2023-06-05

Comparison of the Unweighted Log-Rank and Weighted Prentice-Modified Wilcoxon Tests with Left-Truncated Data

Ellen Wight

Honors Thesis

COMPARISON OF THE UNWEIGHTED LOG-RANK AND WEIGHTED PRENTICE-MODIFIED WILCOXON TESTS WITH LEFT-TRUNCATED DATA

by
Ellen Wight

Submitted to Brigham Young University in partial fulfillment
of graduation requirements for University Honors

Statistics Department
Brigham Young University
June 2023

Advisor: Natalie Jean Blades

Honors Coordinator: Del Scott

ABSTRACT

COMPARISON OF THE UNWEIGHTED LOG-RANK AND WEIGHTED PRENTICE-MODIFIED WILCOXON TESTS WITH LEFT-TRUNCATED DATA

Ellen Wight

Statistics Department

Bachelor of Science

In survival analysis, there is a potential interaction between left-truncated data and rank test procedures for the difference between two groups that has not yet been investigated in depth. This study explores the effect of left truncation on the absolute and relative performance of the unweighted log-rank test and the weighted Prentice modification of the Gehan-Wilcoxon test. Resulting power from simulation study replications revealed that while left truncation does not affect which test performs better overall, it does affect the relative difference between the tests. It significantly worsens the individual performance of the Prentice-modified test, likely because of left truncation on the survival curve by which the test is weighted. Left truncation was also associated with an unusual decrease in power with increasing sample size for the log-rank test under non-proportional hazards.

ACKNOWLEDGEMENTS

This thesis would not have been possible without the support of my committee: Dr. Del Scott, honors coordinator; Dr. Natalie Blades, advisor; and Dr. Gregory Snow, reader. I am extremely grateful for the time they spent to help me narrow the scope of this work, provide their knowledge and expertise through various roadblocks, and proofread countless versions of the thesis.

Many thanks to all my professors at Brigham Young University who taught and encouraged me. I am also grateful to my roommates and friends who listened to me practice presentations and helped me edit this thesis for clarity. Finally, I would also like to thank my family, especially my parents, for their support throughout my undergraduate career.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

1 INTRODUCTION

Survival analysis is a branch of statistics that models how long it takes for an event to occur, commonly death in biostatistics or machine failure in reliability analysis. More specifically, it estimates the probability of surviving up to a certain time and considering both observed and incomplete data, the latter of which arises due to study limitations such as loss to follow-up or delayed entry. The most common form of incomplete data is right censoring, in which researchers only know that a subject survived sometime after a certain point. Hypothesis testing for the significance of a difference between two groups, generally a control and treatment, is a common application of the resulting model. The effect of right censoring on the inferential power of survival analysis has been studied extensively, unlike the other major form of incomplete data in survival analysis: left truncation. This research seeks to fill that gap.

The premise of left truncation states that subjects are only included in a study because they survived long enough to enter said study (Moore 2016). For example, patients generally enter a medical research study upon diagnosis. However, if their records need to be transferred to another institution, they are at risk for the event before they can be formally enrolled in the study. If they were to recover or die before records were transferred, they may never be recorded. Thus, their inclusion is conditional on the probability of them surviving long enough to enter the study. If such a study were to ignore left truncation, bias is be introduced into analyses.

As an example of the need to consider left truncation, survival analysis has recently been applied to linguistics to study the factors affecting the survival of language (Rosemeyer 2014; van de Velde and Keersmaekers 2020); however, these studies didn't

appear to consider the inherent left truncation involved in linguistics. When studying language change, a corpus of recorded words in books and other media captures the common tongue of the time period of interest. If a word was used in the vernacular for a time but never recorded, it didn't survive long enough to enter a linguistics study and is considered lost. If these words aren't theoretically considered when constructing a survival curve, linguists may incorrectly conclude which factors most affect language change.

The practical effect of incorporating left truncation lies in how subjects enter and exit the study and therefore the risk set, which is the number of subjects at risk for the event at a given time. Under a model without left truncation, the presence (or rather absence) of those "lost" subjects is never considered. The hypothetical study begins with all subjects at time zero, and only the time to event for each subject is considered (Fig. 1A). This requires the assumption that the probability of a subject surviving long enough to enter the study is one. For example, a study on treatments to quit smoking likely wouldn't involve left truncation because all patients quit at time zero, with no risk of relapsing before the study because they hadn't yet quit. When left truncation is included, however, we condition each subject's survival on the fact that they survived up to the entry time, which theoretically includes missing subjects even though they are still never in the risk set. We consider not only when a subject experienced the event, but also when they became at risk of the event relative to the other subjects (Fig. 1B, with lost subjects in grey). This requires that we know when the subject was first exposed to the risk of the event, that is, the start time upon which we condition their survival to entry into the study (Guo 1993).

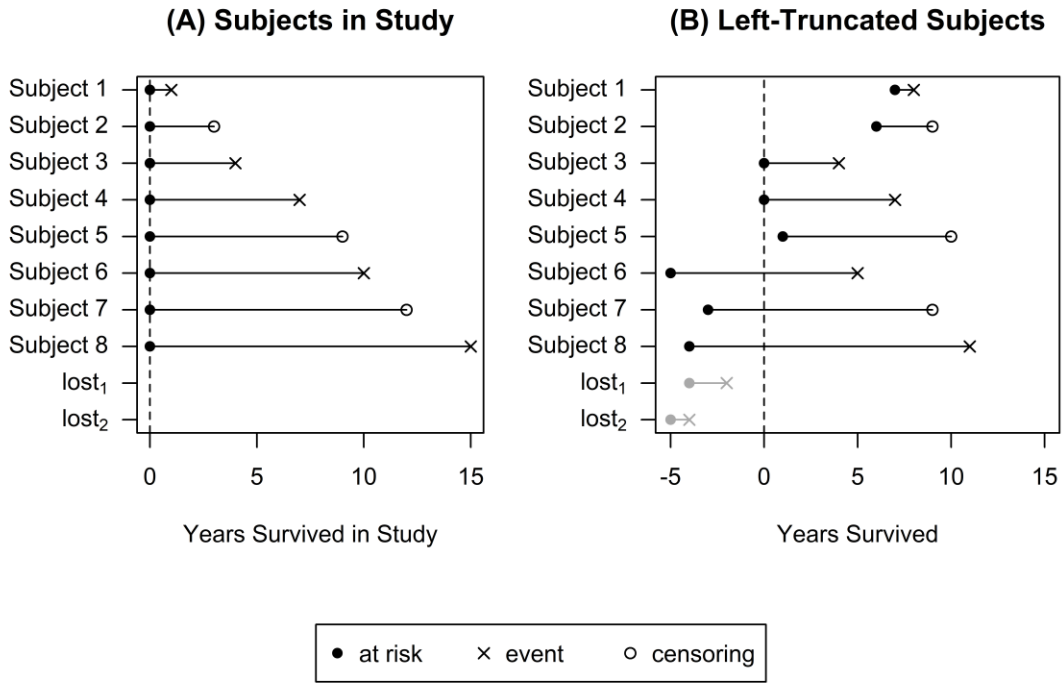

Figure 1: Sample data to illustrate the presence vs. absence of left truncation. (A) A study where left truncation is not considered. Subjects enter both the timeline and the study at time zero, which is also when they presumably become at risk for the event. At each subject's time to event or censoring, they exit the study. (B) A study where left truncation is considered. Subjects enter the timeline when they become at risk for the event, enter the study at time zero, and exit the study when they either experience the event or are censored (that is, they experience the event sometime after but are lost to follow-up, etc.). Time of entry and exit are both considered in building a model. Lost subjects, marked in grey, represent unobserved patients that are theoretically accounted for through left truncation.

The difference in risk set calculation due to left truncation then affects the calculation of the survival curve. An analysis that didn't involve left truncation would simply consider time from diagnosis to death. The risk set would begin at time zero with all subjects in the study and constantly decrease through either censoring or events, as shown by the n_i column in Table 1. Under left truncation, on the other hand, the size of

the risk set is dependent on how many subjects are in the study at each event time; those who have already exited the study and those who have not yet entered are not included. As shown in Table 2, this results in a risk set that can both increase and decrease. While the survival curve remains a nonincreasing function in either case, the estimate of the survival curve S_i differs because of the change to the risk set.

Table 1: Kaplan-Meier estimate of the survival function without left truncation. The size of the risk set n_i begins with all subjects and constantly decreases. See Figure 1A for a visual representation of these data.

t_i	n_i	d_i	$q_i = \frac{d_i}{n_i}$	$S_i = \prod(1 - q_i)$
1	8	1	0.125	0.8750
4	6	1	$0.1\overline{66}$	0.7292
7	5	1	0.2	0.5833
10	3	1	$0.\overline{33}$	0.3889
15	1	1	1.0	0.0000

Table 2: Kaplan-Meier estimate of the survival function with left truncation. The size of the risk set n_i is calculated at each event time t_i according to which subjects are in the study based on entry and exit times (see Fig. 1B), which means that it increases and decreases.

t_i	n_i	d_i	$q_i = \frac{d_i}{n_i}$	$S_i = \prod(1 - q_i)$
4	6	1	$0.1\overline{66}$	0.8333
5	5	1	0.2	0.6667
7	6	1	$0.1\overline{66}$	0.5556
8	5	1	0.2	0.4444
11	1	1	1.0	0.0000

An analysis that incorrectly omitted left truncation would be biased and potentially lead to different conclusions than an analysis that included left truncation. Figure 2 plots the example data from Table 1 in grey and Table 2 in black. When left truncation is not considered, the resulting survival curve has a more gradual decline compared to a curve with left truncation, even though the probability of survival begins to drop earlier. It also has a median survival time of 10 compared to 8 from a model that

includes left truncation. If researchers did not account for left truncation, they would generally overestimate the probability of survival.

