in\mathcal{T}\times\mathcal{B}} P_{T_e,B}(\tau,\beta) P_{X|\tau,\beta}(\mathbf{x})} \right) \\
&= \frac{\sum_{t'=1}^{t} P_{T_e,B}(t',b) P_{X|t,b}(\mathbf{x})}{\sum_{(\tau,\beta)\in\mathcal{T}\times\mathcal{B}} P_{T_e,B}(\tau,\beta) P_{X|\tau,\beta}(\mathbf{x})} \\
&= \frac{\sum_{t'=1}^{t} \pi_{t',b} \pi_{\mathbf{x}|t',b}}{\sum_{\tau\in\mathcal{T}} \sum_{\beta\in\mathcal{B}} \pi_{\tau,\beta} \pi_{\mathbf{x}|\tau,\beta}}
\end{aligned}$$

The survival function is given by

$$\begin{aligned}
S_{\mathbf{x}}(t,b) &= \sum_{t'=t}^{\mathsf{T}_e} P_{T_e,B|\mathbf{x}}(t',b) \\
&= \sum_{t'=t}^{\mathsf{T}_e} \left(\frac{P_{T_e,B}(t',b) P_{X|t,b}(\mathbf{x})}{\sum_{(\tau,\beta)\in\mathcal{T}\times\mathcal{B}} P_{T_e,B}(\tau,\beta) P_{X|\tau,\beta}(\mathbf{x})} \right) \\
&= \frac{\sum_{t'=t}^{\mathsf{T}_e} P_{T_e,B}(t',b) P_{X|t,b}(\mathbf{x})}{\sum_{(\tau,\beta)\in\mathcal{T}\times\mathcal{B}} P_{T_e,B}(\tau,\beta) P_{X|\tau,\beta}(\mathbf{x})} \\
&= \frac{\sum_{t'=t}^{\mathsf{T}_e} \pi_{t',b} \pi_{\mathbf{x}|t',b}}{\sum_{\tau\in\mathcal{T}} \sum_{\beta\in\mathcal{B}} \pi_{\tau,\beta} \pi_{\mathbf{x}|\tau,\beta}}
\end{aligned}$$

of the probability function. Indeed, knowing the probability at one value $(\boldsymbol{\xi}, t_e, b)$ says nothing about the probability at a neighboring value. In particular this model does not capture the ordering of the time values and the possible relationship between covariate values.

The hazard function is given by

$$\begin{aligned}
h_{\mathbf{x}}(t,b) &= \frac{P_{T_e,B|\mathbf{x}}(t,b)}{\sum_{\beta\in\mathcal{B}} P_{T_e,B|\mathbf{x}}(T_e \geq t,\beta)} \\
&= \frac{P_{T_e,B|\mathbf{x}}(t,b)}{\sum_{t'=t}^{\mathsf{T}_e}\sum_{\beta\in\mathcal{B}} P_{T_e,B|\mathbf{x}}(t',\beta)} \\
&= \frac{\frac{P_{T_e,B}(t,b)P_{X|t,b}(\mathbf{x})}{P_X(\mathbf{x})}}{\sum_{t'=t}^{\mathsf{T}_e}\sum_{\beta\in\mathcal{B}} \frac{P_{T_e,B}(t',\beta)P_{X|t',\beta}(\mathbf{x})}{P_X(\mathbf{x})}} \\
&= \frac{P_{T_e,B}(t,b)P_{X|t,b}(\mathbf{x})}{\sum_{t'=t}^{\mathsf{T}_e}\sum_{\beta\in\mathcal{B}} P_{T_e,B}(t',\beta)P_{X|t',\beta}(\mathbf{x})} \\
&= \frac{\pi_{t,b}\pi_{\mathbf{x}|t,b}}{\sum_{t'=t}^{\mathsf{T}_e}\sum_{\beta\in\mathcal{B}} \pi_{t',\beta}\pi_{\mathbf{x}|t',\beta}}
\end{aligned}$$

We assume that data supplied to the algorithms below consists of iid data samples that are generated and preprocessed as follows.

1. First a total of N samples of the form $(\kappa, \mathbf{x}, t, c, b)$ are generated according to **Sample Plan 6**. In practice these are samples of student data whose *drop* labels have been determined using the *dropout estimation procedure* in Section 5.

2. Then, to compensate for the drop delay bias, the samples are processed using one of the following two options.

   - **reassign:** The data is modified using the *sample reassignment* method described in Section 5.1.3. This data is then processed by the EM algorithm in Section 10.2 below to obtain estimates of the distribution parameters ϕ.
   - **weighted data:** Sample *weights* are added as described in Section 5.1.4 so that the data samples now take the form $(\kappa, \mathbf{x}, t, c, b, w)$. Note that with this option the weights must be determined separately for each distinct value of $\boldsymbol{\xi} \in \mathcal{X}$. For example, the weights for each $\boldsymbol{\xi}$ can be determined by applying the *weighted data method* in Section 5.1.4 to the nearest neighbor data $D_{\mathcal{N}_{\boldsymbol{\xi},k}}$ formed as described in Section 8.2. This data is then processed by the *Weighted EM* algorithm in Section 10.3 below to obtain estimates of the distribution parameters ϕ.

10.1 Standard ML Approach

We assume the the data set D_N consists of iid data samples of the form $(\kappa, \mathbf{x}, t, c, b)$ generated according to **Sample Plan 6** with the *dropout estimation procedure* in Section 5 and the *sample reassignment* method in Section 5.1.3[15]. First we note that stage label κ is not used by the method developed in this section. In addition, if $c = 1$ then b has no meaning so the information content of a censored data sample is simply $(\mathbf{x}, t)$ where $t = t_c$ is a censored time. Similarly the information content of an uncensored data sample is $(\mathbf{x}, t, b)$ where $t = t_e$ is an event time.

Let I_U and I_C be the index sets for *uncensored* and *censored* samples respectively, i.e.

$$I_U = \{i : c_i = 0\}, \qquad I_C = \{i : c_i = 1\}$$

[15]The method developed in this section and the next is also applicable to data generated according to **Sample Plan 5**.

Then the likelihood for the uncensored data (where $t_i = t_{e_i}$) is

$$\begin{aligned} l_U(\phi) &= \prod_{i \in I_U} P_{X,T_e,B}(\mathbf{x}_i, t_i, b_i) \\ &= \prod_{i \in I_U} P_{T_e,B}(t_i, b_i) P_{X|T_e=t_i,B=b_i}(\mathbf{x}_i) \\ &= \prod_{i \in I_U} \pi_{t_i,b_i} \pi_{\mathbf{x}_i|t_i,b_i} \end{aligned} \tag{27}$$

Censoring masks both the true event time and the event type, so the likelihood for the censored data (where $t_{e_i} > t_i$ and b is unknown) is

$$l_C(\phi) = \prod_{i \in I_C} P_{X,T_e}(\mathbf{x}_i, T_e > t_i)$$

Note that the marginal distribution can be written

$$\begin{aligned} P_{X,T_e} &= \sum_{b \in \mathcal{B}} P_{B=b} P_{X,T_e|B=b} \\ &= \sum_{b \in \mathcal{B}} P_{B=b} P_{T_e|B=b} P_{X|T_e,B=b} \\ &= \sum_{b \in \mathcal{B}} P_{T_e,B=b} P_{X|T_e,B=b} \end{aligned}$$

and so the likelihood for the censored samples becomes

$$\begin{aligned} l_C(\phi) &= \prod_{i \in I_C} P_{X,T_e}(\mathbf{x}_i, T_e > t_i) \\ &= \prod_{i \in I_C} \left[\sum_{b \in \mathcal{B}} P_{T_e,B=b}(T_e > t_i) P_{X|T_e>t_i,B=b}(\mathbf{x}_i) \right] \end{aligned}$$

This can be simplified using

$$\begin{aligned} P_{T_e,B=b}(T_e > t_i) &= P_{B=b} \sum_{t_e=t_i+1}^{\mathsf{T}_e} P_{T_e|B=b}(t_e) \\ &= \sum_{t_e=t_i+1}^{\mathsf{T}_e} P_{B=b} P_{T_e=t_e|B=b} \\ &= \sum_{t_e=t_i+1}^{\mathsf{T}_e} P_{T_e=t_e,B=b} \\ &= \sum_{t_e=t_i+1}^{\mathsf{T}_e} \pi_{t_e,b} \end{aligned}$$

and

$$P_{X|T_e>t_i,B=b}(\mathbf{x}_i) = \frac{\sum_{t_e=t_i+1}^{\mathsf{T}_e} \pi_{t_e,b} \pi_{\mathbf{x}_i|t_e,b}}{\sum_{t_e=t_i+1}^{\mathsf{T}_e} \pi_{t_e,b}}$$

where $\pi_{t_e,b}$ and $\pi_{\mathbf{x}_i|t_e,b}$ are defined in (26). This gives

$$\begin{aligned}
l_C(\phi) &= \prod_{i\in I_C}\left[\sum_{b\in\mathcal{B}} P_{T_e,B=b}(T_e > t_i) P_{X|T_e>t_i,B=b}(\mathbf{x}_i)\right] \\
&= \prod_{i\in I_C}\left[\sum_{b\in\mathcal{B}}\left(\sum_{t_e=t_i+1}^{\mathsf{T}_e}\pi_{t_e,b}\right)\left(\frac{\sum_{t_e=t_i+1}^{\mathsf{T}_e}\pi_{t_e,b}\pi_{\mathbf{x}_i|t_e,b}}{\sum_{t_e=t_i+1}^{\mathsf{T}_e}\pi_{t_e,b}}\right)\right] \\
&= \prod_{i\in I_C}\left[\sum_{b\in\mathcal{B}}\left(\sum_{t_e=t_i+1}^{\mathsf{T}_e}\pi_{t_e,b}\pi_{\mathbf{x}_i|t_e,b}\right)\right] \\
&= \prod_{i\in I_C}\left[\sum_{t_e=t_i+1}^{\mathsf{T}_e}\left(\sum_{b\in\mathcal{B}}\pi_{t_e,b}\pi_{\mathbf{x}_i|t_e,b}\right)\right]
\end{aligned}$$

The total likelihood is the product of the uncensored and censored likelihoods, and so the corresponding log likelihood takes the form

$$\begin{aligned}
L(\phi) &= \log\left(l_C(\phi) l_U(\phi)\right) \\
&= \log\left(\prod_{i\in I_C} P_{X,T_e}(\mathbf{x}_i, T_e > t_i)\right) \;+\; \log\left(\prod_{i\in I_U} P_{X,T_e,B}(\mathbf{x}_i, t_i, b_i)\right) \\
&= \log\left(\prod_{i\in I_C}\left[\sum_{t_e=t_i+1}^{\mathsf{T}_e}\left(\sum_{b\in\mathcal{B}}\pi_{t_e,b}\pi_{\mathbf{x}_i|t_e,b}\right)\right]\right) \;+\; \log\left(\prod_{i\in I_U}\pi_{t_i,b_i}\pi_{\mathbf{x}_i|t_i,b_i}\right) \\
&= \sum_{i\in I_C}\log\left(\sum_{t_e=t_i+1}^{\mathsf{T}_e}\left(\sum_{b\in\mathcal{B}}\pi_{t_e,b}\pi_{\mathbf{x}_i|t_e,b}\right)\right) \;+\; \sum_{i\in I_U}\log\left(\pi_{t_i,b_i}\pi_{\mathbf{x}_i|t_i,b_i}\right)
\end{aligned} \tag{28}$$

The standard ML method would employ an optimization algorithm to find the parameters $\left\{\pi_{t_e,b}, \pi_{\boldsymbol{\xi}|t_e,b}\right\}$ that maximize L subject to constraints such as $0 \le \pi_{t_e,b} \le 1$, $0 \le \pi_{\boldsymbol{\xi}|t_e,b} \le 1$, $\sum_{t_e}\sum_b \pi_{t_e,b} = 1$, and $\sum_{\boldsymbol{\xi}\in\mathcal{X}}\pi_{\boldsymbol{\xi}|t_e,b} = 1$. But maximizing the log of a sum is a complicated nonlinear optimization problem. The EM algorithm in the next section gives a way to mitigate this complexity, possibly at the price of an increase in estimation error due to the additional hidden variables that are introduced.

10.2 The EM Algorithm

The EM approach treats the (t_e, b) values of censored samples as *missing data* whose values are estimated as part of the learning algorithm (in addition to the estimation of the model parameters). The EM algorithm iterates the following two steps:

1. estimate the missing values with the model parameters fixed
2. solve for the model parameters with the missing values fixed

Under some very general conditions this algorithm is known to converge to a ML solution.

To employ the EM algorithm we must derive an expression for the expected value of the joint log likelihood with respect to the conditional distribution of missing data given the observed data

and the current parameter values ϕ_k. We start by constructing the conditional distribution of the missing random variables given the observed data and the current model parameters ϕ_k.

