On Estimating Survival Function of Stochastic Order[*]

Juan Gallegos, Daisy (Yan) Huang, Thien T. Nguyen, Gregory Schrage

July 30, 2004

Abstract

Let $\overline{F}$, $\overline{G}$, and $\overline{H}$ be survival functions satisfying the constraint $\overline{F} \leq \overline{H} \leq \overline{G}$. Lee, Yan, and Shi (1999) had developed an algorithm to estimate the survival function $\overline{H}$ when $\overline{F}$ and $\overline{G}$ are known. However, lacking a closed form of the estimator makes the investigations of the properties of the estimator difficult. In this paper, we propose alternative estimators for $\overline{H}$ in the case where $\overline{F}$ and $\overline{G}$ are known and in the case where they are unknown. The estimators are proved to be strongly uniformly consistent in both cases; the formulas for the bias and the mean squared error (MSE) are also derived. In the simulations the MSE of our estimators, when $\overline{F}$ and $\overline{G}$ are known, are uniformly better than that of Lee, Yan, and Shi when the sample size is small(30); when the sample size is large, futher investigation is needed.

I. Introduction

In 1955, Lehmann introduced the stochastic ordering concept, which had played an important role in statistics. In particular, a related concept, stochastic ordering survival functions, had found applications in verious fields. We might be interested in estimating a survival function of a certain electrical insulating fluid subject to the level of voltage stress: the higher the voltage, the faster the material fails (see Rojo (1995)), or in corrosion engineer, estimating the rate of pit creation, which depends on the environments: the harsher they are, the higher the rate (Shibata and Takeyama (1977)). In medical science, doctors find it useful to know the survival times of patients who had heart pacemaker implanted. Since, it is "well documented" that females live longer than males, we might want to estimate the survival times of these patients under this constraint (see Dykstra (1982)).

[*]This project was partial sponsor by the NSF grant #... and NSA grant #...; we thank Pro. Rojo, and Mr. Covarrubias for guidance and assistance.

Let $\overline{F}$, $\overline{G}$, and $\overline{H}$ be survival functions with corresponding cummulative distribution functions, F, G, and H, respectively. Suppose these survival functions are of the stochastic order $\overline{F} \leq \overline{H} \leq \overline{G}$. The problems of estimating $\overline{H}$ when there is a one sided constraint, i.e., when $\overline{H} \leq \overline{G}$ or $\overline{F} \leq \overline{H}$ and $\overline{F}$ and $\overline{G}$ are either known or unknown, were considered by various researchers. Dykstra (1982) had found the nonparametric maximum likelihood estimator (NPMLE) for the problem; however, the estimator came in the form of an algorithm and, hence, rendered the difficulty of investigating the properties of the estimator. In dealing with such hassle, Rojo (1995) and Rojo and Ma (1996) had come up with closed-form estimators that converge weakly to the underlying survival functions. It was shown that these estimators have a smaller positive bias and mean squared error than the NPMLE. Their idea was to use the empirical survival functions and modify them where needed according to the constraint. It is well known that the empirical survival function is an unbiased estimator that converges strongly to the underlying survival function. We use the empirical survival function as the estimator when it satisfies the constraint; otherwise, we will use the boundary functions as the estimator. As the sample size increases the estimator will converge to the true function; thus, the estimator will essentially be the empirical survival function alone.

In 1999, Lee, Yan, and Shi had examined the one-sample problem with the two-sided constraint, i.e., to estimate $\overline{H}$ satisfying $\overline{F} \leq \overline{H} \leq \overline{G}$ when $\overline{F}$ and $\overline{G}$ are known. The NPMLE was, like Dykstra's, an algorithm and thus inherited the same drawbacks. Motivated by the work of Rojo (1995) and Rojo and Ma (1999), we use the same idea to find the estimator for the function in question.