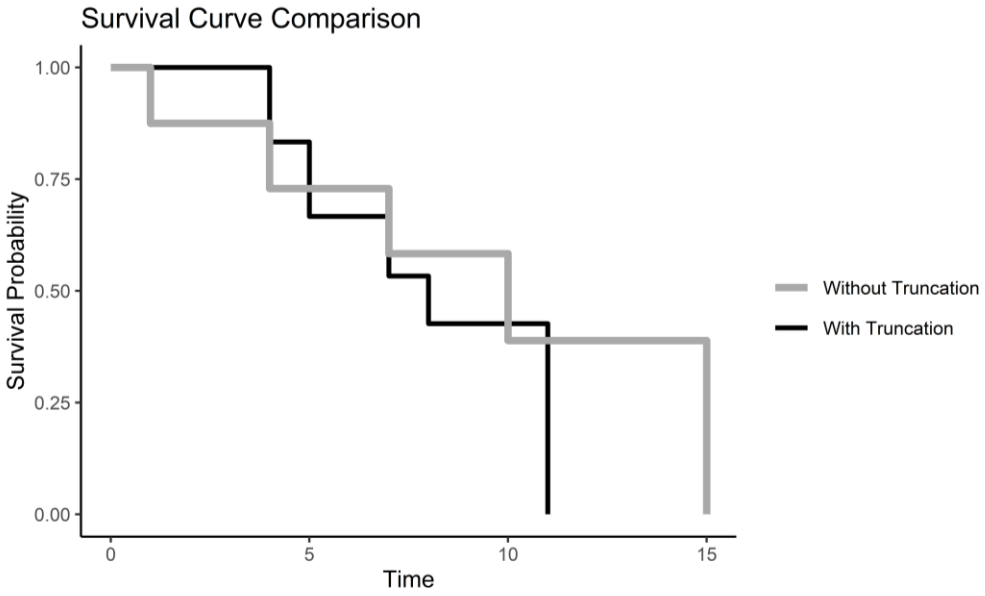

Figure 2: Comparison of survival curves with and without considering left truncation. In grey is the estimated survival curve without left truncation from Table 1; in black is the estimated curve with left truncation from Table 2. Note that including left truncation results in an overall steeper curve and a shorter timeline for survival.

After missing data have been considered and an appropriate survival curve built, rank test procedures test the difference between two independent groups. The null hypothesis of $S_1(t) = S_0(t)$ is posited against a Lehman alternative of $S_1(t) = [S_0(t)]^{\psi}$, suggesting that the control group is different from the treatment by a constant proportion. The difference is known as the proportional hazards assumption. (The hazard function is another way to represent survival by estimating the risk or hazard of experiencing the event at any given time.) This reduces to the one-sided test $H_0: \psi = 1$ and $H_A: \psi < 1$.

With this, we now test whether group 1 has a uniformly lower hazard or survival curve than group 0.

At each event time t_i, the expected value and variance of the number of events from the control group d_{0i} is calculated. d_{0i} follows a hypergeometric distribution with observed parameters n_{0i}, n_i, and d_i (the size of the risk set for each group and the total number of events at each event time) (Moore 2016). The expected value and variance of d_{0i} at each event time t_i are therefore

$$e_{0i} = \frac{n_{0i} d_i}{n_i} \qquad \text{and} \qquad v_{0i} = \frac{n_{0i} n_{1i} d_i (n_i - d_i)}{n_i^2 (n_i - 1)}.$$

To construct a test statistic, these values are summed over all observed event times. Broadly, U_0 is the aggregated difference between the expected number of events e_{0i} and observed number of events d_{0i} for group 0. V_0 is the aggregated variance v_{0i} of group 0. The resulting test statistic with its distribution is

$$\frac{U_0}{\sqrt{V_0}} \sim N(0,1) \qquad \text{or} \qquad \frac{U_0^2}{V_0} \sim \chi_1^2.$$

The difference between the rank tests lies in the calculation of U_0 and V_0. Two of the most common variations, which will be explored in this study, are the log-rank test and the Prentice modification of the Gehan-Wilcoxon test, also known as the Peto-Peto test (Peto and Peto 1972). The former is an unweighted rank test, meaning that all event times are weighted equally. It generally performs the best compared to other tests under proportional hazards but rapidly loses power when that assumption is violated. Its test statistic calculation is

$$U_0 = \sum_{i=1}^{N} (d_{0i} - e_{0i}) \qquad \text{and} \qquad V_0 = \sum_{i=1}^{N} v_{0i}.$$

The Prentice-modified test, on the other hand, is weighted by the Kaplan-Meier estimate of the survival curve $\hat{S}(t)$ (Kaplan and Meier 1958). Since the survival curve is strictly nonincreasing, this test places greater weight on early event times. This is done with the presupposition that the risk set will be largest at these early times so that the most information (or the greatest number of subjects) with which to weight the test will be present at the beginning. It generally performs best when hazards are not proportional and the two survival curves have greater differences early in the timeline (Chen et al. 2020). Its test statistic calculation is instead calculated as

$$U_0(w) = \sum_{i=1}^{N} w_i(d_{0i} - e_{0i}) \qquad \text{and} \qquad V_0(w) = \sum_{i=1}^{N} w_i^2 \, v_{0i}$$

$$\text{where} \qquad w_i = \hat{S}(t_i).$$

There is a potential yet largely unexplored interaction between left-truncated data and rank tests. As mentioned earlier, the Prentice-modified test assumes that the most information is present at early event times, which is always true in the absence of left truncation. However, when dealing with left-truncated data, the risk set is no longer strictly decreasing; the risk set could be largest in the middle of the survival curve rather than the beginning. This doesn't invalidate the use of the Prentice-modified test, but it does call into question whether it would still perform better than the log-rank given data with left truncation and non-proportional hazards.

While preliminary exploration of the effect of left truncation on rank tests has been carried out by Pan (1998), only one left truncation pattern was considered, albeit with interval censoring and a variety of differences between groups. Apart from Pan's research, rank test comparison research generally focuses on the effect of various

censoring patterns or extreme non-proportional hazards (Demirhan et al. 2009; Karadeniz and Ercan 2017; Karrison 2016; Özen et al. 2021; Royston and Parmar 2020; Teniola et al. 2022). We propose that the effect of left truncation on rank tests merits further exploration. This simulation study compares the log-rank and Prentice-modified tests under a variety of left truncation patterns, types of differences between groups, and target sample sizes. This will provide a guideline for future research by determining which test most accurately detects differences between groups given a study that involves left truncation.

2 METHODS

2.1 Simulation Design

We used a Monte Carlo simulation study (n = 10,000 replications per scenario) to explore scenarios in which left truncation and rank tests could have a significant interaction (see Appendix B for full simulation code). For all simulations, subjects were randomly assigned to one of two groups and became at risk for the event (e.g., received a diagnosis) at time zero. Each patient's entry, event, and censoring times were simulated from Weibull distributions. A subject was censored if their simulated censoring time was less than their event time (lost to follow-up before the event) and was "lost" if their event time was less than their entry time (experienced the event before entry). For each simulation, the proportion of subjects censored was approximately 25%. The three factors of the study were the left truncation pattern, the difference between groups, and the target sample size of each group, the exhaustive combinations of which were simulated and applied to each rank test.

Two left truncation patterns — discrete uniform and exponential — were compared to a reference scenario without left truncation. For both truncation patterns, 50% of subjects entered the study at time zero. This ensured that the risk set would remain populated for the remainder of the study. Under discrete uniform truncation, 25% each entered the study at time 7 and time 28. Under exponential truncation, the remaining 50% entered the study along an exponential distribution with rate $\lambda = 0.1$. Due to subject loss from left truncation, the true sample size of each simulation was always slightly lower than the target sample size (25, 50, and 100 per group) under each left truncation pattern.

The various differences between groups included in the study were designed to definitively favor one of the two tests in the absence of left truncation (Fig. 3). As a starting point, a scenario with no difference between groups ($\psi = 1$) was simulated. Then, three differences followed the proportional hazards assumption ($\psi = 0.9$, 0.95, and 0.98) to favor the log-rank test. Two differences involved non-proportional hazards — mid and late-crossing survival curves — to favor the Prentice-modified test. Both of these scenarios also had an overall difference of $\psi = 0.98$ so that one group could still be considered the higher survival curve overall.