Let $m = |I_C|$ and $n = |I_U|$ be the number of censored and uncensored samples respectively. Without loss of generality let $I_C = \{1, 2, ..., m\}$ and $I_U = \{m+1, m+2, ..., n\}$. Then the *missing* data are

$$\big((t_{e_1}, b_1), (t_{e_2}, b_2), ..., (t_{e_m}, b_m)\big)$$

Let

$$\big((T_{e_1}, B_1), (T_{e_2}, B_2), ..., (T_{e_m}, B_m)\big)$$

be the random variable corresponding to the m censored samples. Since the samples are iid we denote this random variable by $(T_e, B)^m$. Similarly the *observed* censored samples are

$$\big((\mathbf{x}_1, t_{c_1}), (\mathbf{x}_2, t_{c_2}), ..., (\mathbf{x}_m, t_{c_m})\big)$$

which we write as $(\mathbf{x}, t_c)^m$. Let

$$\big((X_1, T_{c_1}), (X_2, T_{c_2}), ..., (X_m, T_{c_m})\big)$$

be the random variable corresponding to these m observations. Since the samples are iid we denote this random variable by $(X, T_c)^m$. Similarly the *observed* uncensored samples are

$$\big((\mathbf{x}_{m+1}, t_{e_{m+1}}, b_{m+1}), (\mathbf{x}_{m+2}, t_{c_{m+2}}, b_{m+2}), ..., (\mathbf{x}_{m+n}, t_{c_{m+n}}, b_{m+n})\big)$$

which we write as $(\mathbf{x}, t_e, b)^n$. Let

$$\big((X_{m+1}, T_{e_{m+1}}, B_{m+1}), (X_{m+2}, T_{e_{m+2}}, B_{m+2}), ..., (X_{m+n}, T_{e_{m+n}}, B_{m+n})\big)$$

be the random variable corresponding to these m observations. Since the samples are iid we denote this random variable by $(X, T_e, B)^n$. Now, the distribution of the missing random variables conditioned on the observed data and the current model parameters ϕ_k is

$$P_{(T_e,B)^m|(X,T_c)^m=(\mathbf{x},t_c)^m,(X,T_e,B)^n=(\mathbf{x},t_e,b)^n,\phi_k}$$

Using our short-hand notation this distribution can be written

$$P_{(T_e,B)^m|(\mathbf{x},t_c)^m,(\mathbf{x},t_e,b)^n,\phi_k}$$

To employ the EM algorithm we must derive an expression for the expected value of the joint log likelihood with respect to this distribution, i.e. we seek an expression for

$$Q_{\phi_k}(\phi) = E_{(T_e,B)^m|(\mathbf{x},t_c)^m,(\mathbf{x},t_e,b)^n,\phi_k}\left[\log\left(\prod_{i=1}^{m+n} P_{X,T_e,B}(\mathbf{x}_i, t_{e_i}, b_i)\right)\right]$$

$Q_{\phi_k}(\cdot)$ is a function of ϕ that is expressed in terms of the *expected values of the missing random variables conditioned on the observed data and the current model parameters* ϕ_k. Now we develop an expression for the log likelihood. Note that the likelihood can be written

$$\prod_{i=1}^{m+n} P_{X,T_e,B}(\mathbf{x}_i, t_{e_i}, b_i) = \bar{l}_C(\phi) l_U(\phi)$$

where

$$\bar{l}_C(\phi) = \prod_{i=1}^{m} P_{X,T_e,B}(\mathbf{x}_i, t_{e_i}, b_i)$$
$$l_U(\phi) = \prod_{i=m+1}^{m+n} P_{X,T_e,B}(\mathbf{x}_i, t_{e_i}, b_i)$$

Thus, Q_{ϕ_k} can be written

$$Q_{\phi_k}(\phi) = E_{(T_e,B)^m|(\mathbf{x},t_c)^m,(\mathbf{x},t_e,b)^n,\phi_k}\left[\log\left(\bar{l}_C(\phi) l_U(\phi)\right)\right]$$

Using (27) the uncensored likelihood l_U can be written

$$l_U(\phi) = \prod_{i=m+1}^{m+n} P_{X,T_e,B}(\mathbf{x}_i, t_i, b_i) = \prod_{i\in I_U} P_{X,T_e,B}(\mathbf{x}_i, t_{e_i}, b_i) = \prod_{i\in I_U} \pi_{t_{e_i},b_i}\pi_{\mathbf{x}_i|t_{e_i},b_i}$$

where we have replaced t_i with t_{e_i} for uncensored samples to help distinguish between event and censor times in this derivation. Similarly, $\bar{l}_C$ is given by

$$\bar{l}_C(\phi) = \prod_{i=1}^{m} P_{X,T_e,B}(\mathbf{x}_i, t_{e_i}, b_i) = \prod_{i\in I_C} \pi_{t_{e_i},b_i}\pi_{\mathbf{x}_i|t_{e_i},b_i}$$

Thus, the log likelihood is

$$\begin{aligned}\log\left(\bar{l}_C(\phi) l_U(\phi)\right) &= \log(\bar{l}_C(\phi)) + \log(l_U(\phi)) \\ &= \sum_{i\in I_C} \log\left(\pi_{t_{e_i},b_i}\pi_{\mathbf{x}_i|t_{e_i},b_i}\right) \;+\; \sum_{i\in I_U} \log\left(\pi_{t_{e_i},b_i}\pi_{\mathbf{x}_i|t_{e_i},b_i}\right)\end{aligned}$$

For each censored sample $i \in I_C$ define the $|\mathcal{T}\times\mathcal{B}|$ dimensional *ground truth vector*

$$z_i = \begin{bmatrix} 0 \\ . \\ 1 \\ . \\ 0 \end{bmatrix}$$

which has a 1 in the ground truth position and a 0 in all other positions, i.e. the $(t_e, b)^{th}$ position of z_i is given by

$$z_{i,t_e,b} = \begin{cases} 1, & t_i = t_e, b_i = b \\ 0, & t_i \neq t_e \end{cases} \qquad \text{for } i \in I_C$$

Then we can write

$$\log\left(\pi_{t_{e_i},b_i}\pi_{\mathbf{x}_i|t_{e_i},b_i}\right) = \sum_{t_e=1}^{\mathsf{T}_e}\sum_{b\in\mathcal{B}} z_{i,t_e,b}\log\left(\pi_{t_e,b}\pi_{\mathbf{x}_i|t_e,b}\right)$$

so that the log likelihood becomes

$$\log\left(\bar{l}_C(\phi) l_U(\phi)\right) = \sum_{i\in I_C}\sum_{t_e=1}^{\mathsf{T}_e}\sum_{b\in\mathcal{B}} z_{i,t_e,b}\log\left(\pi_{t_e,b}\pi_{\mathbf{x}_i|t_e,b}\right) + \sum_{i\in I_U}\log\left(\pi_{t_{e_i},b_i}\pi_{\mathbf{x}_i|t_{e_i},b_i}\right)$$

Note that we have replaced the missing values (t_{e_i}, b_i) for each censored sample with the missing vector z_i[16]. Now we form the function Q_{ϕ_k} by taking the expected value,

$$\begin{aligned}
Q_{\phi_k}(\phi) &= E_{(T_e,B)^m|(\mathbf{x},t_c)^m,(\mathbf{x},t_e,b)^n,\phi_k}\left[\log\left(\bar{l}_C(\phi)l_U(\phi)\right)\right] \\
&= E_{(T_e,B)^m|(\mathbf{x},t_c)^m,(\mathbf{x},t_e,b)^n,\phi_k}\left[\sum_{i\in I_C}\sum_{t_e=1}^{\mathsf{T}_e}\sum_{b\in\mathcal{B}} z_{i,t_e,b}\log\left(\pi_{t_e,b}\pi_{\mathbf{x}_i|t_e,b}\right)\right] \;+\; \sum_{i\in I_U}\log\left(\pi_{t_{e_i},b_i}\pi_{\mathbf{x}_i|t_{e_i},b_i}\right) \\
&= \sum_{i\in I_C}\sum_{t_e=1}^{\mathsf{T}_e}\sum_{b\in\mathcal{B}} E_{(T_e,B)^m|(\mathbf{x},t_c)^m,(\mathbf{x},t_e,b)^n,\phi_k}\left[z_{i,t_e,b}\right]\log\left(\pi_{t_e,b}\pi_{\mathbf{x}_i|t_e,b}\right) \;+\; \sum_{i\in I_U}\log\left(\pi_{t_{e_i},b_i}\pi_{\mathbf{x}_i|t_{e_i},b_i}\right) \\
&= \sum_{i\in I_C}\sum_{t_e=1}^{\mathsf{T}_e}\sum_{b\in\mathcal{B}} \bar{z}_{i,t_e,b}\log\left(\pi_{t_e,b}\pi_{\mathbf{x}_i|t_e,b}\right) \;+\; \sum_{i\in I_U}\log\left(\pi_{t_{e_i},b_i}\pi_{\mathbf{x}_i|t_{e_i},b_i}\right)
\end{aligned}$$

where

$$\bar{z}_{i,t_e,b} = E_{(T_e,B)^m|(\mathbf{x},t_c)^m,(\mathbf{x},t_e,b)^n,\phi_k}\left[z_{i,t_e,b}\right] = \begin{cases} 0, & t_e \le t_{c_i} \\ P_{T_{e_i},B_i|(\mathbf{x}_i,t_{c_i}),\phi_k}(t_e,b), & t_e > t_{c_i} \end{cases}$$

and[17]

$$\begin{aligned}
P_{T_{e_i},B_i|(\mathbf{x}_i,t_{c_i}),\phi_k}(t_e,b) &= \frac{P_{X|t_e,b,\phi_k}(\mathbf{x}_i)P_{T_e,B,\phi_k}(t_e,b)}{\sum_{v>t_{c_i}}\sum_{\beta\in\mathcal{B}}P_{X|v,\beta,\phi_k}(\mathbf{x}_i)P_{T_e,\beta,\phi_k}(v,\beta)} \\
&= \frac{\pi_{t_e,b}\pi_{\mathbf{x}_i|t_e,b}}{\sum_{v>t_{c_i}}\sum_{\beta\in\mathcal{B}}\pi_{v,\beta}\pi_{\mathbf{x}_i|v,\beta}}
\end{aligned}$$

Now that we have $Q_{\phi_k}(\phi)$ the EM algorithm works as follows:

E–Step: Form $Q_{\phi_k}(\phi)$. With ϕ fixed, and $(\mathbf{x}, t_c)^m, (\mathbf{x}, t_e, b)^n$ given, compute the expected value of the missing variables $\bar{z}_{i,y,b}$

$$\bar{z}_{i,t_e,b} = \begin{cases} 0, & t_e \le t_{c_i} \\ \frac{\pi_{t_e,b}\pi_{\mathbf{x}_i|t_e,b}}{\sum_{v>t_{c_i}}\sum_{\beta\in\mathcal{B}}\pi_{v,\beta}\pi_{\mathbf{x}_i|v,\beta}}, & t_e > t_{c_i} \end{cases}, \qquad \forall\; i\in I_C,\; (t_e,b)\in\mathcal{T}\times\mathcal{B} \tag{29}$$

M–Step: Find ϕ^* that maximizes $Q_{\phi_k}(\phi)$ with respect to ϕ, subject to constraints

$$\sum_{t_e=1}^{\mathsf{T}_e}\sum_{b\in\mathcal{B}}\pi_{t_e,b} = 1$$

$$\sum_{\boldsymbol{\xi}\in\mathcal{X}}\pi_{\boldsymbol{\xi}|t_e,b} = 1 \text{ for each } (t_e,b)\in\mathcal{T}\times\mathcal{B}$$

We now derive a solution to the optimization problem in the M–Step. For the uncensored samples we define the index set $I_{t_e,b}$ as follows

$$I_{t_e,b} = \{i : i\in I_U \text{ and } t_{e_i} = t_e \text{ and } b_i = b\}, \qquad (t_e,b)\in\mathcal{T}\times\mathcal{B} \tag{30}$$

[16]This is a key step in the EM derivation. It means that the expected value now operates on z_i instead of t_{e_i}, b_i.
[17]Using $P_{A|B,C} = P_{B|A,C}P_{A,C}/P_B$.

and we define $n_{t_e,b} = |I_{t_e,b}|$ to be the number of samples in each of these index sets. Now, the objective function can be written

$$\begin{aligned}
Q_{\phi_k}(\phi) &= \sum_{i\in I_C}\sum_{t_e=1}^{\mathsf{T}_e}\sum_{b\in\mathcal{B}} \bar{z}_{i,t_e,b}\log\left(\pi_{t_e,b}\pi_{\mathbf{x}_i|t_e,b}\right) \;+\; \sum_{i\in I_U}\log\left(\pi_{t_{e_i},b_i}\pi_{\mathbf{x}_i|t_{e_i},b_i}\right) \\
&= \sum_{i\in I_C}\sum_{t_e=1}^{\mathsf{T}_e}\sum_{b\in\mathcal{B}} \bar{z}_{i,t_e,b}\log\left(\pi_{t_e,b}\right) \;+\; \sum_{i\in I_C}\sum_{t_e=1}^{\mathsf{T}_e}\sum_{b\in\mathcal{B}} \bar{z}_{i,t_e,b}\log\left(\pi_{\mathbf{x}_i|t_e,b}\right) \;+\; \sum_{i\in I_U}\log\left(\pi_{t_{e_i},b_i}\right) + \sum_{i\in I_U}\log\left(\pi_{\mathbf{x}_i|t_{e_i},b_i}\right) \\
&= \sum_{t_e,b}\sum_{i\in I_C} \bar{z}_{i,t_e,b}\log\left(\pi_{t_e,b}\right) \;+\; \sum_{t_e,b}\sum_{i\in I_C} \bar{z}_{i,t_e,b}\log\left(\pi_{\mathbf{x}_i|t_e,b}\right) \;+\; \sum_{t_e,b}\sum_{i\in I_{t_e,b}}\log\left(\pi_{t_e,b}\right) + \sum_{t_e,b}\sum_{i\in I_{t_e,b}}\log\left(\pi_{\mathbf{x}_i|t_e,b}\right) \\
&= \sum_{t_e,b}\left(\sum_{i\in I_C} \bar{z}_{i,t_e,b}\log\left(\pi_{t_e,b}\right) + \sum_{i\in I_{t_e,b}}\log\left(\pi_{t_e,b}\right)\right) \;+\; \sum_{t_e,b}\left(\sum_{i\in I_C} \bar{z}_{i,t_e,b}\log\left(\pi_{\mathbf{x}_i|t_e,b}\right) \;+\; \sum_{i\in I_{t_e,b}}\log\left(\pi_{\mathbf{x}_i|t_e,b}\right)\right) \\
&= \sum_{t_e,b}\left(\sum_{i\in I_C} \bar{z}_{i,t_e,b}\log\left(\pi_{t_e,b}\right) + n_{t_e,b}\log\left(\pi_{t_e,b}\right)\right) \;+\; \sum_{t_e,b}\left(\sum_{i\in I_C} \bar{z}_{i,t_e,b}\log\left(\pi_{\mathbf{x}_i|t_e,b}\right) \;+\; \sum_{i\in I_{t_e,b}}\log\left(\pi_{\mathbf{x}_i|t_e,b}\right)\right)
\end{aligned}$$

Note the $\pi_{t_e,b}$ parameters are isolated to the first summation above and the $\pi_{\boldsymbol{\xi}|t_e,b}$ parameters are isolated to the second sum above.