In particular, if $\overline{F}$ and $\overline{G}$ are known, let $X_1, ..., X_n$ be an independent random sample from H, and let $\overline{H}_n$ be the empirical survival function, define the estimator $\widehat{\overline{H}}_n(x)$ on $\mathbf{R}$ by:

$$\widehat{\overline{H}}_n(x) = \begin{cases} \overline{F}(x), \, if \, \overline{H}_n(x) < \overline{F}(x); \\ \overline{H}_n(x), \, if \, \overline{F}(x) \leq \overline{H}_n(x) \leq \overline{G}(x); \\ \overline{G}(x), \, if \, \overline{H}_n(x) > \overline{G}(x). \end{cases} \tag{1.1}$$

We will show that $\widehat{\overline{H}}_n$ is strongly uniformly consistent and that it renders the empirical survival function inadmissible. Also through simulations, the root mean squared errors of the estimator defined in (1.1) were found to be smaller than that of the NPMLE suggested by Lee, Yan, and Shi (1999).

In the case where $\overline{F}$ and $\overline{G}$ are unknown, let $Y_1, ..., Y_m$, $X_1, ..., X_n$, and $Z_1, ..., Z_k$ be independent random samples from F, H, and G, respectively; also let $\overline{F}_m$, $\overline{H}_n$, and $\overline{G}_k$ be the empirical survival functions defined on the random samples, respectively.

$$\text{(1.2)} \qquad \widehat{\overline{F}}_{mk} = min(\overline{F}_m, \overline{G}_k)$$

and

$$\text{(1.3)} \qquad \widehat{\overline{G}}_{mk} = max(\overline{F}_m, \overline{G}_k)$$

The asymptotic behavior of the functions defined by (1.2) and (1.3) was investigated by Rojo (1995); specifically, $\widehat{\overline{F}}_{mk}$ and $\widehat{\overline{G}}_{mk}$ were proved to be strongly uniformly consistent estimators of $\overline{F}$ and $\overline{G}$. Now, define the estimator for $\overline{H}$ by:

$$\text{(1.4)} \qquad \widehat{\overline{H}}_{mnk}(x) = \begin{cases} \widehat{\overline{F}}_{mk}(x),\ if\, \overline{H}_n(x) < \widehat{\overline{F}}_{mk}(x); \\ \overline{H}_n(x),\ if\, \widehat{\overline{F}}_{mk}(x) \le \overline{H}_n(x) \le \widehat{\overline{G}}_{mk}(x); \\ \widehat{\overline{G}}_{mk}(x),\ if\, \overline{H}_n(x) > \widehat{\overline{G}}_{mk}(x). \end{cases}$$

The estimator defined above cannot be consistent if either m, n, or k stops short of infinity; however, when they all go to infinity, $\widehat{\overline{H}}_{mnk}$ converges to the underlying survival function.

II. The Bias, the MSE, and the Consistency of $\widehat{\overline{H}}_n$ on the One-Sample Problem with Known Boundaries

For the two sided problem with the constraint, such that, $\overline{F} \le \overline{H} \le \overline{G}$, satisfying the conditions for our estimator (1.1) is equivalent to letting the estimator for the c.d.f. H to be

$$\text{(2.1)} \qquad \widehat{H}_n(x) = \begin{cases} G(x),\ if\, H_n(x) < G(x); \\ H_n(x),\ if\, G(x) \le H_n(x) \le F(x); \\ F(x),\ if\, H_n(x) > F(x). \end{cases}$$

where $\widehat{H}_n(x) = 1 - \widehat{\overline{H}}_n(x)$.

The Bias of the estimator is

$$\begin{aligned} \text{(2.2)} \qquad Bias(\widehat{\overline{H}}_n(x)) &= E(\widehat{\overline{H}}_n(x) - \overline{H}(x)) \\ &= E\left[(1 - \widehat{H}_n(x)) - (1 - H(x))\right] \\ &= 1 - E(\widehat{H}_n(x)) - (1 - H(x)) \\ &= H(x) - E(\widehat{H}_n(x)) \end{aligned}$$

To calculate $E(\widehat{H}_n(x))$, note that given x, $nH_n(x)$ has Binomial $(n, H(x))$ distribution.

thus,

$$\begin{aligned}P(\widehat{H}_n(x) = G(x)) &= P(H_n(x) < G(x)) \\ &= P(nH_n(x) < nG(x)) \\ &= B(n;\ [nG(x)];\ H(x))\end{aligned}$$

where $[a]$ is the greatest integer that is smaller than or equal to a, and $B(n\,;\ a;\ H(x)) = P(x \leq a)$.