2.2 Rank Tests

Both the log-rank and Prentice-modified tests were performed on each simulated dataset. To accommodate left truncation, a modified version of the `survival::survdiff` function was used (Appendix A), which performed the following calculations on the data set:

1. At each unique event time t_i, calculate and save d_{0i}, e_{0i}, and v_{0i} to vectors.
2. Calculate the Kaplan-Meier estimate of the survival curve $\hat{S}(t)$ for the entire data set, ignoring groups.
3. Calculate U_0 ($\sum(d_{0i} - e_{0i})$) and V_0 ($\sum v_{0i}$), weighted by 1 for the log-rank test (unweighted) and by $\hat{S}(t)$ for the Prentice test.
4. Using the test statistic for a χ_1^2 distribution, calculate the p-value for a one-sided test.

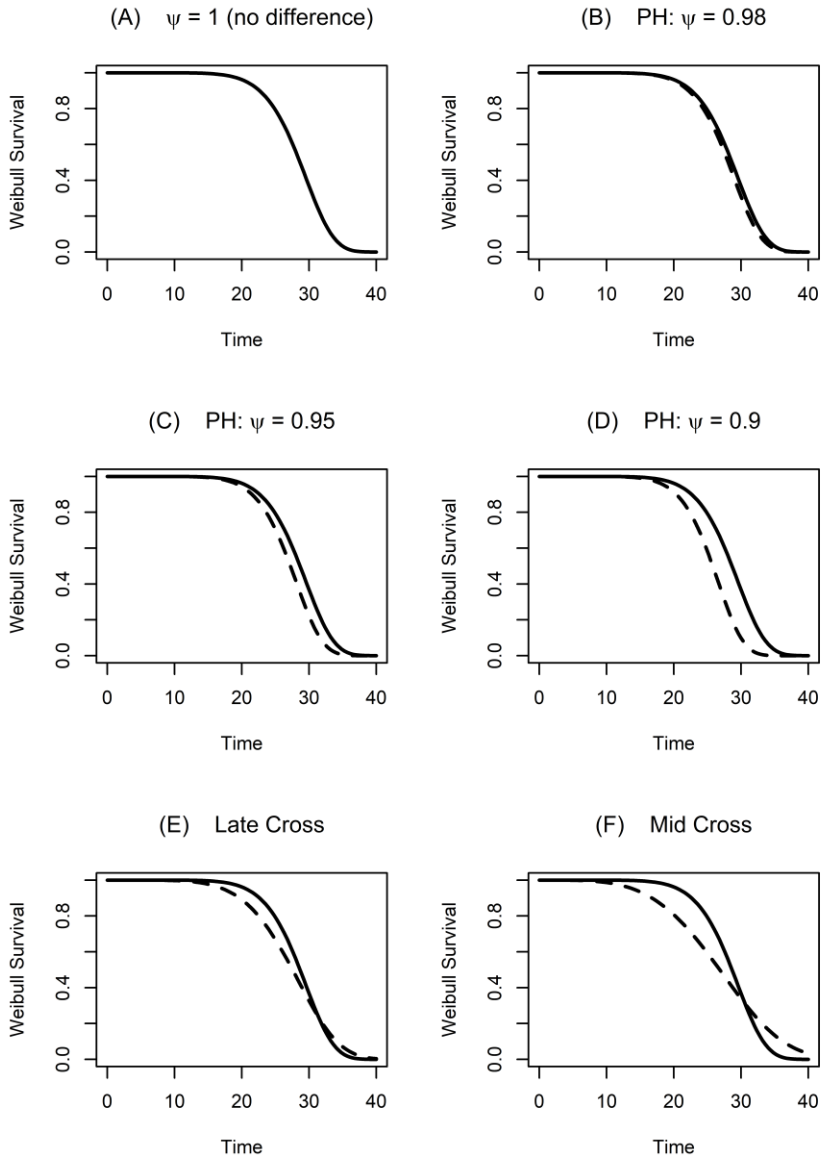

Figure 3: Graphs of all differences between groups considered in study with a baseline Weibull(8, 30) survival curve. (A) Case 1: no difference between groups. (B–D) Case 2: Varying degrees of proportional hazards as determined by the Weibull scale parameter. The solid line is the baseline; the dashed line is a Weibull(8, 30ψ) distribution. (E–F) Case 3: Varying degrees of non-proportional hazards as determined by the Weibull shape parameter. The dashed line in E follows a Weibull($8 * 0.7$, $30 * 0.98$) distribution and in F follows a Weibull($8 * 0.5$, $30 * 0.98$) distribution.

5. Accept the null hypothesis of no difference between groups when the p-value is greater than 0.5; otherwise, reject the null hypothesis in favor of the alternative that one group has a lower survival curve than the other.

When creating the modified function, resulting test statistics and p-values from sample data were compared to those from the original `survival::survdiff` function in the absence of left truncation (Terry M. Therneau and Patricia M. Grambsch 2000). The log-rank test matched exactly, while the Prentice-modified test was slightly different. This happened because the modified `survival::survdiff` function calculates $\hat{S}(t)$ by calling the `survfit` function, which calculates the Kalbfleisch-Prentice approximation of the Kaplan-Meier curve (Kalbfleisch and Prentice 2002). The original function instead calculates the Kaplan-Meier curve directly, but the modified function is similar enough to use in this study.

2.3 Comparison Metrics

When there was no difference between groups, the type I error rate was the proportion of simulations that incorrectly identified a difference between the two groups. Assuming proper study setup and randomization, the error rate for each replicated scenario should be about 0.05 to match the $\alpha = 0.05$ significance level. When there was a difference between groups, the power of each test was the proportion of simulations that correctly detected a difference between groups. While there is no standard cut-off value, higher power is associated with better performance. Without left truncation, we expected that the log-rank would have higher power than the Prentice-modified test under proportional hazards. Prentice was expected to have higher power under non-proportional hazards. No hypotheses were made about the effect of left truncation on test performance.

However, increasing the target sample size and increasing the difference between groups were both expected to increase power for both tests and under all left truncation patterns.

3 RESULTS

3.1 No Difference Between Groups (ψ = 1)

As expected, nearly all simulations contained the true α of 0.05 in a 95% confidence interval for type I error (Fig. 4). The three exceptions all came from the log-rank test, one from each left truncation pattern, and from either the 25 or 50 target sample size. The confidence intervals of these exceptions were all above 0.05; however, they were all within 0.0025 of containing the true significance level (Table 3). The log-rank test also had a higher error rate than the Prentice except at the largest sample sizes.

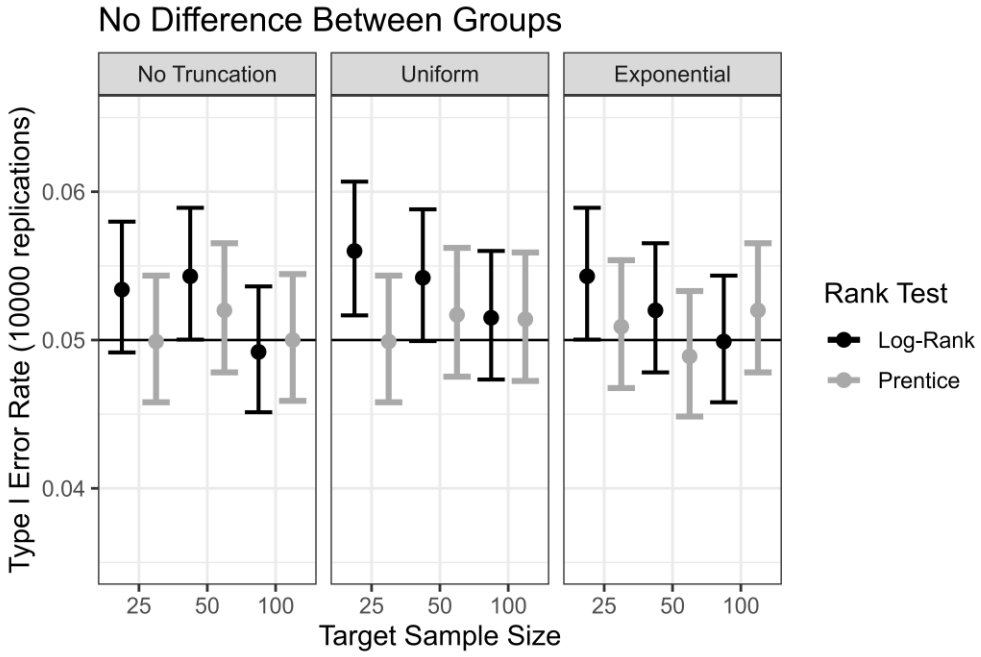

Figure 4: Resulting type I errors with 95% confidence intervals for no difference between groups compared to the true significance level of 0.05. Note that while some intervals are on the edge of 0.05 or do not contain it, the scale of the y-axis is very small. Any intervals that don't contain 0.05 do so by a small amount.

Table 3: Resulting type I error rate with 95% confidence intervals for each test when there is no difference between groups (n=10,000).