First we solve for the $\pi_{t_e,b}$ parameters. These parameters are the solution to the constrained optimization problem

$$\begin{aligned}
&\max_{\{\pi_{t_e,b}\}} \sum_{t_e=1}^{\mathsf{T}_e}\sum_{b\in\mathcal{B}}\left(\sum_{i\in I_C} \bar{z}_{i,t_e,b}\log\left(\pi_{t_e,b}\right) + n_{t_e,b}\log\left(\pi_{t_e,b}\right)\right) \\
&\text{subject to } \sum_{t_e=1}^{\mathsf{T}_e}\sum_{b\in\mathcal{B}}\pi_{t_e,b} = 1
\end{aligned}$$

The Lagrangian is

$$l = \sum_{t_e=1}^{\mathsf{T}_e}\sum_{b\in\mathcal{B}}\left(\sum_{i\in I_C} \bar{z}_{i,t_e,b}\log\left(\pi_{t_e,b}\right) + n_{t_e,b}\log\left(\pi_{t_e,b}\right)\right) - \lambda\left(\sum_{t_e=1}^{\mathsf{T}_e}\sum_{b\in\mathcal{B}}\pi_{t_e,b} - 1\right)$$

Taking the derivatives wrt $\pi_{t_e,b}$ and λ, and setting to zero gives

$$\frac{\partial l}{\partial \pi_{t_e,b}} = \left(\sum_{i\in I_C}\frac{\bar{z}_{i,t_e,b}}{\pi_{t_e,b}} + \frac{n_{t_e,b}}{\pi_{t_e,b}}\right) - \lambda = 0$$

$$\frac{\partial l}{\partial \lambda} = \sum_{t_e=1}^{\mathsf{T}_e}\sum_{b\in\mathcal{B}}\pi_{t_e,b} - 1 = 0$$

Simplifying these two expressions yields

$$\frac{1}{\pi_{t_e,b}}\left(n_{t_e,b} + \sum_{i\in I_C}\bar{z}_{i,t_e,b}\right) = \lambda$$

$$\sum_{t_e=1}^{\mathsf{T}_e} \sum_{b \in \mathcal{B}} \pi_{t_e,b} = 1$$

Solving the first for $\pi_{t_e,b}$ and substituting into the second gives

$$\frac{1}{\lambda} \sum_{t_e=1}^{\mathsf{T}_e} \sum_{b \in \mathcal{B}} \left(n_{t_e,b} + \sum_{i \in I_C} \bar{z}_{i,t_e,b} \right) = 1$$

or equivalently

$$\begin{aligned}
\lambda &= \sum_{t_e=1}^{\mathsf{T}_e} \sum_{b \in \mathcal{B}} n_{t_e,b} + \sum_{t_e=1}^{\mathsf{T}_e} \sum_{b \in \mathcal{B}} \sum_{i \in I_C} \bar{z}_{i,t_e,b} \\
&= n + \sum_{i \in I_C} \sum_{t_e=1}^{\mathsf{T}_e} \sum_{b \in \mathcal{B}} \bar{z}_{i,t_e,b} \\
&= n + \sum_{i \in I_C} 1 \\
&= n + m \\
&= N
\end{aligned}$$

where the third line follows directly from the expression for $\bar{z}_{i,t_e,b}$ in (29) above. Now solving the first for $\pi_{t_e,b}$ and substituting $\lambda = n + m$ gives

$$\pi_{t_e,b} = \frac{n_{t_e,b} + \sum_{i \in I_C} \bar{z}_{i,t_e,b}}{N} \tag{31}$$

Next we solve for the parameters $\pi_{\boldsymbol{\xi}|t_e,b}$. These parameters are the solution to the constrained optimization problem

$$\begin{aligned}
\max_{\{\pi_{\boldsymbol{\xi}|t_e,b}\}} & \sum_{t_e=1}^{\mathsf{T}_e} \sum_{b \in \mathcal{B}} \left(\sum_{i \in I_C} \bar{z}_{i,t_e,b} \log\left(\pi_{\mathbf{x}_i|t_e,b}\right) \;+\; \sum_{i \in I_{t_e,b}} \log\left(\pi_{\mathbf{x}_i|t_e,b}\right) \right) \\
& \text{subject to } \sum_{\boldsymbol{\xi} \in \mathcal{X}} \pi_{\boldsymbol{\xi}|t_e,b} = 1, \qquad \text{for every } (t_e, b) \in \mathcal{T} \times \mathcal{B}
\end{aligned}$$

Note that the optimization for each value of (t_e, b) can be performed separately. Thus, for each $(t_e, b) \in \mathcal{T} \times \mathcal{B}$ we want to solve the optimization problem

$$\begin{aligned}
\max_{\pi_{\boldsymbol{\xi}|t_e,b}} & \left(\sum_{i \in I_C} \bar{z}_{i,t_e,b} \log\left(\pi_{\mathbf{x}_i|t_e,b}\right) \;+\; \sum_{i \in I_{t_e,b}} \log\left(\pi_{\mathbf{x}_i|t_e,b}\right) \right) \\
& \text{subject to } \sum_{\boldsymbol{\xi} \in \mathcal{X}} \pi_{\boldsymbol{\xi}|t_e,b} = 1
\end{aligned}$$

This criterion can be written

$$\begin{aligned} criterion &= \sum_{i \in I_C} \bar{z}_{i,t_e,b} \log\left(\pi_{\mathbf{x}_i|t_e,b}\right) \;+\; \sum_{i \in I_{t_e,b}} \log\left(\pi_{\mathbf{x}_i|t_e,b}\right) \\ &= \sum_{\boldsymbol{\xi} \in \mathcal{X}} \left(\sum_{i: i \in I_C, \mathbf{x}_i = \boldsymbol{\xi}} \bar{z}_{i,t_e,b} \log\left(\pi_{\mathbf{x}_i|t_e,b}\right) \right) \;+\; \sum_{\boldsymbol{\xi} \in \mathcal{X}} \left(\sum_{i: i \in I_{t_e,b}, \mathbf{x}_i = \boldsymbol{\xi}} \log\left(\pi_{\mathbf{x}_i|t_e,b}\right) \right) \\ &= \sum_{\boldsymbol{\xi} \in \mathcal{X}} \left(\sum_{i: i \in I_C, \mathbf{x}_i = \boldsymbol{\xi}} \bar{z}_{i,t_e,b} \log\left(\pi_{\mathbf{x}_i|t_e,b}\right) \;+\; \sum_{i: i \in I_{t_e,b}, \mathbf{x}_i = \boldsymbol{\xi}} \log\left(\pi_{\mathbf{x}_i|t_e,b}\right) \right) \end{aligned}$$

This can be simplified as follows

$$criterion = \sum_{\boldsymbol{\xi} \in \mathcal{X}} \chi_{\boldsymbol{\xi}}(t_e, b) \log\left(\pi_{\boldsymbol{\xi}|t_e,b}\right)$$

where

$$\begin{aligned} \chi_{\boldsymbol{\xi}}(t_e, b) &= \left(\sum_{i: i \in I_C, \mathbf{x}_i = \boldsymbol{\xi}} \bar{z}_{i,t_e,b} \;+\; \sum_{i: i \in I_{t_e,b}, \mathbf{x}_i = \boldsymbol{\xi}} 1 \right) \\ &= \left(\sum_{i: i \in I_C, \mathbf{x}_i = \boldsymbol{\xi}} \bar{z}_{i,t_e,b} \right) \;+\; n_{t_e,b,\boldsymbol{\xi}} \end{aligned}$$

with

$$n_{t_e,b,\boldsymbol{\xi}} = |\{i : i \in I_{t_e,b}, \mathbf{x}_i = \boldsymbol{\xi}\}|$$

Thus, the Lagrangian for this problem is

$$l(\pi_{\boldsymbol{\xi}|t_e,b}, \lambda) = \left(\sum_{\boldsymbol{\xi} \in \mathcal{X}} \chi_{\boldsymbol{\xi}}(t_e, b) \log\left(\pi_{\boldsymbol{\xi}|t_e,b}\right) \right) - \lambda \left(\sum_{\boldsymbol{\xi} \in \mathcal{X}} \pi_{\boldsymbol{\xi}|t_e,b} - 1 \right)$$

Taking the gradient and setting to zero yields

$$\frac{\partial l}{\partial \pi_{\boldsymbol{\xi}|t_e,b}} = \frac{\chi_{\boldsymbol{\xi}}(t_e, b)}{\pi_{\boldsymbol{\xi}|t_e,b}} - \lambda = 0$$

$$\frac{\partial l}{\partial \lambda} = \sum_{\boldsymbol{\xi} \in \mathcal{X}} \pi_{\boldsymbol{\xi}|t_e,b} - 1 = 0$$

Solving the first equation for $\pi_{\boldsymbol{\xi}|t_e,b}$ gives

$$\pi_{\boldsymbol{\xi}|t_e,b} = \frac{\chi_{\boldsymbol{\xi}}(t_e, b)}{\lambda}$$

and substituting this into the second equation gives

$$\sum_{\boldsymbol{\xi} \in \mathcal{X}} \frac{\chi_{\boldsymbol{\xi}}(t_e, b)}{\lambda} = 1.$$

Solving for λ gives

$$\lambda = \sum_{\boldsymbol{\xi} \in \mathcal{X}} \chi_{\boldsymbol{\xi}}(t_e, b)$$

and therefore

$$\pi_{\boldsymbol{\xi}|t_e,b} = \frac{\chi_{\boldsymbol{\xi}}(t_e, b)}{\sum_{\boldsymbol{\xi}' \in \mathcal{X}} \chi_{\boldsymbol{\xi}'}(t_e, b)} \tag{32}$$

Equations (31) and (32) provide the closed form solution $\phi^* = \{\pi_{t_e,b}, \pi_{\boldsymbol{\xi}|t_e,b}\}$ for the optimization problem in the M–Step.

The complete EM algorithm for computing ML estimates of the probability distribution parameters $\{\pi_{t_e,b}, \pi_{\boldsymbol{\xi}|t_e,b}\}$ is shown in Algorithm 1. This algorithm terminates when a *stopping condition* is met. There are many way of specifying this stopping condition. The following is a common approach. It terminates the algorithm when the changes in both the log-likelihood criterion value and the parameter values are sufficiently small. Let L be the log-likelihood criterion in (28). At each iteration k compute

$$\theta_1^k = \frac{L(\phi_{k+1}) - L(\phi_k)}{1.0 + L(\phi_{k+1})}$$

$$\theta_2^k = \left(\frac{\|\phi^{k-1} - \phi^k\|}{1.0 + \|\phi^k\|} \right)^2$$

and take the max

$$\theta^k = \max(\theta_1^k, \theta_2^k).$$

Stop when

$$\theta^k < \tau$$

where τ is on the order of the number of significant digits.

Note that in addition to estimating the parameters ϕ, the EM algorithm also estimates the probability of each event type (*grad* and *drop*) at each event time for each censored data sample. Indeed each $\bar{z}_{i,t_e,b}$ is an estimate of the probability that the true (event time, event type) for uncensored sample i is equal to (t_e, b).

10.3 The Weighted EM Algorithm

In this section we develop the *Weighted EM* algorithm, i.e. an EM algorithm for weighted data of the form $(\kappa, \mathbf{x}, t, c, b, w)$. As in the previous section, the stage label κ will not be used here. Similar to the previous section a weighted censored data sample will take the form $(\mathbf{x}, t, w)$ where $t = t_c$ is a censored time, and a weighted uncensored data sample will take the form $(\mathbf{x}, t, b, w)$ where $t = t_e$ is an event time.

In a weighted data collection we assume that the weights are proportional to the *empirical* probability mass of a sample value. For example in a weighted data collection with a total of $\acute{n}$ samples, the sample $(\mathbf{x}_i, t_i, b_i, w_i)$ has an empirical probability mass of $\frac{w_i}{\sum_{i=1}^{\acute{n}} w_i}$. Furthermore, if this collection was derived from an unweighted collection in the manner describe in Section 5.1.5 then $\acute{n} = N$ and $\sum_{i=1}^{\acute{n}} w_i = N$ and the empirical probability mass of $(\mathbf{x}_i, t_i, b_i, w_i)$ becomes $\frac{w_i}{N}$. In contrast, every sample in the unweighted data collection has an empirical probability mass of $1/N$. These weighted and unweighted data collections provide different representations of the same empirical distribution. The weighted EM algorithm derived in this section is valid for *any weighted data set where the weights form a legitimate representation of an empirical distribution.*

Algorithm 1 EM Algorithm for computing ML Estimates of the probability distribution parameters $\phi = \{\pi_{t_e,b}, \pi_{\boldsymbol{\xi}|t_e,b}\}$

Inputs: censored samples $(\mathbf{x}_i, t_{c_i}),\ i \in I_C = \{1, 2, ..., m\}$
uncensored samples $(\mathbf{x}_i, t_{e_i}, b_i),\ i \in I_U = \cup_{(t_e,b)} I_{t_e,b}$
and T_e

Assign initial values to the parameters $\{\pi_{t_e,b}, \pi_{\boldsymbol{\xi}|t_e,b}\}$. The initial values must satisfy
$\sum_{t_e=1}^{\mathsf{T}_e} \sum_{b \in \mathcal{B}} \pi_{t_e,b} = 1$ and $\sum_{\boldsymbol{\xi} \in \mathcal{X}} \pi_{\boldsymbol{\xi}|t_e,b} = 1$ for each $(t_e, b) = \{1, 2, ..., \mathsf{T}_e\} \times \{\mathsf{G}, \mathsf{D}\}$

for $(\boldsymbol{\xi} \in \mathcal{X})$ **do**
$n_{t_e,b,\boldsymbol{\xi}} = |\{i : i \in I_{t_e,b}, \mathbf{x}_i = \boldsymbol{\xi}\}|$
end for

repeat

{E-Step:}
for $(i = 1, 2, ..., m)$ **do**
for $(b = \mathsf{G}, \mathsf{D})$ **do**
for $(t_e = 1, 2, ..., \mathsf{T}_e)$ **do**

$$\bar{z}_{i,t_e,b} = \begin{cases} 0, & t_e \leq t_{c_i} \\ \frac{\pi_{t_e,b} \pi_{\mathbf{x}_i|t_e,b}}{\sum_{v > t_{c_i}} \sum_{\beta=0}^{1} \pi_{v,\beta} \pi_{\mathbf{x}_i|v,\beta}}, & t_e > t_{c_i} \end{cases}$$

end for
end for
end for

{M-Step:}
for $(t_e = 1, 2, ..., \mathsf{T}_e)$ **do**
for $(b = \mathsf{G}, \mathsf{D})$ **do**
$\pi_{t_e,b} \leftarrow \frac{n_{t_e,b} + \sum_{i \in I_C} \bar{z}_{i,t_e,b}}{N}$
for $(\boldsymbol{\xi} \in \mathcal{X})$ **do**
$\chi_{\boldsymbol{\xi}}(t_e, b) = n_{t_e,b,\boldsymbol{\xi}} \ + \ \sum_{i: i \in I_C, \mathbf{x}_i = \boldsymbol{\xi}} \bar{z}_{i,t_e,b}$
end for
for $(\boldsymbol{\xi} \in \mathcal{X})$ **do**
$\pi_{\boldsymbol{\xi}|t_e,b} = \frac{\chi_{\boldsymbol{\xi}}(t_e,b)}{\sum_{\boldsymbol{\xi}' \in \mathcal{X}} \chi_{\boldsymbol{\xi}'}(t_e,b)}$
end for
end for
end for

until (stopping condition is met)