Similarly, we can obtain $P(\widehat{H}_n(x) = H_n(x))$ and $P(\widehat{H}_n(x) = F(x))$.

Thus, the Bias of the estimator is:

$$\begin{aligned}(2.3)\qquad Bias(\widehat{\overline{H}}_n(x)) &= H(x) - E(\widehat{H}_n(x)) \\ &= H(x) - \{G(x)B(n;\ [nG(x)];\ H(x)) \\ &+ \tfrac{1}{n}\textstyle\sum_{i=[nG(x)]+1}^{f-1} ib(n;\ i;\ H(x)) + F(x)\sum_{i=f}^{n}\ b(n;\ i:\ H(x))\Big\}\end{aligned}$$

where f is the smallest integer that is greatest or equal to $nF(x)$, and $b(n;\ i;\ H(x)) = P(x = i)$.

In a similar manner, we can find the MSE of the estimator as follows.

$$\begin{aligned}(2.4)\quad MSE(\widehat{\overline{H}}_n(x)) &= E\left[\left(\widehat{\overline{H}}_n(x) - \overline{H}(x)\right)^2\right] \\ &= E\left\{\left[(1 - \widehat{H}_n(x)) - (1 - H(x))\right]^2\right\} \\ &= E\left[\left(\widehat{H}_n(x) - H(x)\right)^2\right] \\ &= \Big\{[G(x) - H(x)]^2\, B(n;\ [nG(x)];\ H(x)) + \tfrac{1}{n^2}\textstyle\sum_{i=[nG(x)]+1}^{f-1} [i - nH(x)]^2\, b(n;\ i;\ H(x)) \\ &\quad + [F(x) - H(x)]^2 \textstyle\sum_{i=f}^{n}\ b(n;\ i;\ H(x))\Big\}\end{aligned}$$

Theorem 1. *The function defined in* (1.1) *is strongly uniformly consistent estimator of* $\overline{H}$.

PROOF: For all $x \in \mathbf{R}$,

$$(2.5)\quad \mid \widehat{\overline{H}}_n(x) - \overline{H}(x) \mid \leq \mid \overline{H}_n(x) - \overline{H}(x) \mid \leq \sup_{y\in R} \mid \overline{H}_n(y) - \overline{H}(y) \mid.$$

The result follows from the Glivenko-Cantelli Lemma (see Chung (1974), p. 133).

III. The Bias, the MSE, and the Consistency of $\widehat{\overline{H}}_{mnk}$ on the Two-Sample Problem with Unknown Boundaries

For the two sided problem in which $\overline{F}$ and $\overline{G}$ are unknown, the bias and the MSE of our estimator (1.4) can be found as the following.

Let

$$\widehat{G}_{mk} = min(F_m, G_k)$$

and

$$\widehat{F}_{mk} = max(F_m, G_k)$$

Then, (1.4) is equivalent to letting the estimator of the c.d.f. H to be

$$\widehat{H}_{mnk}(x) = \begin{cases} \widehat{G}_{mk}(x),\ if\, H_n(x) < \widehat{G}_{mk}(x); \\ H_n(x),\ if\, \widehat{G}_{mk}(x) \leq H_n(x) \leq F_{mk}(x); \\ \widehat{F}_{mk}(x),\ if\, H_n(x) > \widehat{F}_{mk}(x); \end{cases} \tag{3.1}$$

where $\widehat{H}_{mnk} = 1 - \widehat{\overline{H}}_{mnk}$.

Let I_A be the indicator function for the set A. Then,

$$\begin{aligned} Bias(\widehat{\overline{H}}_{mnk}(x)) &= H(x) - E(\widehat{H}_{mnk}(x)) \\ &= H(x) - \Big\{ E(\widehat{F}_{mk}(x) I_{\left\{H_n(x) > \widehat{F}_{mk}(x)\right\}} \\ +E\left(H_{n(x)} I_{\left\{\widehat{G}_{mk}(x) \leq H_n(x) \leq \widehat{F}_{mk}(x)\right\}} \right) &+ E\left(\widehat{G}_{mk}(x) I_{\{H_n(x) < G_{mk}(x)\}}\right) \Big\} \end{aligned} \tag{3.2}$$

where $\widehat{H}_{mnk} = 1 - \widehat{\overline{H}}_{mnk}$.