	Log-Rank			Prentice		
Sample Size	Lower 95% CI	Type I Error	Upper 95% CI	Lower 95% CI	Type I Error	Upper 95% CI
No Truncation						
25	0.0492	0.0534	0.0580	0.0458	0.0499	0.0543
50	0.0500	0.0543	0.0589	0.0478	0.0520	0.0565
100	0.0451	0.0492	0.0536	0.0459	0.0500	0.0544
Discrete Uniform						
25	0.0517	0.0560	0.0607	0.0458	0.0499	0.0543
50	0.0499	0.0542	0.0588	0.0475	0.0517	0.0562
100	0.0473	0.0515	0.0560	0.0472	0.0514	0.0559
Exponential						
25	0.0500	0.0543	0.0589	0.0468	0.0509	0.0554
50	0.0478	0.0520	0.0565	0.0448	0.0489	0.0533
100	0.0458	0.0499	0.0543	0.0478	0.0520	0.0565

3.2 Proportional Hazards

Under proportional hazards, left truncation did not affect which test performed the best. The log-rank outperformed the Prentice-modified test at all sample sizes and values of ψ regardless of left truncation. As expected, power increased as sample size increased and as the difference between groups increased (or as ψ decreased). The smallest difference between groups had particularly poor power (0.11 on average across both tests). While 95% confidence intervals were calculated (Table 4), they were small with an average width of only 0.01, so any visually recognizable differences in Figure 5 are likely significant at the 0.05 level. Differences between the tests also increased under left truncation. On average, the power of the log-rank test was higher than that of the Prentice by 11.7% under no truncation, 14.8% under uniform truncation, and 13.1% under exponential truncation.

The log-rank and Prentice-modified tests each had slightly lower absolute power under left truncation. For both tests, power was highest on average under no truncation, followed by exponential truncation, then uniform truncation. On average, the power of the log-rank test was only 7% higher and the Prentice only 11% higher under no truncation compared to uniform truncation.

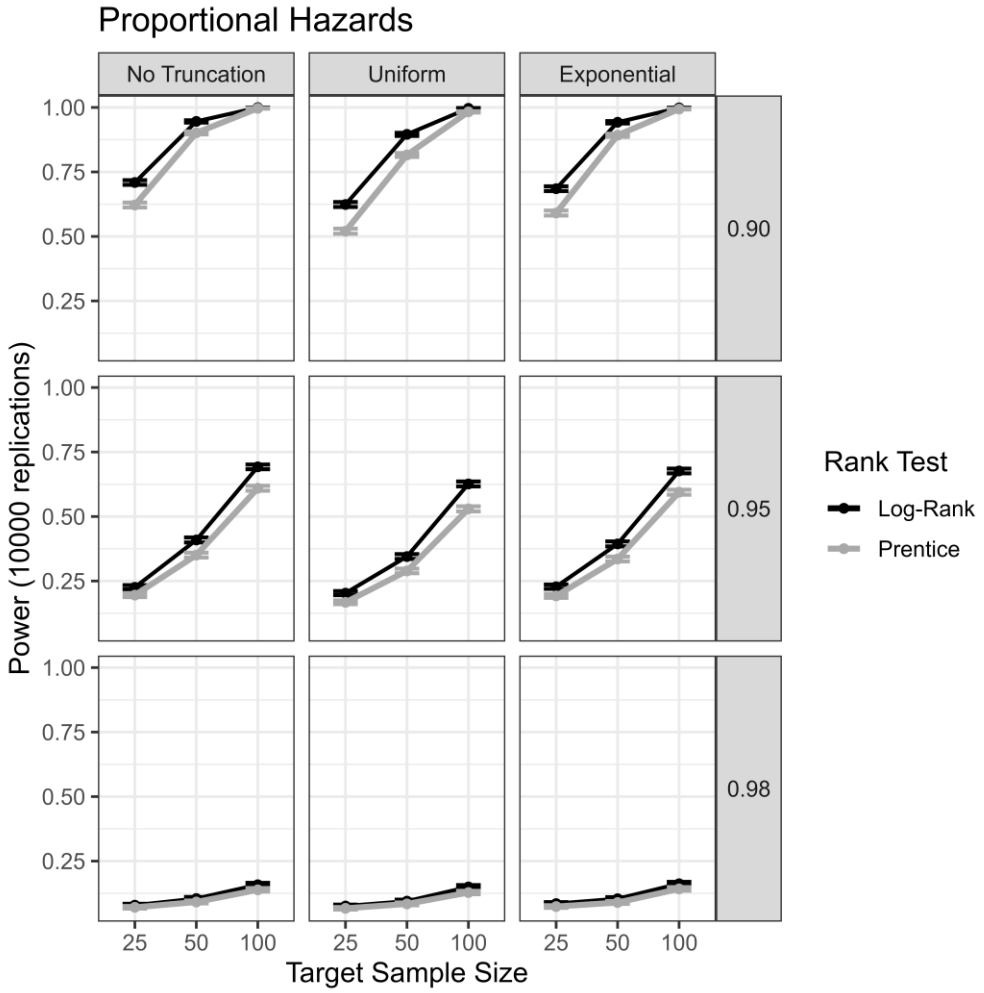

Figure 5: Resulting power with 95% confidence intervals for differences between groups that followed proportional hazards. Increasing the target sample size has a consistent positive effect on the power of each test, as does the increasing the degree of the difference between groups. The log-rank performs better than the Prentice-modified test regardless of truncation, but the power for both tests decreases slightly under left truncation.

Table 4: Resulting power with 95% confidence intervals for each test when the difference between groups follows proportional hazards (n=10,000).

		Log-Rank			Prentice		
	Sample Size	Lower 95% CI	Type I Error	Upper 95% CI	Lower 95% CI	Type I Error	Upper 95% CI
No Truncation							
0.90							
	25	0.7001	0.7091	0.7180	0.6123	0.6219	0.6314
	50	0.9413	0.9459	0.9503	0.8955	0.9015	0.9073
	100	0.9987	0.9994	0.9998	0.9953	0.9966	0.9976
0.95							
	25	0.2174	0.2256	0.2339	0.1877	0.1954	0.2033
	50	0.3994	0.4091	0.4188	0.3411	0.3505	0.3599
	100	0.6834	0.6925	0.7015	0.6000	0.6096	0.6192
0.98							
	25	0.0739	0.0791	0.0846	0.0658	0.0708	0.0760
	50	0.0989	0.1048	0.1110	0.0859	0.0915	0.0973
	100	0.1512	0.1583	0.1656	0.1319	0.1386	0.1455
Discrete Uniform							
0.90							
	25	0.6142	0.6238	0.6333	0.5106	0.5204	0.5302
	50	0.8895	0.8957	0.9016	0.8081	0.8158	0.8234
	100	0.9953	0.9966	0.9976	0.9803	0.9830	0.9854
0.95							
	25	0.1952	0.2030	0.2110	0.1601	0.1674	0.1749
	50	0.3355	0.3448	0.3542	0.2801	0.2890	0.2980
	100	0.6167	0.6263	0.6358	0.5200	0.5298	0.5396
0.98							
	25	0.0708	0.0759	0.0813	0.0621	0.0669	0.0720
	50	0.0890	0.0947	0.1006	0.0787	0.0841	0.0897
	100	0.1429	0.1498	0.1569	0.1216	0.1281	0.1348
Exponential							
0.90							
	25	0.6759	0.6851	0.6942	0.5810	0.5907	0.6004
	50	0.9379	0.9426	0.9471	0.8862	0.8924	0.8984
	100	0.9973	0.9983	0.9990	0.9921	0.9938	0.9952
0.95							
	25	0.2197	0.2279	0.2363	0.1847	0.1924	0.2003
	50	0.3842	0.3938	0.4035	0.3260	0.3353	0.3446
	100	0.6673	0.6766	0.6858	0.5845	0.5942	0.6038
0.98							
	25	0.0799	0.0853	0.0909	0.0688	0.0738	0.0791
	50	0.0982	0.1041	0.1103	0.0840	0.0895	0.0953
	100	0.1549	0.1621	0.1695	0.1354	0.1422	0.1492

3.3 Non-Proportional Hazards

Under non-proportional hazards, the Prentice-modified test outperformed the log-rank in all simulation scenarios. Like proportional hazards, power increased as sample size increased and as the difference between groups increased (more specifically, the degree of non-proportionality). Additionally, apparent differences in power (Fig. 6) are significant at the 0.05 level due to small 95% confidence interval widths. However, the relative difference between the two tests increased dramatically under non-proportional hazards, and left truncation did not consistently cause a larger difference between the two tests. On average, the power of the Prentice-modified test 290% higher than that of the log-rank under no truncation, 249% under uniform truncation, and 294% under exponential truncation. Given the violation of proportional hazards, the decrease in power of the log-rank test is unsurprising.

Left truncation again negatively affected the power of each test. Similar to the increased relative difference between groups for non-proportional hazards, these individual differences also increased when compared to proportional hazards. For both tests, power was highest on average under no truncation, followed by exponential truncation, then uniform truncation. On average, the power of the log-rank test was 31% higher and the Prentice 44% higher under no truncation compared to uniform truncation.

Unexpectedly, a larger target sample size was not consistently associated with an increase in power for the log-rank test under non-proportional hazards. The power of the log-rank test was between about 0.05 and 0.1 for all simulations involving non-proportional hazards, so drops in power were small but at times significant. For the mid-cross difference between groups in particular, an increase in sample size was always

associated with some decrease in power. This was not significant under no truncation, that is, the 95% confidence intervals overlapped. There was a significant drop in power from a sample size of 25 to 50 (and 25 to 100) in both left truncation patterns at the 0.05 level, but not from 50 to 100. For the late-cross difference between groups, power either increased with sample size or decreased by an insignificant amount at 95% confidence.