`Return`$(\{\pi_{t_e,b}, \pi_{\boldsymbol{\xi}|t_e,b}\})$

Recall that the likelihood function for unweighted data is the product of probabilities evaluated at the samples. Thus, the contribution of a sample value $(\mathbf{x}, t, b)$ that is repeated w times is $P_{X,T_e,B}(\mathbf{x}, t, b)^w$. Similarly, a weighted sample $(\mathbf{x}, t, b, w)$ that appears once would contribute $P_{X,T_e,B}(\mathbf{x}, t, b)^w$. Thus, the uncensored and censored likelihoods now take the form

$$l_U(\phi) = \prod_{i \in I_U} P_{X,T_e,B}^{w_i}(\mathbf{x}_i, t_{e_i}, b_i)$$

$$l_C(\phi) = \prod_{i \in I_C} P_{X,T_e}^{w_i}(\mathbf{x}_i, t_e > t_{c_i})$$

Then, after applying the exact same steps as in Section 10.1, the log-likelihood takes the form

$$L(\phi) = \sum_{i \in I_C} w_i \log \left(\sum_{t_e = t_{c_i}+1}^{\mathsf{T}_e} \left(\sum_{b \in \mathcal{B}} \pi_{t_e,b} \pi_{\mathbf{x}_i | t_e, b} \right) \right) \; + \; \sum_{i \in I_U} w_i \log \left(\pi_{t_{e_i}, b_i} \pi_{\mathbf{x}_i | t_{e_i}, b_i} \right) \tag{33}$$

The development of the EM algorithm proceeds along the same steps as in Section 10.2. In particular, the missing data likelihood takes the form

$$\bar{l}_C(\phi) = \prod_{i \in I_C} P_{X,T_e,B}^{w_i}(\mathbf{x}_i, t_{e_i}, b_i)$$

and the log likelihood is

$$\begin{aligned} \log \left(\bar{l}_C(\phi) l_U(\phi) \right) &= \log(\bar{l}_C(\phi)) + \log(l_U(\phi)) \\ &= \sum_{i \in I_C} w_i \log \left(\pi_{t_{e_i}, b_i} \pi_{\mathbf{x}_i | t_{e_i}, b_i} \right) \; + \; \sum_{i \in I_U} w_i \log \left(\pi_{t_{e_i}, b_i} \pi_{\mathbf{x}_i | t_{e_i}, b_i} \right) \end{aligned}$$

Then, with the same definition of $z_{i,t_e,b}$ as before, the log-likelihood can be written

$$\log \left(\bar{l}_C(\phi) l_U(\phi) \right) = \sum_{i \in I_C} \sum_{t_e=1}^{\mathsf{T}_e} \sum_{b \in \mathcal{B}} w_i z_{i,t_e,b} \log \left(\pi_{t_e,b} \pi_{\mathbf{x}_i | t_e, b} \right) + \sum_{i \in I_U} w_i \log \left(\pi_{t_{e_i}, b_i} \pi_{\mathbf{x}_i | t_{e_i}, b_i} \right)$$

The E–Step derivation ends the same exact result for $\bar{z}_{i,t_e,b}$. The M–Step derivation yields

$$\pi_{t_e,b} = \frac{W_{t_e,b} + \sum_{i \in I_C} w_i \bar{z}_{i,t_e,b}}{W_C + W_U} \tag{34}$$

where

$$W_{t_e,b} = \sum_{i \in I_{t_e,b}} w_i$$

$$W_C = \sum_{i \in I_C} w_i$$

$$W_U = \sum_{i \in I_U} w_i$$

and

$$\pi_{\boldsymbol{\xi} | t_e, b} = \frac{\chi_{\boldsymbol{\xi}}(t_e, b)}{\sum_{\boldsymbol{\xi}' \in \mathcal{X}} \chi_{\boldsymbol{\xi}'}(t_e, b)}$$

where

$$\chi_{\boldsymbol{\xi}}(t_e, b) = \left(\sum_{i:i \in I_C, \mathbf{x}_i = \boldsymbol{\xi}} w_i \bar{z}_{i,t_e,b} \right) + W_{t_e,b,\boldsymbol{\xi}}$$

and

$$W_{t_e,b,\boldsymbol{\xi}} = \sum_{i:i \in I_{t_e,b}, \mathbf{x}_i = \boldsymbol{\xi}} w_i$$

The complete weighted EM algorithm is shown in Algorithm 2.

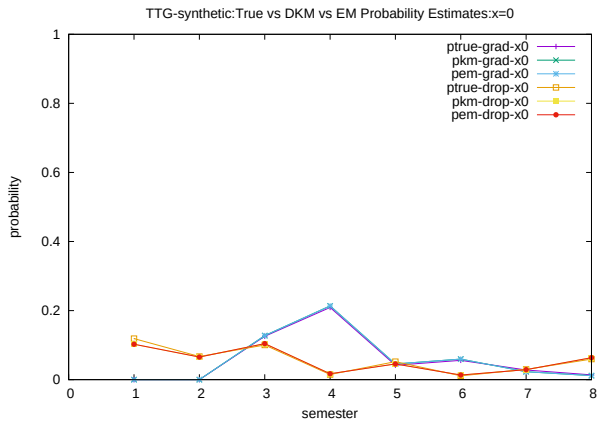
10.4 Example with Synthetic Data

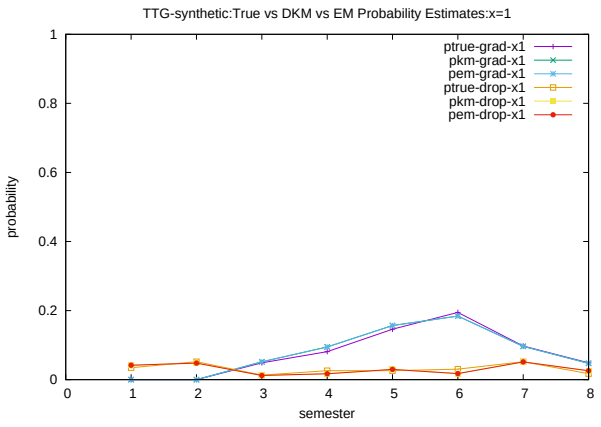
This example applies the semi-supervised approach with the EM algorithm in Section 10.2 to a collection of synthetic data. We compare this approach with the DKM *Distinct Covariate Groups* method in Section 8. The synthetic data was generated according to **Sample Plan 5** with the following parameters.

- The largest possible event time is $\mathsf{T}_e = 10$.
- The drop delay is $K = 2$.
- The NSS option is used to determine the event time.
- The covariate vector $\mathbf{x}$ has only one binary-valued component. This is the simplest nontrivial choice for a synthetic covariate.
- The data contains four cohorts with 1000 samples each. Each cohort starts 2 semesters after the previous cohort.

To mitigate the drop delay bias the *sample reassignment* method was applied to the synthesized data. Thus, with $\mathsf{T}_e = 10$ and $K = 2$ the largest time in the reassigned data is $t = 8$.

The true prediction functions, along with estimates from the EM and DKM methods, are shown in the six plots below. The first two plots show the probability function estimates for $\mathbf{x} = 0$ and $\mathbf{x} = 1$. The EM and DKM estimates are labeled *pem* and *pkm* respectively. It is difficult to differentiate between these two estimates because they are essentially identical. In addition, they are good estimates of the true probability function. The same observations hold for the hazard and cessation function estimates in the next four plots. Note that the EM and DKM hazard estimates are labeled *hem* and *hkm* respectively, and the EM and DKM cessation estimates are labeled *cem* and *ckm* respectively.





Algorithm 2 Weighted EM Algorithm for computing ML Estimates of the probability distribution parameters $\phi = \{\pi_{t_e,b}, \pi_{\boldsymbol{\xi}|t_e,b}\}$

Inputs: censored samples $(\mathbf{x}_i, t_{c_i}, w_i),\ i \in I_C = \{1, 2, ..., m\}$
uncensored samples $(\mathbf{x}_i, t_{e_i}, b_i, w_i),\ i \in I_U = \cup_{(t_e,b)} I_{t_e,b}$
and T_e

Assign initial values to the parameters $\{\pi_{t_e,b}, \pi_{\boldsymbol{\xi}|t_e,b}\}$. The initial values must satisfy $\sum_{t_e=1}^{\mathsf{T}_e} \sum_{b \in \mathcal{B}} \pi_{t_e,b} = 1$ and $\sum_{\boldsymbol{\xi} \in \mathcal{X}} \pi_{\boldsymbol{\xi}|t_e,b} = 1$ for each $(t_e, b) = \{1, 2, ..., \mathsf{T}_e\} \times \{\mathsf{G}, \mathsf{D}\}$

for $((t_e, b) \in \mathcal{T} \times \mathcal{B})$ **do**
  $W_{t_e,b} = \sum_{i \in I_{t_e,b}} w_i$
  for $(\boldsymbol{\xi} \in \mathcal{X})$ **do**
    $W_{t_e,b,\boldsymbol{\xi}} = \sum_{i: i \in I_{t_e,b}, \mathbf{x}_i = \boldsymbol{\xi}} w_i$
  end for
end for

$W_C = \sum_{i \in I_C} w_i, \quad W_U = \sum_{i \in I_U} w_i$
repeat

  {E-Step:}
  for $(i = 1, 2, ..., m)$ **do**
    for $(b = \mathsf{G}, \mathsf{D})$ **do**
      for $(t_e = 1, 2, ..., \mathsf{T}_e)$ **do**

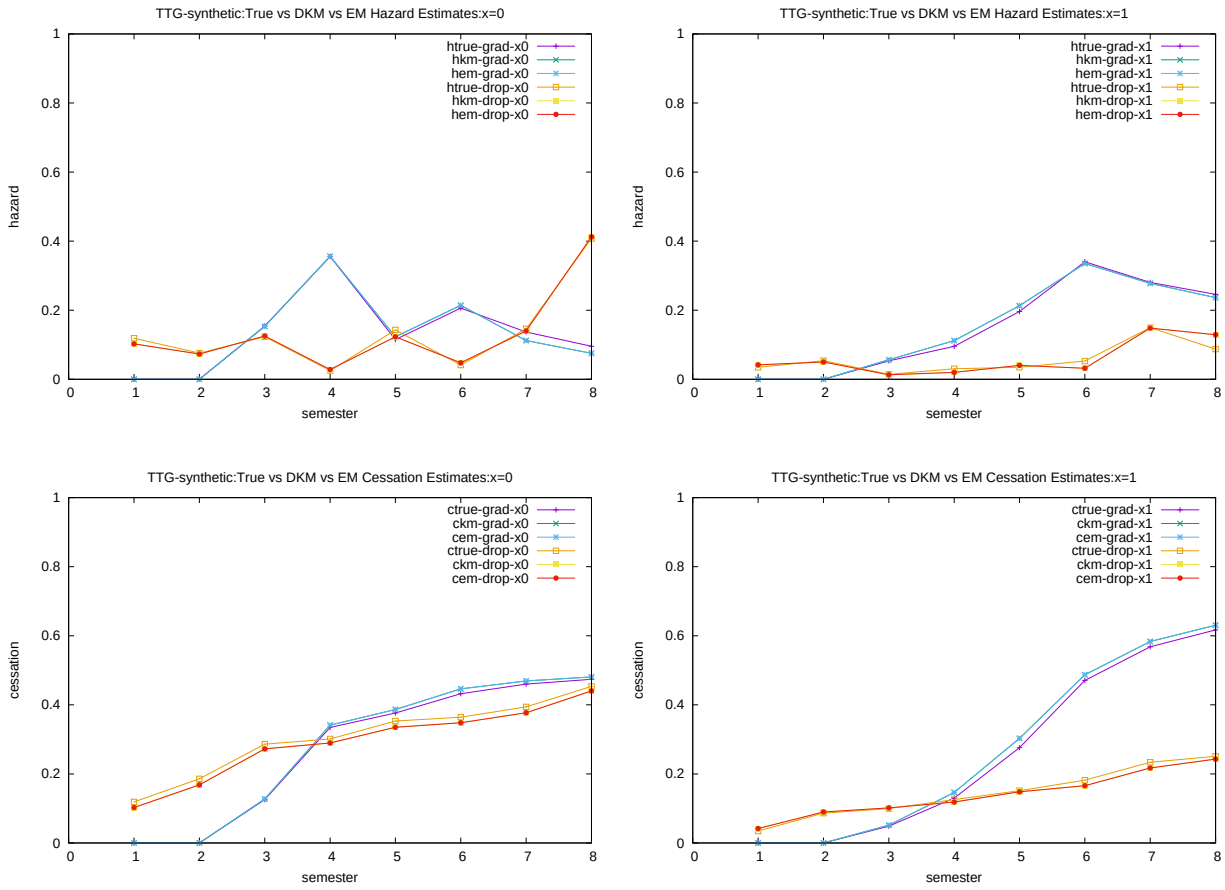
$$\bar{z}_{i,t_e,b} = \begin{cases} 0, & t_e \le t_{c_i} \\ \frac{\pi_{t_e,b} \pi_{\mathbf{x}_i|t_e,b}}{\sum_{v > t_{c_i}} \sum_{\beta=0}^{1} \pi_{v,\beta} \pi_{\mathbf{x}_i|v,\beta}}, & t_e > t_{c_i} \end{cases}$$

      end for
    end for
  end for

  {M-Step:}
  for $(t_e = 1, 2, ..., \mathsf{T}_e)$ **do**
    for $(b = \mathsf{G}, \mathsf{D})$ **do**
      $\pi_{t_e,b} \leftarrow \frac{W_{t_e,b} + \sum_{i \in I_C} w_i \bar{z}_{i,t_e,b}}{W_C + W_U}$
      for $(\boldsymbol{\xi} \in \mathcal{X})$ **do**
        $\chi_{\boldsymbol{\xi}}(t_e, b) = W_{t_e,b,\boldsymbol{\xi}} \ + \ \sum_{i: i \in I_C, \mathbf{x}_i = \boldsymbol{\xi}} w_i \bar{z}_{i,t_e,b}$
      end for
      for $(\boldsymbol{\xi} \in \mathcal{X})$ **do**
        $\pi_{\boldsymbol{\xi}|t_e,b} = \frac{\chi_{\boldsymbol{\xi}}(t_e,b)}{\sum_{\boldsymbol{\xi}' \in \mathcal{X}} \chi_{\boldsymbol{\xi}'}(t_e,b)}$
      end for
    end for
  end for

until (stopping condition is met)

`Return`$(\{\pi_{t_e,b}, \pi_{\boldsymbol{\xi}|t_e,b}\})$



10.5 TTG at UNM as a function of (GENDER, RESIDENCY)

This section uses semi-supervised and DKM methods to produce prediction function estimates for UNM students as a function GENDER and RESIDENCY. Both GENDER and RESIDENCY are binary valued covariates that are coded as follows.