Let $[a]^* = \begin{cases} a-1;\ if\, a\, is\, an\, integer \\ [a]\,;\ otherwise. \end{cases}$

Then,

$$\begin{aligned} & E\left(\widehat{F}_{mk}(x)\, I_{\left\{H_n(x) > \widehat{F}_{mk}(x)\right\}}\right) \\ &= E\left(E\left(\widehat{F}_{mk}(x)\, I_{\left\{H_n(x) > \widehat{F}_{mk}(x)\right\}} \mid \widehat{F}_{mk}(x) = c\right)\right) \\ &= E\left(\widehat{F}_{mk}(x) P\left(H_n(x) > \widehat{F}_{mk}(x)\right) \mid \widehat{F}_{mk}(x) = c\right) \\ &= \textstyle\sum_c \left\{ [cP(H_n(x) > c]\, P\left(\widehat{F}_{mk}(x) = c\right)\right\} \\ &= \textstyle\sum_{j=1}^{m} \{ \frac{j}{m} P\left(nH_n(x) > \left[\frac{nj}{m}\right]\right) P\left(F_m(x) = \frac{j}{m}\right) P\left(G_k(x) \leq \frac{j}{m}\right) \} \\ &\quad + \textstyle\sum_{i=1}^{k} \left\{ \frac{i}{k} P\left(nH_n(x) > \left[\frac{ni}{k}\right]\right) P\left(G_k(x) = \frac{i}{k}\right) P\left(F_m(x) \leq \frac{i}{k}\right) \right\} \end{aligned}$$

since F and G are independent

$$= \sum_{j=1}^{m} \left\{ \frac{j}{m} \left(1 - B(n;\ \left[\frac{nj}{m}\right];\ H(x)\right) b\left(m;\ j;\ F(x)\right) B\left(k, \left[\frac{kj}{m}\right], G(x)\right) \right\}$$

$$+ \sum_{i=1}^{k} \left\{ \frac{i}{k} \left(1 - B(n;\ \left[\frac{ni}{k}\right];\ H(x)\right) b\left(k;\ i;\ G(x)\right) B\left(m;\ \left[\frac{mi}{k}\right];\ F(x)\right) \right\}$$

$$E\left(H_n(x)\, I_{\left\{ \widehat{G}_{mk}(x) \le H_n(x) \le \widehat{F}_{mk}(x) \right\}} \right)$$

$$= E\left\{ E\left(H_n(x)\, I_{\left\{ \widehat{G}_{mk}(x) \le H_n(x) \le \widehat{F}_{mk}(x) \right\}} \mid H_n(x) = c \right) \right\}$$

$$= E\left(H_n(x) P\left(\widehat{G}_{mk}(x) \le H_n(x)\, and\, \widehat{F}_{mk}(x) \ge H_n(x) \right) \mid H_n(x) = c \right)$$

$$= \sum_{i=0}^{n} \left\{ \left[\frac{i}{n} P\left(\widehat{G}_{mk}(x) \le \frac{i}{n} \right) P\left(\widehat{F}_{mk} \ge \frac{i}{n} \right) \right] P\left(H_n(x) = \frac{i}{n} \right) \right\}$$

$$= \sum_{i=0}^{n} \left\{ \left[\frac{i}{n} \left(1 - P\left(G_k(x) > \frac{i}{n}\, and\, F_m(x) > \frac{i}{n}\right)\right) \left(1 - P\left(G_k(x) < \frac{i}{n}\, and\, F_m(x) < \frac{i}{n}\right)\right) \right] b\left(n;\ i;\ H(x)\right) \right\}$$

$$= \sum_{i=0}^{n} \left\{ \frac{i}{n} \left[1 - \left(1 - B\left(k;\ \left[\frac{ki}{n}\right];\ G(x)\right)\right) \left(1 - B\left(m;\ \left[\frac{mi}{n}\right];\ F(x)\right)\right) \right] \right.$$

$$\left. \left[1 - B\left(k;\ \left[\frac{ki}{n}\right]^{*};\ G(x)\right) B\left(m;\ \left[\frac{mi}{n}\right]^{*};\ F(x)\right) \right] b\left(n;\ i;\ H(x)\right) \right\}$$