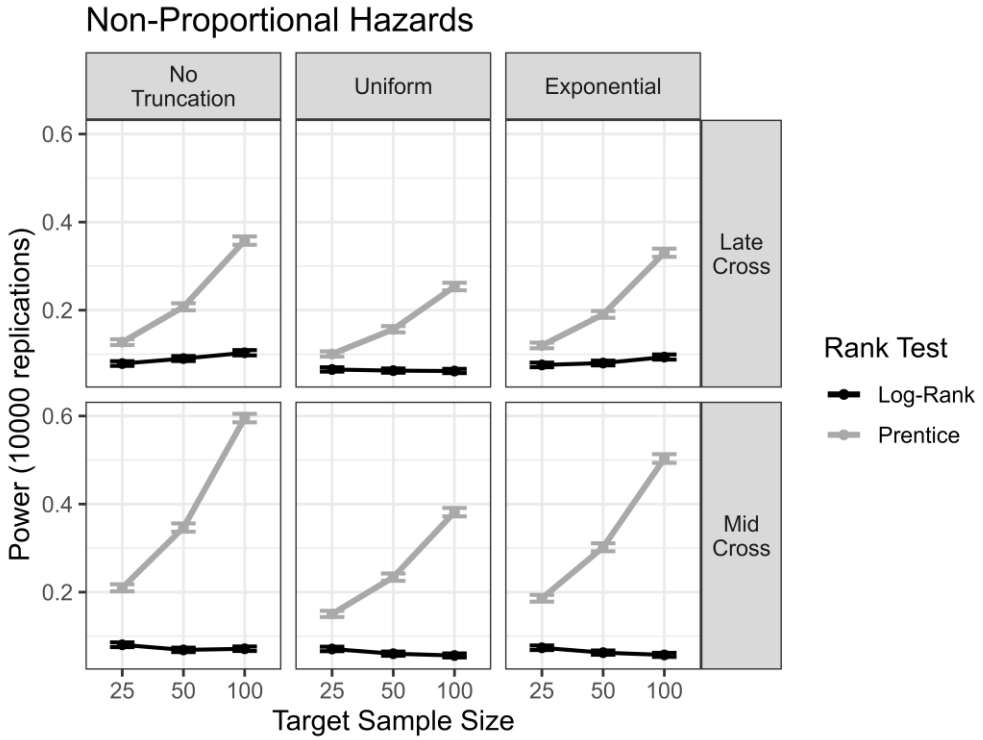

Figure 6: Resulting power with 95% confidence intervals for differences between groups that did not follow proportional hazards. Increasing the target sample size has a consistent positive effect on the power of the Prentice-modified test but has a variable effect on the log-rank test. The mid-cross difference compared to late-cross has an overall higher power for the Prentice-modified test and lower for the log-rank. The Prentice-modified test performs better than the log-rank regardless of truncation, but the power for both tests decreases slightly under left truncation. The decrease in power is more noticeable for the Prentice-modified test.

Table 5: Resulting power with 95% confidence intervals for each test when there is a non-proportional difference between groups (n=10,000).

		Log-Rank			Prentice		
	Sample Size	Lower 95% CI	Type I Error	Upper 95% CI	Lower 95% CI	Type I Error	Upper 95% CI
No Truncation							
Late Cross							
	25	0.0735	0.0787	0.0842	0.1208	0.1273	0.1340
	50	0.0846	0.0901	0.0959	0.1997	0.2076	0.2157
	100	0.0973	0.1032	0.1093	0.3485	0.3579	0.3674
Mid Cross							
	25	0.0751	0.0804	0.0859	0.2018	0.2097	0.2178
	50	0.0639	0.0688	0.0739	0.3373	0.3466	0.3560
	100	0.0664	0.0714	0.0766	0.5856	0.5953	0.6049
Discrete Uniform							
Late Cross							
	25	0.0605	0.0653	0.0703	0.0946	0.1004	0.1065
	50	0.0580	0.0627	0.0676	0.1495	0.1566	0.1639
	100	0.0572	0.0618	0.0667	0.2451	0.2536	0.2622
Mid Cross							
	25	0.0659	0.0709	0.0761	0.1434	0.1504	0.1576
	50	0.0556	0.0602	0.0650	0.2257	0.2340	0.2424
	100	0.0517	0.0561	0.0608	0.3721	0.3816	0.3912
Exponential							
Late Cross							
	25	0.0705	0.0756	0.0810	0.1135	0.1198	0.1263
	50	0.0747	0.0799	0.0854	0.1826	0.1903	0.1981
	100	0.0879	0.0935	0.0994	0.3212	0.3304	0.3397
Mid Cross							
	25	0.0686	0.0736	0.0789	0.1784	0.1860	0.1938
	50	0.0579	0.0626	0.0675	0.2928	0.3018	0.3109
	100	0.0529	0.0574	0.0621	0.4936	0.5034	0.5132

4 DISCUSSION

The simulations involving no difference between groups or no truncation both established proper study design and setup, suggesting that any differences between tests under left truncation are accurately represented by this study. While the log-rank test had higher type I errors on average under no difference between groups, this was likely due to randomization in the study. Under no truncation, the superiority of one test over the other followed expected patterns (log-rank for proportional hazards, Prentice for non-proportional hazards) given the theoretical basis and strengths of each test. When introducing left truncation, this superiority remained constant for all simulation scenarios.

While left truncation does not affect which test performs the best, it does affect the degree to which the tests differ. Under proportional hazards, left truncation is consistently associated with greater differences between the power of the two tests, although differences remain relatively small compared to when the hazards are not proportional. Under non-proportional hazards, there is more variation in the effect of left truncation on the relative difference, but the Prentice still outperforms the log-rank regardless of left truncation.

Left truncation is associated with a change in the relative power between the tests because the power of the Prentice-modified test decreases more under left truncation than the log-rank test. Under proportional hazards, it becomes even worse than the log-rank; under non-proportional hazards, the gap between the Prentice and log-rank is smaller or about the same. This effect on the Prentice test is unsurprising because left truncation affects both the expected value and variance of d_{0i} at each event time, and the estimate of the survival curve. The former is used in both tests, while the latter is used only by the

Prentice to weight the test. Essentially, the calculation of the Prentice test is affected twice by left truncation, whereas the log-rank test is only affected once.

Under left truncation and non-proportional hazards, the log-rank test is rendered almost useless with power consistently close to the significance level of 0.05. Increasing the target sample size from 25 either raises power insignificantly or decreases it significantly. By contrast, the power of the Prentice test always improves with sample size under both proportional and non-proportional hazards. Using the log-rank test on data that do not have proportional hazards will almost always result in a conclusion that misrepresents the true difference between groups.

5 CONCLUSION

Left truncation does not appear to affect the superiority of the Prentice-modified test over the log-rank given non-proportional hazards (and vice-versa under proportional hazards). Studies involving left truncation that are about certain whether hazards will be proportional will not need to consider this form of missing data when deciding which test of the two will best detect the true differences between groups in their analysis. However, either test will need to be performed with the understanding that the power is lower due to left truncation and with the knowledge of how non-proportional hazards could affect which test is more representative. Additionally, if researchers are not certain whether the data will follow proportional hazards, the Prentice-modified test will be the better option as the log-rank is rendered useless under non-proportional hazards and left truncation.

Future exploration of the interaction between left truncation and rank tests could extend to either extreme cases of left truncation, other types of rank tests. Left truncation in this study was limited to 50% of subjects to ensure the risk set was populated at each event time. Increasing the proportion of subjects who are truncated, within reasonable bounds of expected left truncation in applied research, could increase the effect of left truncation on rank test performance. Other rank tests that could be included in future studies, which have been considered by past research involving right censoring, include the Fleming-Harrington family of tests, the Gehan Generalized Wilcoxon test, the Tarone-Ware test, and partitioned log-rank tests, among others (Karadeniz and Ercan 2017; Özen et al. 2021).

If future exploration were to conclude that left truncation does decrease the power of all currently developed rank tests (see Chen et al. 2020 for a list of known rank tests), a

new rank test specifically developed for left truncation may be needed. One potential way to handle left truncation in a rank test is to somehow consider the size of the risk set as a random variable since subjects becoming at risk for an event (receiving a diagnosis) is more random than subjects being enrolled in a study (conditional upon diagnosis or planned as an experiment). By developing a new rank test for left-truncated data, researchers could be more certain about their choice of rank test if they were uncertain whether data would follow proportional hazards.

REFERENCES

Chen, X., Wang, X., Chen, K., Zheng, Y., Chappell, R. J., and Dey, J. (2020), "Comparison of survival distributions in clinical trials: A practical guidance," *Clinical trials (London, England)*, 17, 507–521. https://doi.org/10.1177/1740774520928614.

Demirhan, Y. P., Demirhan, H., and Bacanli, S. (2009), "A simulation study on power comparisons for group sequential tests of non-parametric statistics," *Journal of statistical computation and simulation*, 79, 769–785. https://doi.org/10.1080/00949650801930682.