- GENDER takes the values Female (F) and Male (M)
- RESIDENCY takes the values Resident (R) and Nonresident (N)

The data consists of students from *UNM FTFT fall cohorts 2012-2017*. The number of students in each of the four covariate value groups is summarized in the table below.

Covariate Value	Number of Students
Female Resident	9630
Female NonResident	1684
Male Resident	7171
Male NonResident	1537

The data is processed using the following parameters.

- Event times are determined using the NSS option.
- Drop labels are estimated using $K = 2$ semesters.

The corresponding staged data table is shown below.

```
-------------------------------------------------------------------------------
UNM FTFT Data (Option NSS,K=2)
Cohort Fall 2012:  nsamples 3424
  ngrad:      0    0    0    3    4   41   46  639  338  377   93  125   37   56
  ndrop:    260  392  169  191  123   98   56   59   57   48   30   38    *    *
  ncensor:    0    0    0    0    0    0    0    0    0    0    0    0   28  116
Cohort Fall 2013:  nsamples 3518
  ngrad:      0    0    0    0    6   48   95  879  309  335   82  104    -    -
  ndrop:    293  347  187  189   87  102   60   56   47   48    *    *    -    -
  ncensor:    0    0    0    0    0    0    0    0    0    0   51  193    -    -
Cohort Fall 2014:  nsamples 3132
  ngrad:      0    0    0    1    1   63  116  882  216  270    -    -    -    -
  ndrop:    272  278  152  167  105   86   56   71    *    *    -    -    -    -
  ncensor:    0    0    0    0    0    0    0    0   65  331    -    -    -    -
Cohort Fall 2015:  nsamples 3327
  ngrad:      0    0    0    4    7   92  132  820    -    -    -    -    -    -
  ndrop:    258  342  204  178  130   96    *    *    -    -    -    -    -    -
  ncensor:    0    0    0    0    0    0  101  963    -    -    -    -    -    -
Cohort Fall 2016:  nsamples 3402
  ngrad:      0    0    0    9   13   45    -    -    -    -    -    -    -    -
  ndrop:    290  404  233  207    *    *    -    -    -    -    -    -    -    -
  ncensor:    0    0    0    0  154 2047    -    -    -    -    -    -    -    -
Cohort Fall 2017:  nsamples 3219
  ngrad:      0    0    1    7    -    -    -    -    -    -    -    -    -    -
  ndrop:    386  437    *    *    -    -    -    -    -    -    -    -    -    -
  ncensor:    0    0  239 2149    -    -    -    -    -    -    -    -    -    -

Semester      1    2    3    4    5    6    7    8    9   10   11   12   13   14
-------------------------------------------------------------------------------
```

Models are built using the following parameters.

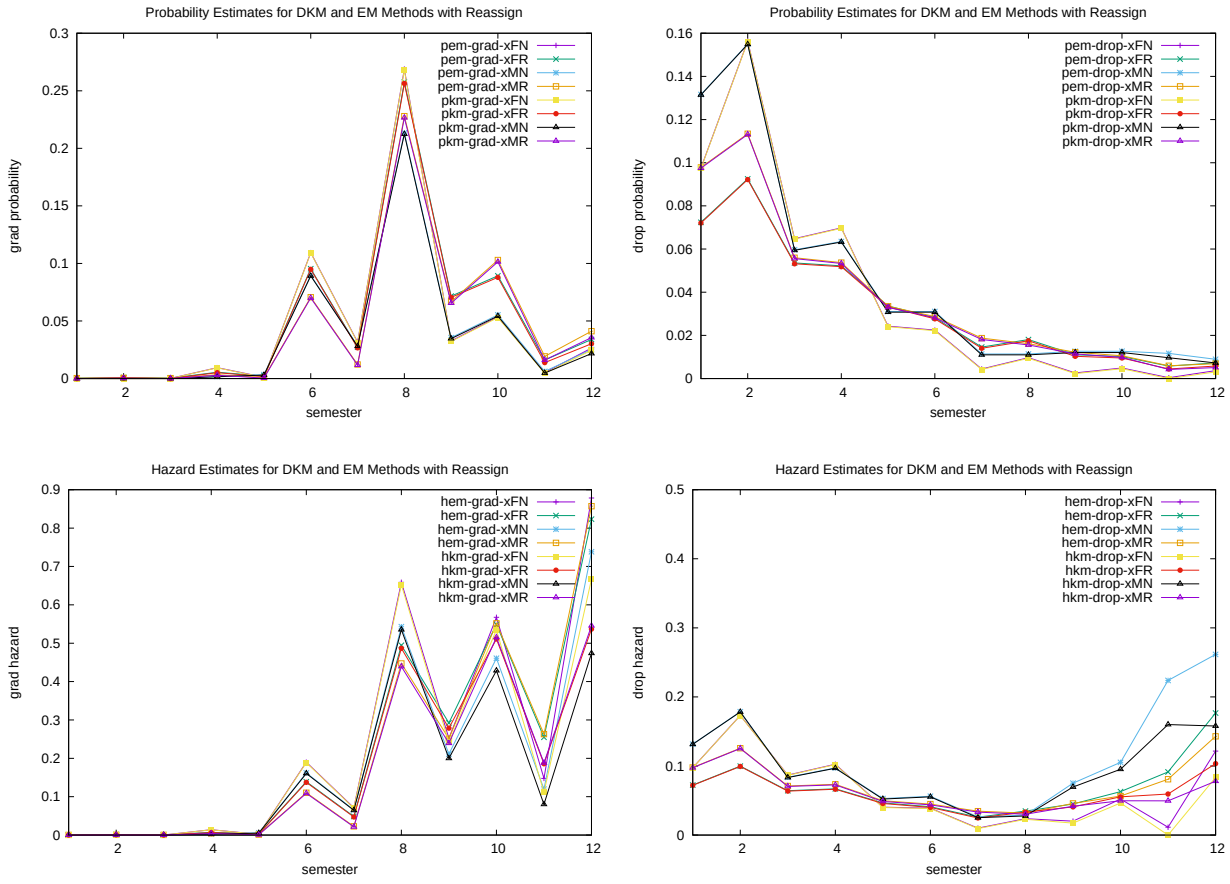
- Both the *reassign* (Section 10.5.1) and *weighted data* (WD) (Section 10.5.2) methods are used to mitigate the drop delay bias.
- DKM models are built using the *Distinct Covariate Groups* method, i.e. by applying the DKM method, or the weighted DKM method, separately to each of the four covariate value groups.
- EM models are built using either the the EM algorithm in Section 10.2 or the weighted EM algorithm in Section 10.3.
- Models are built for each of the four (GENDER, RESIDENCY) values. Each of the four models contains prediction function estimates for both the *grad* and *drop* event types.

The next section shows results for data that has been pre-processed using the *reassign* method.

10.5.1 Estimates using the Reassign Method

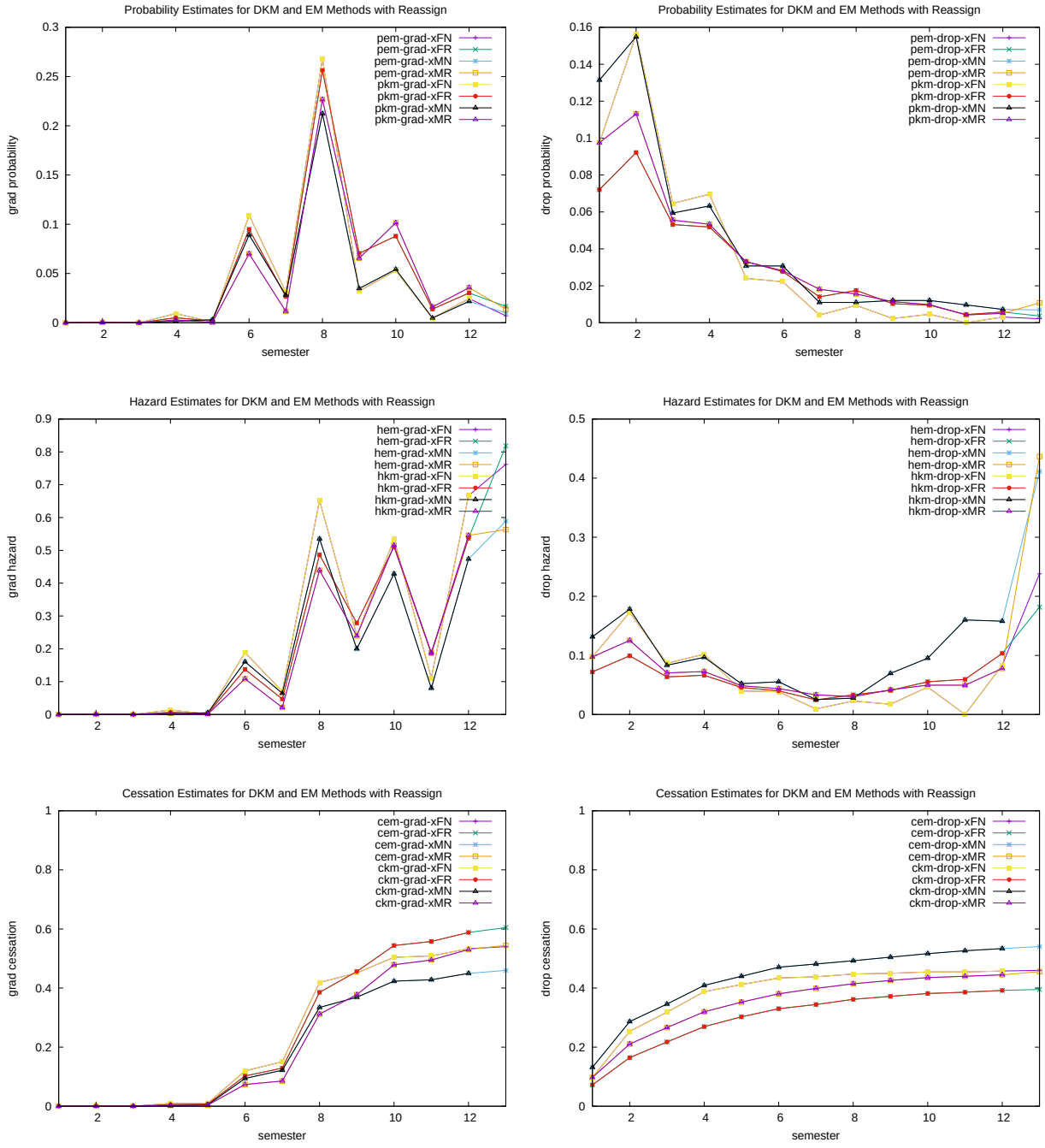
This section shows prediction function estimates obtained by applying the unweighted EM and DKM methods to data that has been pre-processed with the *reassign* method. The first two plots show the four *grad* probability functions on the left and the four *drop* probability functions on the right. The EM estimates are denoted by *pem* and the DKM estimates by *pkm*, and the covariate

values are denoted by F=Female, M-Male, R=Resident, N=Nonresident. Note that the EM and DKM estimates are similar, but not identical. Differences between these two estimates are more pronounced in hazard function estimates in the second two plots below, especially at the later semester values. The differences between the two methods can be explained by the fact that the EM algorithm always produces a *complete* model[18] within the specified number of time slots, while the DKM method does not.



However, the differences between the two methods can be mitigated by adding one extra time slot to the EM algorithm, i.e. adding a time slot at the end where there is no data. In this example the extra time slot corresponds to semester 13. This modification allows the EM algorithm to place excess probability mass in the extra slot so that it can produce a complete model, but have the exact same estimates as DKM in the first 12 time slots. The corresponding probability estimates are shown in the two plots below. Note that the EM and DKM estimates are identical for semesters 1-12. The EM probability estimates in time slot 13 are probably not meaningful extrapolations, they simply allow the algorithm to form complete model. Estimates of the corresponding hazard and cessation functions are shown in the next four plots below. The cessation plots indicate that the (Female, Resident) group has the highest final graduation rate and the (Male, Nonresident) group as the lowest final graduation rate (approximately 14% lower). These results also indicate that the final graduation rate is approximately the same for the (Female, Nonresident) and (Male, Resident) groups, but the four-year graduation rate is much higher for the (Female, Nonresident) group.

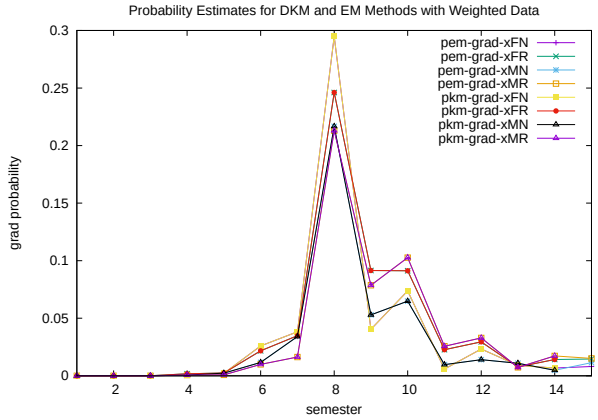
[18]Recall that a complete model is one where the probability functions integrate to 1.