$$E\left(\widehat{G}_{mk}(x)\, I_{\left\{ H_n(x) < \widehat{G}_{mk}(x) \right\}} \right)$$

$$= E\left\{ E\left(\widehat{G}_{mk}(x)\, I_{\left\{ H_n(x) < \widehat{G}_{mk}(x) \right\}} \mid \widehat{G}_{mk}(x) = c \right) \right\}$$

$$= \sum_{c} \left\{ \left[cP\left(H_n(x) < c\right) \right] P\left(\widehat{G}_{mk}(x) = c \right) \right\}$$

$$= \sum_{j=1}^{m} \left\{ \frac{j}{m} P\left(nH_n(x) < \left[\frac{nj}{k}\right] \right) P\left(G_k(x) = \frac{j}{k} \right) P\left(F_m(x) \ge \frac{j}{k} \right) \right\}$$

$$+ \sum_{i=1}^{k} \left\{ \frac{i}{m} P\left(nH_n(x) < \left[\frac{ni}{k}\right] \right) P\left(G_k(x) = \frac{i}{k} \right) P\left(F_m(x) \ge \frac{i}{k} \right) \right\}$$

$$= \sum_{j=1}^{m} \left\{ \frac{j}{m} B\left(n; \left[\frac{nj}{k}\right]^{*};\ H(x) \right) b\left(m; j; F(x)\right) \left(1 - B\left(k;\ \left[\frac{kj}{m}\right]^{*};\ G(x) \right) \right) \right\}$$

$$= \sum_{i=1}^{k} \left\{ \frac{i}{k} B\left(n;\ \left[\frac{ni}{k}\right]^{*};\ H(x) \right) b\left(k;\ i;\ G(x)\right) \left(1 - B\left(m;\ \left[\frac{mi}{k}\right]^{*};\ F(x) \right) \right) \right\}$$

The MSE can be derived as the following.

$$MSE(\widehat{\overline{H}}_{mnk}(x)) = E\left[\left(\widehat{\overline{H}}_{mnk}(x) - \overline{H}(x) \right)^2 \right]$$

$$= E\left\{ \left[\left(1 - \widehat{H}_{mnk}(x)\right) - \left(1 - H(x)\right) \right]^2 \right\}$$

$$= \left[\left(H(x) - \widehat{H}_{mnk}(x)\right)^2\right]$$

$$= E\left(\left(H(x) - \widehat{F}_{mk}(x)\right)^2 I_{\left\{H_n(x) > \widehat{F}_{mk}(x)\right\}}\right)$$

$$+E\left((H(x) - H_n(x))^2 I_{\left\{\widehat{G}_{mk}(x) \leq H_n(x) \leq \widehat{F}_{mk}(x)\right\}}\right)$$

$$= E\left(\left(H(x) - \widehat{G}_{mk}(x)\right)^2 I_{\left\{H_n(x) < \widehat{G}_{mk}(x)\right\}}\right)$$

Since $H(x)$ is a constant, for a given x, the MSE for $\widehat{\overline{H}}_{mnk}$ can be derived by replacing

$\widehat{F}_{mk}(x)$ with $\left(H(x) - \widehat{\overline{F}}_{mk}(x)\right)^2$, $H_n(x)$ with $(H(x) - H_n(x))^2$, and $\widehat{G}_{mk}$ with $\left(H(x) - \widehat{\overline{G}}_{mk}(x)\right)^2$

in the derivation for the $Bias(\widehat{\overline{H}}_{mnk}(x))$.

Theorem 2. *The function defined in* (1.4) *is strongly uniformly consistent estimator of* $\overline{H}$.

PROOF: (1.4) can be written as

$$(3.3)\ \widehat{\overline{H}}_{mnk}(x) = \widehat{\overline{F}}_{mk}(x) I_{\left\{\overline{H}_n(x) < \widehat{\overline{F}}_{mk}(x)\right\}} + \overline{H}_n(x) I_{\left\{\widehat{\overline{F}}_{mk}(x) \leq \overline{H}_n(x) \leq \widehat{\overline{G}}_{mk}(x)\right\}}$$
$$+ \widehat{\overline{G}}_{mk}(x) I_{\left\{\overline{H}_n(x) > \widehat{\overline{G}}_{mk}(x)\right\}}$$

Assume that $\overline{F} < \overline{H} < \overline{G}$. Then for an arbitrary x, let $\overline{H}(x) - \overline{F}(x) = \varepsilon > 0$.