Guo, G. (1993), "Event-History Analysis for Left-Truncated Data," *Sociological methodology*, 23, 217–243. https://doi.org/10.2307/271011.

Kalbfleisch, J. D., and Prentice, R. L. (2002), *The statistical analysis of failure time data*, Hoboken, N.J: Wiley-Interscience.

Kaplan, E. L., and Meier, P. (1958), "Nonparametric Estimation from Incomplete Observations," *Journal of the American Statistical Association*, 53, 457–481. https://doi.org/10.1080/01621459.1958.10501452.

Karadeniz, P. G., and Ercan, I. (2017), "Examining Tests for Comparing Survival Curves with Right Censored Data," *Statistics in Transition*, 18, 311–328. https://doi.org/10.21307/stattrans-2016-072.

Karrison, T. G. (2016), "Versatile Tests for Comparing Survival Curves Based on Weighted Log-rank Statistics," *The Stata journal*, 16, 678–690. https://doi.org/10.1177/1536867X1601600308.

Moore, D. F. (2016), *Applied survival analysis using R*, Cham: Springer. https://doi.org/10.1007/978-3-319-31245-3.

Özen, H., Bal, C., and Çolak, E. (2021), "Comparison of the Statistical Tests for Two Crossing Survival Functions," *Türkiye klinikleri bıyoıstatıstık*, 13, 13–27. https://doi.org/10.5336/biostatic.2020-80208.

Pan, W. (1998), "Rank invariant tests with left truncated and interval censored data," *Journal of statistical computation and simulation*, 61, 163–174. https://doi.org/10.1080/00949659808811907.

Peto, R., and Peto, J. (1972), "Asymptotically Efficient Rank Invariant Test Procedures," *Journal of the Royal Statistical Society. Series A. General*, 135, 185–207. https://doi.org/10.2307/2344317.

R Core Team (2022), "R: A Language and Environment for Statistical Computing," Vienna, Austria.

Rosemeyer, M. (2014), “Gradualness and conservation in the loss of ser + PtcP,” Auxiliary Selection in Spanish : Gradience, gradualness, and conservation, eds. W. Abraham and E. van Gelderen, Amsterdam: John Benjamins Publishing Company, pp. 185–261.

Royston, P., and Parmar, M. K. B. (2020), “A simulation study comparing the power of nine tests of the treatment effect in randomized controlled trials with a time-to-event outcome,” *Current controlled trials in cardiovascular medicine*, 21, 315. https://doi.org/10.1186/s13063-020-4153-2.

Teniola, B. B., Ayantunji, A. R., Yusuf, H. O., Olufunmilola, O. A., Ebenezer, O. R., Abimbola, A. K., Oluwaseun, A. S., Idowu, A. M., Femi, O. O., Emmanuel, F. S., Augustine, E. O., Oluwadare, A., Daniel, O. O., Olanrewaju, F., and Martin, K. (2022), “Statistical Powers of Some Tests for Checking Homogeneity of Survival Distributions with Disjointed Ends in the Presence of Censoring,” *International journal of analysis and applications*, 20, 46. https://doi.org/10.28924/2291-8639-20-2022-46.

Terry M. Therneau, and Patricia M. Grambsch (2000), *Modeling Survival Data: Extending the Cox Model*, New York: Springer.

van de Velde, F., and Keersmaekers, A. (2020), “What are the determinants of survival curves of words?: An evolutionary linguistics approach,” *Evolutionary Linguistic Theory*, 2, 127–137. https://doi.org/https://doi.org/10.1075/elt.00019.vel.

Wickham, H. (2022), “stringr: Simple, Consistent Wrappers for Common String Operations.”

APPENDICES

Appendix A. Modified `survdiff` Function

The original `survdiff` function was not compatible with left truncation due to the algorithm used to iterate through the event times. Its algorithm assumed that the size of the risk set was nonincreasing. In other words, it looped through a while statement that stopped when the number of subjects accounted for by the algorithm, which constantly increased, was equal to the number of subjects in the study. The new algorithm to work with left truncation therefore had to iterate through all unique event times rather than rely on the size of the risk set to determine if all subjects had been accounted for.

The `survdiff` source code called a function called `survdiff.fit`, which in turn called a function called `Csurvdiff2`. As the name suggests, it was written in C, a language with which the author is not familiar. As such, we rewrote the algorithm in R, removing any calls to other functions (including `survdiff.fit`). Since code outside of `Csurvdiff2` did primarily error checking, significant revisions to the code were made. The strata and offset functionality in the original function were removed since they were not considered in the study, but the function can now handle both untruncated and left truncated data. The author is currently working to add those features back in to submit to the `survival` package. The modified code used in this study is shown on the next page.

```
library(survival)

## source code: survdiff, survdiff.fit, and Csurvdiff2
# survdiff
# https://rdrr.io/cran/survival/src/R/survdiff.fit.R
# https://github.com/cran/survival/blob/master/src/survdiff2.c

## Major changes:
# Removed error check that only accepted right censor
# Removed all strata/offset functionality (not considered in study)
# Removed survdiff.fit (and thus Csurvdiff2)
# Calculated test statistic by looping through unique event times
  # (original function looped through risk set until n = 0)

survdiff.mod <- function (formula, data, subset, na.action,
                          rho = 0, timefix = TRUE) {
  # get elements from function input
  call <- match.call()
  m <- match.call(expand.dots = FALSE)
  m$rho <- NULL

  ## ## ## ## ## ##
  ## error checking
  ## ## ## ## ## ##

  # from original survdiff function
  if (!inherits(formula, "formula"))
    stop("The 'formula' argument is not a formula")
  Terms <- if (missing(data))
    terms(formula, "strata")
  else terms(formula, "strata", data = data)
  m$formula <- Terms
  m[[1L]] <- quote(stats::model.frame)
  m <- eval(m, parent.frame())
  y <- model.extract(m, "response") # Surv object
  if (!inherits(y, "Surv"))
    stop("Response must be a survival object")
  ny <- ncol(y) # 2 (w/o left truncation) or 3 (w/ left truncation)
  n <- nrow(y) # number of observations
  if (!is.logical(timefix) || length(timefix) > 1)
    stop("invalid value for timefix option")
  if (timefix)
    y <- aeqSurv(y)
  ll <- attr(Terms, "term.labels")
  if (length(ll) == 0)
```

```
  stop("No groups to test")
else groups <- strata(m[ll]) # group for each observation

## ## ## ## ##
## rank test
## ## ## ## ##

# move untruncated to left truncation "format" with all entry = 0
if (ny == 2) {
  times <- data.frame(entry = rep(0, n),
                      exit = y[,1],
                      cens = y[,2])
}

# left truncation stays in the same format
else {
  times <- data.frame(entry = y[,1],
                      exit = y[,2],
                      cens = y[,3])
}

if (nrow(y) !=n | length(groups) !=n) stop("Data length mismatch")
if (length(unique(groups)) < 2) stop ("There is only 1 group")

# format group as numeric 0 or 1
if (inherits(groups, "factor")) times$groups <- as.numeric(groups) - 1
else times$groups <- match(groups, unique(groups))

# initialize vectors and i
e0i_all <- c() # expected value
v0i_all <- c() # variance
d0i_all <- c() # number of events in control group
i = 1

# create vector of all unique event times (exclude censoring)
event_times <- sort(unique(times[times$cens == 1,]$exit))

# at each observed event time t:
for (t in event_times) {

  # number in risk set
  n0i <- nrow(
    times[(times$entry <= t) & (t <= times$exit) & (times$groups == 0),]
    )
  n1i <- nrow(
```

```
      times[(times$entry <= t) & (t <= times$exit) & (times$groups == 1),]
      )
    ni <- n0i + n1i

    # number of events
    d0i <- nrow(
      times[(t == times$exit) & (times$groups == 0) & (times$cens == 1),]
      )
    d0i_all[i] <- d0i
    d1i <- nrow(
      times[(t == times$exit) & (times$groups == 1) & (times$cens == 1),]
      )
    di <- d0i + d1i

    # E(d0i) and Var(d0i)
    e0i <- (n0i * di) / ni
    e0i_all[i] <- e0i
    v0i <- (n0i * n1i * di * (ni - di)) / ((ni^2) * (ni - 1))
    v0i_all[i] <- v0i

    i = i + 1
  }

  # Kaplan-Meier S(t) at each event, ignoring group
  # note: survfit uses the Kalbfleisch-Prentice approximation
  sf <- survfit(Surv(entry, exit, cens) ~ 1, data = times, type="kaplan-
meier")
  surv <- data.frame(time = sf$time, event = sf$n.event, surv = sf$surv)

  # raise S(t) to the rho power (equivalent to FH(p))
  weights_rho <- (surv[surv$event > 0,]$surv)^rho

  # calculate chi-sq test statistic and p-value
  U <- sum(weights_rho*(d0i_all - e0i_all), na.rm=TRUE)
  V <- sum((weights_rho^2) * v0i_all, na.rm=TRUE)
  chi <- (U^2)/V
  pval <- pchisq(chi, 1, lower.tail = FALSE)

  # both test statistic and p-value for comparison with survdiff
  # return(c(paste("test statistic:", round(chi,4)),
  #          paste("p-value:", round(pval,4))))

  return(pval)
}
```

Appendix B. Simulation Code

The following three subsections display code used to simulate data for each of the three left truncation patterns. The main difference between each subsection is the `sim.each` function, which controlled truncation.