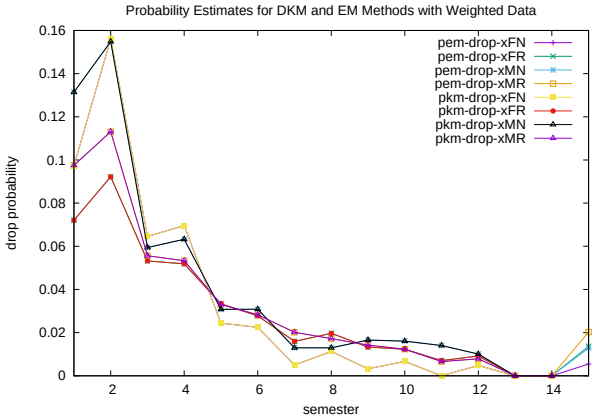


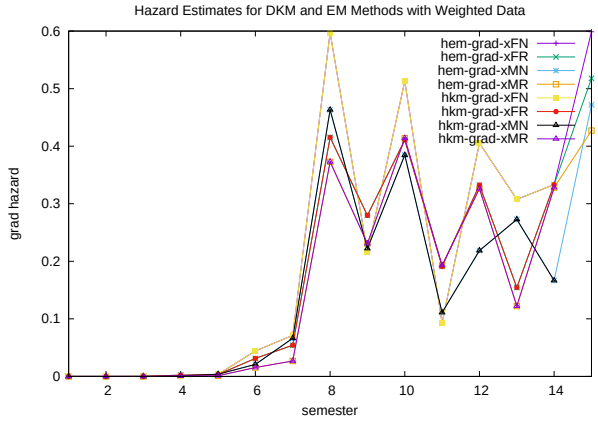
10.5.2 Estimates using the Weighted Data Method

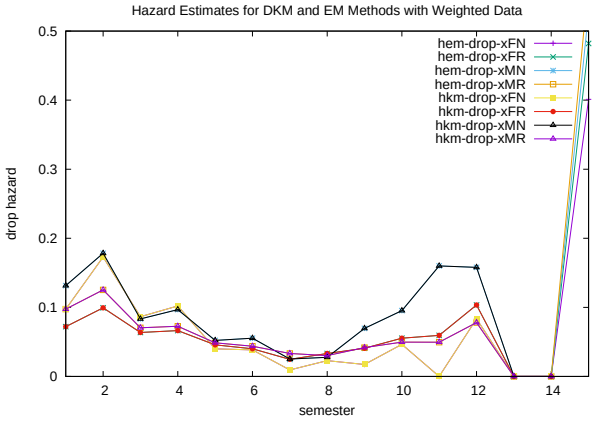
This section shows prediction function estimates obtained by applying the weighted EM and DKM methods to data that has been pre-processed with the *weighted data* method. The *weighted data* method allows predictions out to semester 14 (instead of 12), and we have added an extra time slot $t = 15$ for the EM algorithm as described in the previous section. The first two plots show the four *grad* probability functions on the left and the four *drop* probability functions on the right. The EM estimates are denoted by *pem* and the DKM estimates by *pkm*, and the covariate values are denoted by F=Female, M-Male, R=Resident, N=Nonresident. Once again the EM and DKM

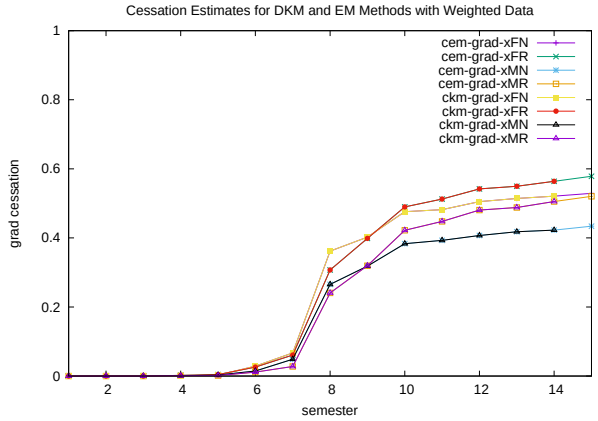
estimates are identical for semesters 1-14. The same is true for the estimates of the hazard and cessation functions in the next four plots below.

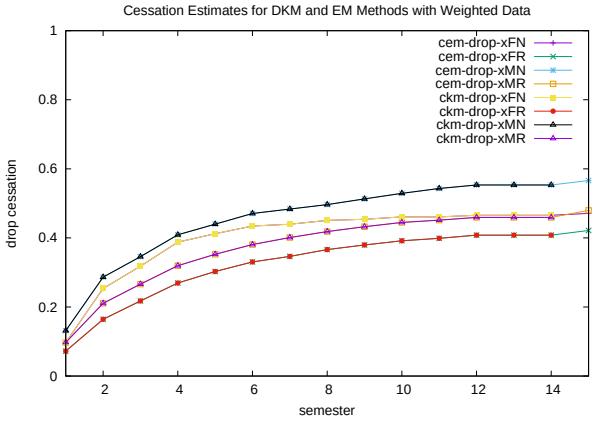












A Survival Analysis Literature Review

A good introduction to survival analysis can be found in [11], and an excellent description of its application to educational data analysis can be found in [24]. Singer and Willett also provide a much more detailed description in their book [25]. The use of survival analysis to study student graduation and retention in education is ubiquitous, e.g. see [2, 4, 7, 8, 9, 10, 13, 12, 14, 19, 20, 21, 22, 29]. Many of these papers treat survival analysis as if it is a turn-key method that is so well understood that a detailed description is not needed. But this can lead to a miss-match between the application and the solution method.

Perhaps the most common survival analysis problem is the clinical trial where the event of interest is *death*, and the goal is to make predictions about the survival time (i.e. time-to-death) for subjects under different procedures/treatments/drugs (hence the name *survival* analysis). This problem is so ubiquitous, and the medical community has such a stringent requirement for a unified approach to analysis, that this problem is nearly always solved using a turn-key method based on the continuous-time model in Cox's famous paper [5]. But the TTG problem differs from the clinical trial problem in at least two important ways.

First, in the clinical trial there is a single type of event that all subjects will eventually experience (e.g. death), but in the TTG problem *graduation* is not the only type of event that a student may experience. Instead the student may *drop* out of the institution. It turns out that if we wish to make accurate predictions about the time-to-graduate then we must include dropout events in our analysis. Problems with more than one type of event are generally referred to as *competing events* problems. Although there are prominent examples where a *single event type* model has been used to study student graduation or retention [7, 24, 25], it is generally recognized that a *competing events* model is more appropriate [2, 4, 23, 27]. Furthermore, dropout statistics are just as important as graduation statistics when it comes to monitoring student progress. Thus, in the TTG problem our goal is to make predictions about both the event type, i.e. *grad* or *drop*, and the time it takes to experience the event.

A second difference is that the clinical trial solution assumes that *time* takes on a continuous range of positive values, and that it is rare for multiple subjects to experience the exact same event time. In the TTG problem however, where time = number of semesters, the time value is clearly discrete (i.e. a finite positive integer), and it is extremely common for large numbers of subjects to experience the exact same event time. Indeed, in any given semester there are typically a large number of students that either graduate or drop. Problems like this require *discrete-time* survival analysis methods. Cox introduced a discrete-time method in his original paper, although his main focus was on a continuous-time proportional hazards model [5]. The use of discrete-time methods for educational data analysis is now well recognized [2, 4, 23, 27]. In the TTG problem we restrict our scope to *discrete-time* survival analysis methods. A more general treatment of discrete-time competing events survival analysis methods can be found in [1, 3].

There are numerous additional aspects of the TTG problem that require special attention. Several of these are mentioned in the excellent paper by (Scott and Kennedy [23]). Examples of the unique aspects that are addressed in the body of this report include drop delay bias, staged censoring, label error analysis, and incomplete prediction functions.

Finally we mention that there has been some interest in the application of *machine learning* methods to survival analysis problems. The survey in [28] provides one perspective. The emphasis of that work appears to be on continuous-time single event type problems, and so is not directly relevant to the TTG problem in this report.

B Notation

This report adopts the following notational conventions. Probability distributions will be denoted with a capital P. Subscripts will be used to identify the specific distribution, e.g. P_X is used to denote the probability distribution for the random variable X, and $P_{X|T}$ or $P_{X|T=t}$ is used to denote a conditional distribution. Furthermore we will use the common short-hand notation for distribution function arguments[19], i.e. we use the short-hand notation $P_T(t)$ to mean $P_T(\{t\})$. Similarly we use the short-hand notation $P_X(f(x) > 0)$ to mean $P_X(\{x : f(x) > 0\})$. In addition, for conditional distributions we use the shorthand notation $P_{X|t}$ to mean $P_{X|T=t}$.

C Synthesis Issues

This section addresses a number of miscellaneous issues related to the synthesis of TTG data. For example, we show the equivalence between Sample Plan 2 and the synthesis method described by (Scott and Kennedy [23]). We also discuss synthesis from an incomplete distribution, the impacts of the NSS versus NSE options, and the difference between *random* and *staged* censoring. We adopt a *covariate dependent* model throughout this section, but because the covariate samples $\mathbf{x}_i$ are synthesized separately it is easy to make direct comparisons to the covariate free Sample Plans 2, 3, and 4 in the main text.

C.1 Synthesis from a Hazard Model

Assume for the moment that there is no censoring. Then samples of the form $(\mathbf{x}, t_e, b)$ can be generated according to the distribution $P_{X,T_e,B}$. Since $P_{X,T_e,B} = P_{T_e,B|X}P_X$ these samples can be created by first generating $\mathbf{x}$ according to P_X, and then generating (t_e, b) according to $P_{T_e,B|\mathbf{x}}$. However, because of the dominant influence of the hazard function much of the survival literature emphasizes a different approach that generates data according to a hazard model $\mathsf{h}_{\mathbf{x}}(\cdot,\cdot)$, instead of $P_{T_e,B|\mathbf{x}}$. Now consider censoring. Given a collection of covariate samples $(\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_N)$ generated from P_X we want to generate a corresponding collection of observations $((t_1, c_1, b_1), (t_2, c_2, b_2), ..., (t_N, c_N, b_N))$ that are consistent with a hazard model $\mathsf{h}_{\mathbf{x}}(\cdot,\cdot)$ and a censor probability function p_c, where the function $p_c : \{1, 2, ..., \mathsf{T}\} \to [0, 1]$ specifies the probability $p_c(t)$ that an at-risk sample will be censored at time t. Since the hazard probability is conditioned on samples that survive to time t it would appear that the synthesis method must keep track of the samples that survive each time step (i.e. the at-risk samples). One such method is described in (Scott and Kennedy [23]). Their description is repeated here for convenience (recall that K is the number of event types, so that in the TTG problem $K = |\mathcal{B}| = 2$).

> "A common description of such a noninformative censoring process is that it is *independent of events*. If we articulate how we imagine our data to be generated, we can make this criterion precise. One possibility is the following. Imagine that, in each period, two experiments take place: a $(K + 1)$-tomous experiment deciding which event occurs to each subject, and a dichotomous experiment determining whether or not the student is censored (K = number of event types). The noncensored students who receive event 0 go to the next stage, where the two experiments occur again, independent of earlier stages. Under this data-generating mechanism, a criterion for censoring to be ignorable is that the censoring experiment and the event experiment are independent at every stage."

[19] Strictly speaking the argument of the distribution is a subset of the Borel set that corresponds to the pre-image of the domain subset. This paper adopts the standard notational convention where this relationship is implied.

This method can be implemented using an outer loop over t and an inner loop over the samples that remain at-risk at each time step as shown below.

Given covariates $\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_N$.
$I_{risk}(1) \leftarrow \{1, 2, ..., N\}$ {initialize index set for at-risk samples at initial time $t = 1$}
for $(t = 1, 2, 3, ...)$ **do**
  for (each data sample $i \in I_{risk}(t)$) **do**
    if (random binary sample with success probability $\mathsf{h}_{\mathbf{x}_i}(t)$ is successful) **then**
      $(t_i, c_i) \leftarrow (t, 0)$ {event is triggered}
    else if (random binary sample with success probability $p_c(t)$ is successful) **then**
      $(t_i, c_i) \leftarrow (t, 1)$ {censor is triggered}
    else
      insert sample i into $I_{risk}(t + 1)$
    end if
  end for
end for

We can eliminate the need to track at-risk samples at each time step by exchanging the inner and outer loops, i.e. by creating an outer loop over the N data samples and an inner loop that steps through the time values until an event or censor is triggered. A complete realization of this method is shown in Algorithm 3. The inputs to this algorithm are the (marginal) covariate distribution P_X, the hazard model $\mathsf{h}_{\mathbf{x}}$, the censor probability function p_c, and the largest time value T. Note that p_c is not necessarily a probability distribution function, i.e. its values do not necessarily sum to one. This algorithm starts by generating all N covariate values, and then uses the method just described to generate an observation (t_i, c_i) corresponding to each $\mathbf{x}_i$ value. The structure of this algorithm is essentially the same as one described in [26], but here it is adapted for the discrete-time problem and we add code to accommodate incomplete hazard models.

Lines 17-20 determine if an event is triggered at time t by drawing a sample b from a three-category *multinoulli* distribution F_3 with success probabilities $\mathsf{h}_{\mathbf{x}}(t, \mathsf{G}), \mathsf{h}_{\mathbf{x}}(t, \mathsf{D})$ and then triggering the event if $b > 0$. That is, $F_3(\rho_{\mathsf{G}}, \rho_{\mathsf{D}})$ takes the value $b = \mathsf{G}$ with probability ρ_{G}, $b = \mathsf{D}$ with probability ρ_{D}, and $b = 0$ with probability $1 - (\rho_{\mathsf{G}} + \rho_{\mathsf{D}})$. Similarly lines 23-26 determine if a censor is triggered at time t by drawing a sample c from a Bernoulli distribution F_2 with success probability $p_c(t)$ and then triggering the censor when $c = 1$. That is, $F_2(\rho)$ is equal to 1 with probability ρ and a 0 with probability $1 - \rho$. If the hazard function $\mathsf{h}_{\mathbf{x}}$ and the censor probability function p_c are independent then the event decision on line 18 and the censor decision on line 24 will be independent, and the independent censoring assumption will hold. To generate dependently censored samples we need only choose a censor probability function p_c that depends on $\mathsf{h}_{\mathbf{x}}$. Note that lines 22-27 prevent an observed event from being censored by requiring $b = 0$ before a sample is considered for censoring.

Lines 29-31 handle the special case where the hazard function is incomplete, i.e. when $\mathsf{h}_{\mathbf{x}}(\mathsf{T}) < 1$. Without lines 29-31 the loop over t could proceed past the last time value T. If $\mathsf{h}_{\mathbf{x}}$ is complete then $\mathsf{h}_{\mathbf{x}}(\mathsf{T}) = 1$ will guarantee that all samples that reach the last event time T will experience the event with probability 1 (in line 18), and lines 29-31 will be superfluous. But $\mathsf{h}_{\mathbf{x}}(\mathsf{T}) < 1$ suggests the existence of event times greater than T for which the hazard is unknown. In this case the samples that reach event time T but do not trigger an event in line 18 (or censor in line 24) are labeled as censored and assigned the observation time T.

Note that Algorithm 3 assigns event and censor times according to the NSS option. To implement

Algorithm 3 Synthesize TTG data according to a hazard model.

Inputs:
1. covariate distribution P_X
2. hazard functions $\mathsf{h}_{\mathbf{x}}$ for each $\mathbf{x} \in \mathcal{X}$
3. censor probability function p_c
4. largest time value T

{generate covariate data}
for ($i = 1$ to N) **do**
  $\mathbf{x}_i \leftarrow$ sample from distribution P_X
end for

{generate event and censor times}
for ($i = 1$ to n) **do**
  $t \leftarrow 1, \qquad trigger \leftarrow 0$
  while ($trigger = 0$) **do**
    {determine if the event occurs at the current time t for this sample}
    $b \leftarrow$ sample from multinoulli distribution $F_3(\mathsf{h}_{\mathbf{x}_i}(t, \mathsf{G}),\ \mathsf{h}_{\mathbf{x}_i}(t, \mathsf{D}))$
    if ($b \neq 0$) **then**
      $(t_i, c_i, b_i) \leftarrow (t, 0, b), \qquad trigger \leftarrow 1$
    end if
    {determine if this sample is censored ... but only if the event time has not occurred}
    if ($b = 0$) **then**
      $c \leftarrow$ sample from Bernoulli distribution $F_2(p_c(t))$
      if ($c \neq 0$) **then**
        $(t_i, c_i, b_i) \leftarrow (t, 1, b), \qquad trigger \leftarrow 1$
      end if
    end if
    {In case of incomplete hazard, force a censor at the last time}
    if (($b = 0$) AND ($c = 0$) AND ($t = \mathsf{T}$)) **then**
      $(t_i, c_i, b_i) \leftarrow (t, 1, b), \qquad trigger \leftarrow 1$
    end if
    $t \leftarrow t + 1$
  end while
end for
`Return`$(\{(\mathbf{x}_i, t_i, c_i, b_i)\})$

the NSE option we would need to synthesize an enrollment flag that determines the student's enrollment status at each time. Then we would need to keep track of the number of enrolled semesters in the loop. Implementation of the NSE option would require another stochastic model, possibly a state-space model such as a Markov model.

Note also that Algorithm 3 implements a random censor model. To implement a staged censor model lines 23-26 would be replaced by

if $(t = L)$ **then**
  $(t_i, c_i, b_i) \leftarrow (t, 1, b)$, $\quad trigger \leftarrow 1$
end if

where L is the observation length for the cohort to which $\mathbf{x}_i$ belongs.

C.2 Synthesis from a Distribution Model

Algorithm 3 uses a hazard model to determine the event time. The hazard function plays such a dominant role in survival analysis that this may seem natural, or even imperative. But it is not essential. In this section we describe a synthesis algorithm that uses a distribution model instead of a hazard model to determine the event time. We also generalize the censor mechanism in two ways. First we use a distribution model instead of an (unnormalized) probability model to determine the censor time, and second we compute a censor time for every sample and then set the output observation time to the smaller of the event and censor times. These changes have several advantages. First they allow a direct computation of the event and censor times without having to loop over time values until one of them is triggered. Second they provide more flexibility in the model choice and synthesis method. Third they make the choice of independent versus non-independent censoring more transparent. Fourth they make the unobserved event times for censored samples available for subsequent analysis.

Let T_e be the event time random variable taking values from $\{1, 2, ..., \mathsf{T}_e\}$, and T_c be the censor time random variable taking values from $\{1, 2, ..., \mathsf{T}_e - 1\}$. Let $P_{X,T_e,T_c,B}$ be a joint (covariate, event time, censor time, event type) distribution. Then to synthesize an observation $(\mathbf{x}, t, c, b)$ we first generate a sample $(\mathbf{x}, t_e, t_c, b_e)$ from $P_{X,T_e,T_c,B}$, and then set (e.g. see [17])

$$(\mathbf{x}, t, c, b) = \begin{cases} (\mathbf{x}, t_e, 0, b_e), & t_e \le t_c \\ (\mathbf{x}, t_c, 1, b_e), & t_e > t_c \end{cases} \tag{35}$$

Note that the observed time satisfies $t = \min(t_e, t_c)$. Also, if $c = 1$ then the true value of the event type b is not observed in practice even though it is produced as a part of this synthesis process. Note that if we were to modify Algorithm 3 by splitting the loop over t (lines 14-33) into two separate loops over t, one that computes an (event time, event type) (t_e, b_e) and another that computes a (censor time, default event type) (t_c, b_{e_*}), then the subsequent application of (35) would provide an equivalent result[20].

In the case of independent censoring we assume that the censor time is independent of the (covariate value, event time, event type) so that

$$P_{X,T_e,T_c,B} = P_{X,T_e,B} P_{T_c} \qquad \text{(independent censoring)}$$

Using $P_{X,T_e,B} = P_{T_e,B|X} P_X$ we obtain the decomposition

$$P_{X,T_e,T_c,B} = P_{T_e,B|X} P_X P_{T_c}$$

which is the foundation for the synthesis method in Algorithm 4.

[20] Of course this assumes that $P_{T_e,T_c,B|\mathbf{x}}$ and $h_{\mathbf{x}}$ are two different representations the same random process.

Algorithm 4 Synthesize TTG data according to a distribution model.

1: **Inputs:**
2: covariate distribution P_X
3: censor time distribution P_{T_c}
4: conditional distribution $P_{T_e,B|X}$
5:
6: **for** ($i = 1$ to N) **do**
7:
8: {generate (covariate, event time, censor time, event type) sample}
9: $\mathbf{x} \leftarrow$ generate a covariate sample according to P_X
10: $(t_e, b_e) \leftarrow$ generate an (event time, event type) sample according to $P_{T_e,B|\mathbf{x}}$
11: $t_c \leftarrow$ generate a censor time sample according to P_{T_c}
12:
13: {generate observed sample}
14: **if** ($t_e \leq t_c$) **then**
15: $(\mathbf{x}_i, t_i, c_i, b_i) \leftarrow (\mathbf{x}, t_e, 0, b_e)$
16: **else**
17: $(\mathbf{x}_i, t_i, c_i, b_i) \leftarrow (\mathbf{x}, t_c, 1, b_e)$
18: **end if**
19:
20: **end for**
21:
22: `Return`$(\{(\mathbf{x}_i, t_i, c_i, b_i)\})$

Note that Algorithm 4 implements a random censor model. To implement a staged censor model we would set $t_c \leftarrow L$ in line 11, where L is the observation length for the cohort to which $\mathbf{x}$ belongs.

Note also that Algorithm 4 assigns event and censor times according to the NSS option. To implement the NSE option we would need to synthesize an *enrollment vector* $\mathbf{e} = (e_1, e_2, ..., e_{\mathsf{T}_e})$ that represents the student's enrollment over all time. Then we would compute the event semester

$$s_e \;=\; (\text{largest value of } t \text{ where } e_t = 1) \;=\; \max\{t : e_t = 1\}$$

and the event and censor times

$$t_e = \textstyle\sum_{t=1}^{\mathsf{T}_e} e_t$$
$$t_c = \textstyle\sum_{t=1}^{L} e_t$$

and replace the test on line 14 with $s_e \leq L$. In the end lines 8-18 would be replaced with the following.

$\mathbf{x} \leftarrow$ generate a covariate sample according to P_X
$(\mathbf{e}, b_e) \leftarrow$ generate an (enrollment vector, event type) sample according to $P_{\mathbf{E},B|\mathbf{x}}$
$s_e \leftarrow \max\{t : e_t = 1\}$
$t_e = \sum_{t=1}^{\mathsf{T}_e} e_t$
$t_c = \sum_{t=1}^{L} e_t$

{generate observed sample}
if $(s_e \leq L)$ **then**
  $(\mathbf{x}_i, t_i, c_i, b_i) \leftarrow (\mathbf{x}, t_e, 0, b_e)$
else
  $(\mathbf{x}_i, t_i, c_i, b_i) \leftarrow (\mathbf{x}, t_c, 1, b_e)$
end if

D Proof of Theorem 1

Given the probability distribution of the observed samples $P_{T_o,C,B}$, the marginal distribution $P_{T_o,B}$ is given by

$$P_{T_o,B}(t,b) = \sum_{c=0}^{1} P_{T_o,C,B}(t,c,b)$$

and the marginal distribution P_{T_o} is given by

$$P_{T_o}(t) = \sum_{b\in\mathcal{B}} P_{T_o,B}(t,b)$$

The expressions for h and h_o in (8) and (9) are

$$h(t,b) = \frac{P_{T_e,B}(t,b)}{P_{T_e}(T_e \geq t)} \qquad h_o(t,b) = \frac{P_{T_o,C,B}(t,0,b)}{P_{T_o}(T_o \geq t)}$$

Our task is to prove that if censoring is independent then

$$\frac{P_{T_o,C,B}(t,0,b)}{P_{T_o}(T_o \geq t)} = \frac{P_{T_e,B}(t,b)}{P_{T_e}(T_e \geq t)}$$

This will be accomplished by proving that the distribution of observed uncensored samples satisfies

$$P_{T_o,C,B}(t,0,b) = P_{T_e,B}(t,b)P_{T_c}(T_c \geq t) \tag{36}$$

and the complementary cumulative distribution of observed times satisfies

$$P_{T_o}(T_o \geq t) = P_{T_e}(T_e \geq t)P_{T_c}(T_c \geq t) \tag{37}$$

so that substituting (36) and (37) into the expression for h_o will will produce the expression for h.

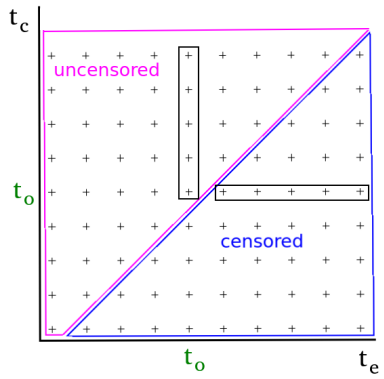
This proof is divided into two parts. The first part derives key relations between the probability functions of observed and unobserved times when conditioned on a fixed value of b, and the second

part uses these relations to prove the general case. The first part uses the following simplified notation

p_{T_o}	$= P_{T_o\|B=b}$	$=$	distribution of observed time for fixed b
$p_{T_{oe}}$	$= P_{T_o,C=0\|B=b}$	$=$	distribution of uncensored observed time for fixed b
$p_{T_{oc}}$	$= P_{T_o,C=1\|B=b}$	$=$	distribution of censored observed time for fixed b
p_{T_e,T_c}	$= P_{T_e,T_c\|B=b}$	$=$	joint distribution of unobserved (event, censor) times for fixed b
p_{T_e}	$= P_{T_e\|B=b}$	$=$	distribution of unobserved event time for fixed b

The goal for the first part is to develop expressions for the observed time functions $p_{T_{oe}}$ and $p_{T_o}(T_o \geq t)$ in terms of the unobserved time functions p_{T_e} and P_{T_c} under the independent censoring assumption.

We begin by developing an expression for the distribution of the observed time random variable when b is fixed. The diagram below shows a grid of (t_e, t_c) values. The (t_e, t_c) pairs that lead to censored observation times are surrounded in blue, and the pairs that lead to uncensored observation times are surrounded in magenta. Consider a particular observation time t_o as shown in the diagram. The event and censor time values (t_e, t_c) that map to $t_o = \min(t_e, t_c)$ are highlighted in the black rectangles.



Thus, the value of $p_{T_o}(t_o)$ is given by two sums, one sum over the samples in each of the two black rectangles. In general the distribution of T_o is given by the following two sums

$$p_{T_o}(t) = \sum_{\tau=t+1}^{\mathsf{T}_e} p_{T_e,T_c}(\tau, t) + \sum_{\tau=t}^{\mathsf{T}_e-1} p_{T_e,T_c}(t, \tau)$$

Let T_{o_c} be the random variable corresponding to the observed time for censored samples (e.g. samples in the lower right portion of the diagram above). Then the distribution of T_{o_c} is given by the first sum[21]

$$p_{T_{oc}}(t) = \sum_{\tau=t+1}^{\mathsf{T}_e} p_{T_e,T_c}(\tau, t), \qquad t = 1, 2, ..., \mathsf{T}_e - 1$$

Let T_{o_e} be the random variable corresponding to the observed time for uncensored samples (e.g. samples in the upper left portion of the diagram above). Then the distribution of T_{o_e} is given by the second sum

$$p_{T_{oe}}(t) = \sum_{\tau=t}^{\mathsf{T}_e-1} p_{T_e,T_c}(t, \tau), \qquad t = 1, 2, ..., \mathsf{T}_e$$

[21] Recall the the distribution of censored times is only defined for times up to $\mathsf{T}_e - 1$.