No Truncation

```
library(survival)
source("modified survdiff.R")

# Pattern: No left truncation for comparison

sim.ref <- function(size, ph, hr, cr) {
  sim.each <- function(size, ph, hr, g) {
    # event times
    event <- rweibull(size, 8*ph, 30*hr)
    # censor times
    c <- rweibull(size, 8*ph, 30*cr)

    # for the stop time, take the minimum of death and censored time
    stop <- pmin(event, c)
    # 0 as start time
    start <- 0
    # censoring vs. event indicator
    cens <- as.numeric(event < c)

    # gather into full dataframe
    set <- data.frame(start, stop, cens)
    # set group
    set$group <- g

    # return final dataset
    set
  }
  # simulate control and treatment
  set <- rbind(sim.each(size, 1, 1, "control"),
               sim.each(size, ph, hr, "treatment"))

  # rank tests
  lr.p <- survdiff.mod(Surv(start, stop, cens)~group, data=set)
  wlr.p <- survdiff.mod(Surv(start, stop, cens)~group, data=set, rho=1)
```

```
  # return simulated dataset:
  # return(set)

  # if interested in seeing % censored and nrow:
  # return(c("cens" = mean(set$cens == 0), "nrow" = nrow(set)))

  # return p-values
  return(c("LR" = lr.p, "WLR" = wlr.p))
}

## n=25
# PH
write.csv(data.frame(t(replicate(10000, sim.ref(25, 1, 1, 1.125)))),
          "results/ref21.csv", row.names=FALSE)
write.csv(data.frame(t(replicate(10000, sim.ref(25, 1, .98, 1.115)))),
          "results/ref298.csv", row.names=FALSE)
write.csv(data.frame(t(replicate(10000, sim.ref(25, 1, .95, 1.11)))),
          "results/ref295.csv", row.names=FALSE)
write.csv(data.frame(t(replicate(10000, sim.ref(25, 1, .9, 1.105)))),
          "results/ref290.csv", row.names=FALSE)
# non-PH
write.csv(data.frame(t(replicate(10000, sim.ref(25, 0.7, 0.98, 1.15)))),
          "results/ref2987.csv", row.names=FALSE)
write.csv(data.frame(t(replicate(10000, sim.ref(25, 0.5, 0.98, 1.2)))),
          "results/ref2985.csv", row.names=FALSE)

## n=50
# PH
write.csv(data.frame(t(replicate(10000, sim.ref(50, 1, 1, 1.125)))),
          "results/ref51.csv", row.names=FALSE)
write.csv(data.frame(t(replicate(10000, sim.ref(50, 1, .98, 1.115)))),
          "results/ref598.csv", row.names=FALSE)
write.csv(data.frame(t(replicate(10000, sim.ref(50, 1, .95, 1.11)))),
          "results/ref595.csv", row.names=FALSE)
write.csv(data.frame(t(replicate(10000, sim.ref(50, 1, .9, 1.105)))),
          "results/ref590.csv", row.names=FALSE)
# non-PH
write.csv(data.frame(t(replicate(10000, sim.ref(50, 0.7, 0.98, 1.15)))),
          "results/ref5987.csv", row.names=FALSE)
write.csv(data.frame(t(replicate(10000, sim.ref(50, 0.5, 0.98, 1.2)))),
          "results/ref5985.csv", row.names=FALSE)

## n=100
# PH
write.csv(data.frame(t(replicate(10000, sim.ref(100, 1, 1, 1.125)))),
```

```
          "results/ref11.csv", row.names=FALSE)
write.csv(data.frame(t(replicate(10000, sim.ref(100, 1, .98, 1.115)))),
          "results/ref198.csv", row.names=FALSE)
write.csv(data.frame(t(replicate(10000, sim.ref(100, 1, .95, 1.11)))),
          "results/ref195.csv", row.names=FALSE)
write.csv(data.frame(t(replicate(10000, sim.ref(100, 1, .9, 1.105)))),
          "results/ref190.csv", row.names=FALSE)
# non-PH
write.csv(data.frame(t(replicate(10000, sim.ref(100, 0.7, 0.98, 1.15)))),
          "results/ref1987.csv", row.names=FALSE)
write.csv(data.frame(t(replicate(10000, sim.ref(100, 0.5, 0.98, 1.2)))),
          "results/ref1985.csv", row.names=FALSE)
```

Uniform Truncation

```
library(survival)
source("modified survdiff.R")

# Pattern: 50% not truncated, 25% truncated at first week,
#          25% truncated at fourth week

sim.unif <- function(size, ph, hr, cr) {
  sim.each <- function(size, ph, hr, g) {
    # event times
    event <- rweibull(size, 8*ph, 30*hr)
    # truncation times
    trunc <- sample(c(0, 7, 28), size=size,
                    replace=TRUE, prob=c(.5, .25, .25))
    # censor times
    c <- rweibull(size, 8*ph, 30*cr)

    # stop: minimum of event and censored times
    stop <- pmin(event, c)
    # start: minimum of the stop and truncation times
    start <- pmin(stop, trunc)
    # censoring vs. event indicator
    cens <- as.numeric(event < c)

    # gather into full dataframe
    set <- data.frame(start, stop, cens)
    # set group
    set$group <- g
    # observations that aren't observed will have the same start/stop time
    set <- set[set$start != set$stop,]
```

```
    # return final dataset
    set
  }
  # simulate control and treatment
  set <- rbind(sim.each(size, 1, 1, "control"),
               sim.each(size, ph, hr, "treatment"))

  # rank test results
  lr.p <- survdiff.mod(Surv(start, stop, cens)~group, data=set)
  wlr.p <- survdiff.mod(Surv(start, stop, cens)~group, data=set, rho=1)

  # return simulated dataset:
  # return(set)

  # if interested in seeing % censored and nrow:
  # return(c("cens" = mean(set$cens == 0), "nrow" = nrow(set)))

  # return p-values:
  return(c("LR" = lr.p, "WLR" = wlr.p))

}

## n=25
# PH
write.csv(data.frame(t(replicate(10000, sim.unif(25, 1, 1, 1.125)))),
          "results/unif21.csv", row.names=FALSE)
write.csv(data.frame(t(replicate(10000, sim.unif(25, 1, .98, 1.115)))),
          "results/unif298.csv", row.names=FALSE)
write.csv(data.frame(t(replicate(10000, sim.unif(25, 1, .95, 1.11)))),
          "results/unif295.csv", row.names=FALSE)
write.csv(data.frame(t(replicate(10000, sim.unif(25, 1, .9, 1.105)))),
          "results/unif290.csv", row.names=FALSE)
# non-PH
write.csv(data.frame(t(replicate(10000, sim.unif(25, 0.7, .98, 1.15)))),
          "results/unif2987.csv", row.names=FALSE)
write.csv(data.frame(t(replicate(10000, sim.unif(25, 0.5, .98, 1.2)))),
          "results/unif2985.csv", row.names=FALSE)

## n=50
# PH
write.csv(data.frame(t(replicate(10000, sim.unif(50, 1, 1, 1.125)))),
          "results/unif51.csv", row.names=FALSE)
write.csv(data.frame(t(replicate(10000, sim.unif(50, 1, .98, 1.115)))),
          "results/unif598.csv", row.names=FALSE)
```

```
write.csv(data.frame(t(replicate(10000, sim.unif(50, 1, .95, 1.11)))),
          "results/unif595.csv", row.names=FALSE)
write.csv(data.frame(t(replicate(10000, sim.unif(50, 1, .9, 1.105)))),
          "results/unif590.csv", row.names=FALSE)
# non-PH
write.csv(data.frame(t(replicate(10000, sim.unif(50, 0.7, .98, 1.15)))),
          "results/unif5987.csv", row.names=FALSE)
write.csv(data.frame(t(replicate(10000, sim.unif(50, 0.5, .98, 1.2)))),
          "results/unif5985.csv", row.names=FALSE)

## n=100
# PH
write.csv(data.frame(t(replicate(10000, sim.unif(100, 1, 1, 1.125)))),
          "results/unif11.csv", row.names=FALSE)
write.csv(data.frame(t(replicate(10000, sim.unif(100, 1, .98, 1.115)))),
          "results/unif198.csv", row.names=FALSE)
write.csv(data.frame(t(replicate(10000, sim.unif(100, 1, .95, 1.11)))),
          "results/unif195.csv", row.names=FALSE)
write.csv(data.frame(t(replicate(10000, sim.unif(100, 1, .9, 1.105)))),
          "results/unif190.csv", row.names=FALSE)
# non-PH
write.csv(data.frame(t(replicate(10000, sim.unif(100, 0.7, .98, 1.15)))),
          "results/unif1987.csv", row.names=FALSE)
write.csv(data.frame(t(replicate(10000, sim.unif(100, 0.5, .98, 1.2)))),
          "results/unif1985.csv", row.names=FALSE)
```