Our next step is to develop expressions for $p_{T_{o_e}}(t)$ and $p_{T_o}(T_o \geq t)$ under the independent censoring assumption. First we note that

$$\begin{aligned} p_{T_o}(T_o \geq t) &= \sum_{t'=t}^{\mathsf{T}_e} p_{T_o}(t') \\ &= \sum_{t'=t}^{\mathsf{T}_e} p_{T_{o_e}}(t') + \sum_{t'=t}^{\mathsf{T}_e-1} p_{T_{o_c}}(t') \\ &= p_{T_{o_e}}(T_{o_e} \geq t) + p_{T_{o_c}}(T_{o_c} \geq t) \end{aligned} \tag{38}$$

Now, if censoring is independent then[22]

$$p_{T_{o_e}}(t) = \sum_{\tau=t}^{\mathsf{T}_e-1} p_{T_e,T_c}(t,\tau) = \sum_{\tau=t}^{\mathsf{T}_e-1} p_{T_e}(t)P_{T_c}(\tau) = p_{T_e}(t) \sum_{\tau=t}^{\mathsf{T}_e-1} P_{T_c}(\tau) = p_{T_e}(t)P_{T_c}(T_c \geq t) \tag{39}$$

$$p_{T_{o_c}}(t) = \sum_{\tau=t+1}^{\mathsf{T}_e} p_{T_e,T_c}(\tau,t) = \sum_{\tau=t+1}^{\mathsf{T}_e} p_{T_e}(\tau)P_{T_c}(t) = P_{T_c}(t) \sum_{\tau=t+1}^{\mathsf{T}_e} p_{T_e}(\tau) \tag{40}$$

and therefore

$$p_{T_{o_e}}(T_{o_e} \geq t) = \sum_{t'=t}^{\mathsf{T}_e} p_{T_{o_e}}(t') = \sum_{t'=t}^{\mathsf{T}_e} \left(p_{T_e}(t') \sum_{\tau=t'}^{\mathsf{T}_e-1} P_{T_c}(\tau) \right) = \sum_{t'=t}^{\mathsf{T}_e} \sum_{\tau=t'}^{\mathsf{T}_e-1} p_{T_e}(t')P_{T_c}(\tau) \tag{41}$$

$$p_{T_{o_c}}(T_{o_c} \geq t) = \sum_{t'=t}^{\mathsf{T}_e-1} p_{T_{o_c}}(t') = \sum_{t'=t}^{\mathsf{T}_e-1} \left(P_{T_c}(t') \sum_{\tau=t'+1}^{\mathsf{T}_e} p_{T_e}(\tau) \right) = \sum_{t'=t}^{\mathsf{T}_e-1} \sum_{\tau=t'+1}^{\mathsf{T}_e} P_{T_c}(t')p_{T_e}(\tau) \tag{42}$$

Swapping the dummy variables τ and t' in (41) gives

$$p_{T_{o_e}}(T_{o_e} \geq t) = \sum_{\tau=t}^{\mathsf{T}_e} \sum_{t'=\tau}^{\mathsf{T}_e-1} p_{T_e}(\tau)P_{T_c}(t') \tag{43}$$

[22]Note that if censoring is independent then $P_{T_e,T_c|B=b} = P_{T_e|B=b}P_{T_c}$, which takes the form $p_{T_e,T_c} = p_{T_e}P_{T_c}$. in our simplified notation.

The sums in (42) can be rearranged as follows

$$
\begin{aligned}
p_{T_{o_c}}(T_{o_c} \geq t) &= \sum_{t'=t}^{\mathsf{T}_e-1} \left(P_{T_c}(t') \sum_{\tau=t'+1}^{\mathsf{T}_e} p_{T_e}(\tau) \right) \\
&= \sum_{t'=t}^{\mathsf{T}_e-1} \sum_{\tau=t'+1}^{\mathsf{T}_e} P_{T_c}(t') p_{T_e}(\tau) \\
&= P_{T_c}(t) \left(p_{T_e}(t+1) + p_{T_e}(t+2) + ... + p_{T_e}(\mathsf{T}_e) \right) + \\
&\quad P_{T_c}(t+1) \left(p_{T_e}(t+2) + p_{T_e}(t+3) + ... + p_{T_e}(\mathsf{T}_e) \right) + \\
&\quad ... \\
&\quad P_{T_c}(\mathsf{T}_e - 1) p_{T_e}(\mathsf{T}_e) \qquad (44) \\
&= p_{T_e}(\mathsf{T}_e) \left(P_{T_c}(t) + P_{T_c}(t+1) + ... + P_{T_c}(\mathsf{T}_e - 1) \right) + \\
&\quad p_{T_e}(\mathsf{T}_e - 1) \left(P_{T_c}(t) + P_{T_c}(t+1) + ... + P_{T_c}(\mathsf{T}_e - 2) \right) + \\
&\quad ... \\
&\quad p_{T_e}(t+1) P_{T_c}(t) \\
&= \sum_{\tau=t+1}^{\mathsf{T}_e} \sum_{t'=t}^{\tau-1} p_{T_e}(\tau) P_{T_c}(t')
\end{aligned}
$$

Combining (43) and (44) gives

$$
\begin{aligned}
p_{T_{o_e}}(T_{o_e} \geq t) + p_{T_{o_c}}(T_{o_c} \geq t) &= \sum_{\tau=t}^{\mathsf{T}_e} \sum_{t'=\tau}^{\mathsf{T}_e-1} p_{T_e}(\tau) P_{T_c}(t') \; + \; \sum_{\tau=t+1}^{\mathsf{T}_e} \sum_{t'=t}^{\tau-1} p_{T_e}(\tau) P_{T_c}(t') \\
&= \left(p_{T_e}(t) \sum_{t'=t}^{\mathsf{T}_e-1} P_{T_c}(t') \right) + \sum_{\tau=t+1}^{\mathsf{T}_e} \sum_{t'=\tau}^{\mathsf{T}_e-1} p_{T_e}(\tau) P_{T_c}(t') \; + \; \sum_{\tau=t+1}^{\mathsf{T}_e} \sum_{t'=t}^{\tau-1} p_{T_e}(\tau) P_{T_c}(t') \\
&= \left(p_{T_e}(t) \sum_{t'=t}^{\mathsf{T}_e-1} P_{T_c}(t') \right) + \sum_{\tau=t+1}^{\mathsf{T}_e} \sum_{t'=t}^{\mathsf{T}_e-1} p_{T_e}(\tau) P_{T_c}(t') \\
&= \left(p_{T_e}(t) \sum_{t'=t}^{\mathsf{T}_e-1} P_{T_c}(t') \right) + \sum_{\tau=t+1}^{\mathsf{T}_e} p_{T_e}(\tau) \sum_{t'=t}^{\mathsf{T}_e-1} P_{T_c}(t') \\
&= \left(p_{T_e}(t) \sum_{t'=t}^{\mathsf{T}_e-1} P_{T_c}(t') \right) + \left(\sum_{\tau=t+1}^{\mathsf{T}_e} p_{T_e}(\tau) \right) \left(\sum_{t'=t}^{\mathsf{T}_e-1} P_{T_c}(t') \right) \\
&= \left(\sum_{\tau=t}^{\mathsf{T}_e} p_{T_e}(\tau) \right) \left(\sum_{t'=t}^{\mathsf{T}_e-1} P_{T_c}(t') \right) \\
&= p_{T_e}(T_e \geq t) P_{T_c}(T_c \geq t)
\end{aligned}
$$

Substituting this result into (38) gives

$$
p_{T_o}(T_o \geq t) = p_{T_{o_e}}(T_{o_e} \geq t) + p_{T_{o_c}}(T_{o_c} \geq t) = p_{T_e}(T_e \geq t) P_{T_c}(T_c \geq t) \qquad (45)
$$

This completes the first part of the proof. We start the second part by rewriting the key results from (39) and (45) in our original notation. To this end (39) and (45) become

$$P_{T_o,C=0|B=b}(t) = p_{T_{oe}}(t) = p_{T_e}(t)P_{T_c}(T_c \geq t) = P_{T_e|B=b}(t)P_{T_c}(T_c \geq t)$$

$$P_{T_o|B=b}(T_o \geq t) = p_{T_o}(T_o \geq t) = p_{T_e}(T_e \geq t)P_{T_c}(T_c \geq t) = P_{T_e|B=b}(T_e \geq t)P_{T_c}(T_c \geq t)$$

Now we multiply both results by $P_B(b)$ and use the relation $P_B(b)P_{T|B=b}(t) = P_{T,B}(t,b)$ to obtain

$$P_{T_o,C=0,B}(t,b) = P_{T_o,C,B}(t,0,b) = P_{T_e,B}(t,b)P_{T_c}(T_c \geq t) \tag{46}$$

$$P_{T_o,B}(T_o \geq t,b) = P_{T_e,B}(T_e \geq t,b)P_{T_c}(T_c \geq t) \tag{47}$$

The result in (46) proves the conjecture in (36), and the result in (47) can be used to prove the conjecture in (37) as follows

$$\begin{aligned} P_{T_o}(T_o \geq t) &= \sum_{b\in\mathcal{B}} P_{T_o,B}(T_o \geq t,b) \\ &= P_{T_c}(T_c \geq t)\sum_{b\in\mathcal{B}} P_{T_e,B}(T_e \geq t,b) \\ &= P_{T_c}(T_c \geq t)P_{T_e}(T_e \geq t) \end{aligned}$$

The proof is completed by substituting the results above into (9) to obtain

$$h_o(t,b) = \frac{P_{T_o,C,B}(t,0,b)}{P_{T_o}(T_o \geq t)} = \frac{P_{T_e,B}(t,b)P_{T_c}(T_c \geq t)}{P_{T_e}(T_e \geq t)P_{T_c}(T_c \geq t)} = \frac{P_{T_e,B}(t,b)}{P_{T_e}(T_e \geq t)} = h(t,b)$$

QED

References

[1] P.D. Allison. Discrete-time methods for the analysis of event histories. *Sociological Methodology*, 13:61–98, 1982.

[2] E. Arias and C. Dehon. The roads to success: Analyzing dropout and degree completion at university. Working paper ECARES 2011-025, Universite Libre de Bruxelles, 2011.

[3] M. Berger and M. Schmid. Semiparametric regression for discrete time-to-event data. *Statistical Modelling*, 18(3–4):322–345, 2018. "doi:10.1177/1471082X17748084".

[4] R. Clerici, A. Giraldo, and S. Meggiolaro. The determinants of academic outcomes in a competing risks approach: evidence from italy. *Studies in Higher Education*, 2014.

[5] D.R. Cox. Regression models and life tables (with discussion). *Journal of the Royal Statistical Society, Series B*, 34:187–220, 1972.

[6] M. Crowder. *Classical competing risks*. Chapman & Hall, New York, New York, 2001.

[7] R.C. Deike. *A Study of College Student Graduation Using Discrete Time Survival Analysis*. PhD thesis, The Pennsylvania State University, 2003.

[8] S.L. DesJardins, D. Ahlburg, and B. McCall. Studying the determinants of student stopout: Identifying "true" from spurious time-varying effects. 1994. Paper presented at the 34th Annual Forum of the Association for Institutional Research, New Orleans, LA.

[9] S.L. DesJardins, D. Ahlburg, and B. McCall. Simulating the longitudinal effects of changes in financial aid on student departure from college. *Journal of Human Resources*, 37(3):653–679, 2002.

[10] S.L. DesJardins, B. McCall, D. Ahlburg, and M. Moye. Adding a timing light to the "tool box". *Research in Higher Education*, 43(1):83–114, 2002.

[11] John Fox. Introduction to survival analysis. Lecture notes, Department of Sociology, McMaster University, 2014. `https://socialsciences.mcmaster.ca/jfox/Courses/soc761/survival-analysis.pdf`.

[12] T. Ishitani. A longitudinal approach to assessing attrition behavior among first-generation students: Time-varying effects of pre-college characteristics. *Research in Higher Education*, 44(4):433–449, 2003.

[13] T. Ishitani and S.L. DesJardins. A longitudinal investigation of dropout from college in the united states. *Journal of College Student Retention*, 4(2):173–201, 2002.

[14] T. Ishitani and K. Snider. Longitudinal effects of college preparation programs on college retention. *IR Applications: Using Advanced Tools, Techniques and Methodologies*, 9:1–10, 2006.

[15] J.D. Kalbfleisch and R.L. Prentice. *The statistical analysis of failure time data.* Wiley, New York, New York, 2002.

[16] E.L. Kaplan and P. Meier. Non parametric estimation from incomplete observations. *Journal of the American Statistical Association*, 53:457–481, 1958.

[17] T. Mackenzie and M. Abrahamowicz. Marginal and hazard ratio specific random data generation: Applications to semi-parametric bootstrapping. *Statistics and Computing*, 12:3:245–252, 2002.

[18] W.L. May and W.D. Johnson. A sas macro for constructing simultaneous confidence intervals for multinomial proportions. *Computer Methods and Programs in Biomedicine*, 53(3):153–162, 1997. DOI: 10.1016/s0169-2607(97)01809-9.

[19] P. Murtaugh, L. Burns, and J. Schuster. Predicting the retention of university students. *Research in Higher Education*, 40:355–371, 1999.

[20] G. Nicholls, H. Wolfe, M. Besterfield-Sacre, and L. Shuman. Predicting post-secondary educational outcomes with survival analysis. *ASEE Annual Conference and Exposition, Conference Proceedings*, pages 14.966.1–14.966.16, 2009.

[21] P.M. Radcliffe, Jr. R.L. Huesman, and J.P. Kellogg. Identifying students at risk: Utilizing survival analysis to study student athlete attrition. Technical report, University of Minnesota, 2006. retrieved at "https://hdl.handle.net/11299/159769".

[22] S. Ronco. Meandering ways: Studying student stopout with survival analysis. 1994. Paper presented at the 34th Annual Forum of the Association for Institutional Research, New Orleans, LA.

[23] M.A. Scott and B.B. Kennedy. Pitfalls in pathways: Some perspectives on competing risks event history analysis in education research. *Journal of Educational and Behavioral Statistics*, 30(4):413–442, Winter 2005.

[24] J.D. Singer and J.B. Willett. It's about time: Using discrete-time survival analysis to study duration and the timing of events. *American Educational Research Association and American Statistical Association*, 18(2):155–195, 1993.

[25] J.D. Singer and J.B. Willett. *Applied Longitudinal Data Analysis: Modeling Change and Event Occurrence*. Oxford University Press, New York, New York, 1st edition, 2003.

[26] M.-P. Sylvestre and M. Abrahamowicz. Comparison of algorithms to generate event times conditional on time-dependent covariates. *Statistics in Medicine*, 27:14:2618–2634, 2010.

[27] C.A. Vallejos and M.F.J. Steel. Bayesian survival modelling of university outcomes. *Statistics in Society: Series A*, 180(2):613–631, 2017.

[28] Ping Wang, Yan Li, and Chandan K. Reddy. Machine learning for survival analysis: A survey. *ACM Computing Surveys*, 51(6), 2019. "doi.org/10.1145/3214306".

[29] J.B. Willet and J.D. Singer. From whether to when: New methods for studying student dropout and teacher attrition. *Review of Educational Research*, 61(4):407–450, 1991.