Exponential Truncation

```
library(survival)
source("modified survdiff.R")

# Pattern: 50% not truncated, 50% following exp

sim.exp <- function(size, ph, hr, cr) {
  sim.each <- function(size, ph, hr, g) {
    # event times
    event <- rweibull(size, 8*ph, 30*hr)
    # randomly sample half as untruncated, half as 1 (placeholder)
    trunc <- sample(c(0, 1), size=size, replace=TRUE)
    # truncation times
    trunc[trunc == 1] <- rexp(length(trunc[trunc == 1]), 1/10)
    # censor times
    c <- rweibull(size, 8*ph, 30*cr)
```

```
    # stop: minimum of event and censored times
    stop <- pmin(event, c)
    # start: minimum of the stop and truncation times
    start <- pmin(stop, trunc)
    # censoring vs. event indicator
    cens <- as.numeric(event < c)

    # gather into full dataframe
    set <- data.frame(start, stop, cens)
    # set group
    set$group <- g
    # observations that aren't observed will have the same start/stop time
    set <- set[set$start != set$stop,]

    # return final dataset
    set
  }
  # simulate control and treatment
  set <- rbind(sim.each(size, 1, 1, "control"),
               sim.each(size, ph, hr, "treatment"))

  # rank test results
  lr.p <- survdiff.mod(Surv(start, stop, cens)~group, data=set)
  wlr.p <- survdiff.mod(Surv(start, stop, cens)~group, data=set, rho=1)

  # return simulated dataset:
  # return(set)

  # if interested in seeing % censored and nrow:
  # return(c("cens" = mean(set$cens == 0), "nrow" = nrow(set)))

  # return p-values
  return(c("LR" = lr.p, "WLR" = wlr.p))
}


## n=25
# PH
write.csv(data.frame(t(replicate(10000, sim.exp(25, 1, 1, 1.125)))),
          "results/exp21.csv", row.names=FALSE)
write.csv(data.frame(t(replicate(10000, sim.exp(25, 1, .98, 1.115)))),
          "results/exp298.csv", row.names=FALSE)
write.csv(data.frame(t(replicate(10000, sim.exp(25, 1, .95, 1.11)))),
          "results/exp295.csv", row.names=FALSE)
write.csv(data.frame(t(replicate(10000, sim.exp(25, 1, .9, 1.105)))),
```

```
          "results/exp290.csv", row.names=FALSE)
# non-PH
write.csv(data.frame(t(replicate(10000, sim.exp(25, 0.7, .98, 1.15)))),
          "results/exp2987.csv", row.names=FALSE)
write.csv(data.frame(t(replicate(10000, sim.exp(25, 0.5, .98, 1.2)))),
          "results/exp2985.csv", row.names=FALSE)

## n=50
# PH
write.csv(data.frame(t(replicate(10000, sim.exp(50, 1, 1, 1.125)))),
          "results/exp51.csv", row.names=FALSE)
write.csv(data.frame(t(replicate(10000, sim.exp(50, 1, .98, 1.115)))),
          "results/exp598.csv", row.names=FALSE)
write.csv(data.frame(t(replicate(10000, sim.exp(50, 1, .95, 1.11)))),
          "results/exp595.csv", row.names=FALSE)
write.csv(data.frame(t(replicate(10000, sim.exp(50, 1, .9, 1.105)))),
          "results/exp590.csv", row.names=FALSE)
# non-PH
write.csv(data.frame(t(replicate(10000, sim.exp(50, 0.7, .98, 1.15)))),
          "results/exp5987.csv", row.names=FALSE)
write.csv(data.frame(t(replicate(10000, sim.exp(50, 0.5, .98, 1.2)))),
          "results/exp5985.csv", row.names=FALSE)

## n=100
# PH
write.csv(data.frame(t(replicate(10000, sim.exp(100, 1, 1, 1.125)))),
          "results/exp11.csv", row.names=FALSE)
write.csv(data.frame(t(replicate(10000, sim.exp(100, 1, .98, 1.115)))),
          "results/exp198.csv", row.names=FALSE)
write.csv(data.frame(t(replicate(10000, sim.exp(100, 1, .95, 1.11)))),
          "results/exp195.csv", row.names=FALSE)
write.csv(data.frame(t(replicate(10000, sim.exp(100, 1, .9, 1.105)))),
          "results/exp190.csv", row.names=FALSE)
# non-PH
write.csv(data.frame(t(replicate(10000, sim.exp(100, 0.7, .98, 1.15)))),
          "results/exp1987.csv", row.names=FALSE)
write.csv(data.frame(t(replicate(10000, sim.exp(100, 0.5, .98, 1.2)))),
          "results/exp1985.csv", row.names=FALSE)
```

Appendix C. Error and Power from Simulations

This code extracted the type I error and power from simulations and compiled those findings into a dataframe with each line representing a unique combination of rank test, truncation pattern, difference between groups, and sample size, as indicated by the file name.

```
library(stringr)

# list files by goal (error rate vs. power)
results <- list.files("Results")
results.error <- results[lapply(results,
                                function(x) str_sub(x, -5, -5)) == 1]
results.power <- results[lapply(results,
                                function(x) str_sub(x, -5, -5)) != 1]

# set up empty dataframes (2 rank tests per file)
df.error <- data.frame(matrix(ncol=0, nrow=2*length(results.error)))
df.power <- data.frame(matrix(ncol=0, nrow=2*length(results.power)))

## ## ## ## ##
## Functions
## ## ## ## ##

# get target sample size
get.size <- function(x) {
  num <- str_extract(x, "([0-9])")

  if(num == 2) {return(25)}
  else if(num == 5) {return(50)}
  else {return(100)}
}

# get type of difference
get.diff <- function(x) {
  num <- str_extract(x, "([0-9]+)")

  if(str_length(num) == 3) {
    return(paste("PH (.", str_sub(num, 2), ")", sep=""))
  } else {
    return(paste("non-PH (.", str_sub(num,2,3), ", .",
                 str_sub(num, 4), ")", sep=""))
```

```
  }
}

# read in file and get type I error rate/se
get.error <- function(user_file) {
  df <- read.csv(paste("Results/", user_file, sep=""))
  means <- apply(df, 2, function(x) prop.test(sum(x<0.05), length(x),
                                              correct=FALSE)$estimate)
  lower <- apply(df, 2, function(x) prop.test(sum(x<0.05), length(x),
                                              correct=FALSE)$conf.int[1])
  upper <- apply(df, 2, function(x) prop.test(sum(x<0.05), length(x),
                                              correct=FALSE)$conf.int[2])
  return(list("means" = means, "lower" = lower, "upper" = upper))
}

# read in file and get power
get.power <- function(user_file) {
  df <- read.csv(paste("Results/", user_file, sep=""))
  means <- apply(df, 2, function(x) binom.test(sum(x<0.05),
                                               length(x))$estimate)
  lower <- apply(df, 2, function(x) binom.test(sum(x<0.05),
                                               length(x))$conf.int[1])
  upper <- apply(df, 2, function(x) binom.test(sum(x<0.05),
                                               length(x))$conf.int[2])
  return(list("means" = means, "lower" = lower, "upper" = upper))
}


## ## ## ## ## ## ## ## ##
## Error (no difference)
## ## ## ## ## ## ## ## ##

## identifiers:
# file name (to check that this is working)
df.error$file <- rep(results.error, each=2)
# truncation pattern
df.error$truncation <- factor(
  rep(unlist(lapply(results.error,
                    function(x) str_extract(x, "([a-z]+)"))), each=2),
  levels=c("ref", "unif", "exp")
  )

# rank test
df.error$test <- factor(rep(c("Log-Rank", "Prentice"),
                            times=length(results.error)))
```

```
# target sample size
df.error$size <- factor(rep(unlist(lapply(results.error, FUN=get.size)),
                            each=2))

## results:
errors <- lapply(results.error, function(x) get.error(x))

# type I error rate
df.error$rate <- unlist(lapply(errors, function(x) x[["means"]]))
# CI lower
df.error$lower <- unlist(lapply(errors, function(x) x[["lower"]]))
# CI upper
df.error$upper <- unlist(lapply(errors, function(x) x[["upper"]]))

## ## ## ## ## ## ## ##
## Power (difference)
## ## ## ## ## ## ## ##

## identifiers
# file name (to check that this is working)
df.power$file <- rep(results.power, each=2)
# truncation pattern
df.power$truncation <- factor(
  rep(unlist(lapply(results.power,
                    function(x) str_extract(x, "([a-z]+)"))), each=2),
  levels=c("ref", "unif", "exp")
  )
# sample size
df.power$size <- factor(rep(unlist(lapply(results.power, FUN=get.size)),
                            each=2))
# difference
df.power$difference <- factor(
  rep(unlist(lapply(results.power, FUN=get.diff)), each=2)
  )

# test
df.power$test <- factor(rep(c("Log-Rank", "Prentice"),
                            times=length(results.power)))

## results:
powers <- lapply(results.power, function(x) get.power(x))

# Power
df.power$power <- unlist(lapply(powers, function(x) x[["means"]]))
# CI lower
```

```
df.power$lower <- unlist(lapply(powers, function(x) x[["lower"]]))
# CI upper
df.power$upper <- unlist(lapply(powers, function(x) x[["upper"]]))
```