Since $\overline{H}_n$ converges to $\overline{H}$ as $n \to \infty$,$\widehat{\overline{F}}_{mk}$ converges to $\overline{F}$ as m and $k \to \infty$

(Rojo (1995)), then $\exists\, M$, N, and K such that $m \geq M$, $n \geq N$, and $k \geq K$,

$\Rightarrow \ \mid \overline{H}_n(x) - \overline{H}(x) \mid < \frac{\varepsilon}{2}$, and $\mid \overline{F}_{mk}(x) - \overline{F}(x) \mid < \frac{\varepsilon}{2}$.

Thus, when $m \geq M$, $n \geq N$, and $k \geq K$,

$\overline{F}_{mk}(x) - \overline{H}_n(x) + \overline{H}(x) - \overline{F}(x) = \overline{F}_{mk}(x) - \overline{F}(x) + \overline{H}(x) - \overline{H}_n(x)$

$\leq \mid \overline{F}_{mk}(x) - \overline{F}(x) \mid + \mid \overline{H}_n(x) - \overline{H}(x) \mid < \frac{\varepsilon}{2} + \frac{\varepsilon}{2} = \varepsilon.$

Hence, $\overline{H}_n(x) - \overline{F}_{mk}(x) > 0$.

As a result, $\left\{\overline{H}_n(x) < \widehat{\overline{F}}_{mk}(x)\right\}$ becomes empty, as n, m and $k \to \infty$.

Similarly, $\left\{\overline{H}_n(x) > \widehat{\overline{G}}_{mk}(x)\right\}$ becomes empty, as n, m and $k \to \infty$.

Therefore, $I_{\left\{\overline{H}_n(x) < \widehat{\overline{F}}_{mk}(x)\right\}} \to 0$, and $I_{\left\{\overline{H}_n(x) > \widehat{\overline{G}}_{mk}(x)\right\}} \to 0$, as n, m and $k \to \infty$.

Thus, $\widehat{\overline{H}}_{mnk} \to \overline{H}_n$ almost everywhere, as n, m, and $k \to \infty$. The proof is complete, for $\overline{H}_n \to H$ with probability one, as $n \to \infty$.

If there exists an x such that $\overline{H}(x) = \overline{F}(x)$ or $\overline{H}(x) = \overline{G}(x)$ then ....(we provide the proof if we figure it out later!)

IV. Simulation Studies

In order to analyze the differences between our proposed estimator and the NPMLE offered by Lee, Yan, and Shi (1991), we ran simulations for these two estimators. The simulations are run for two purposes.

The first reason for us to run the simulations is to compare the simulation results for the NPMLE with the data reported by Lee, Yan, and Shi (1999). These results are important because they can either confirm or deny Lee, Yan, and Shi's findings, and the comparison between our estimator and Lee, Yan, and Shi's makes sense only if their data is accurate. If the simulations show that there is an error in the published data, we will use our simulation results for the NPMLE do the comparison.

The main purpose of the simulations is to find out which of Lee, Yan, and Shi's and our estimators is better. The $\sqrt{MSE}$, which measures the accuracy of an estimator, is being used. By running simulations we are able to estimate the $\sqrt{MSE}$ for the NPMLE and compare them with those published by Lee, Yan, and Shi (1999). The $\sqrt{MSE}$ is used in place of MSE because it was the quantity used in Lee, Yan, and Shi's paper to measure the accuracy of the NPMLE. Also, since both estimators are good and return the MSE's in the magnitude of the ten thousandths, the use of the $\sqrt{MSE}$ avoids cluttering the data with unnecessary zeros.

In Lee, Yan, and Shi's paper , there are four cases of simulations run and the $\sqrt{MSE}$'s were presented. Due to time constraints, we are able to analyze only two cases.

For the first case, we have:

$\overline{H}(t) = exp(-t)$ with stochastic bounds

$\overline{F}(t) = exp(-t/0.8)$, no upper bound.

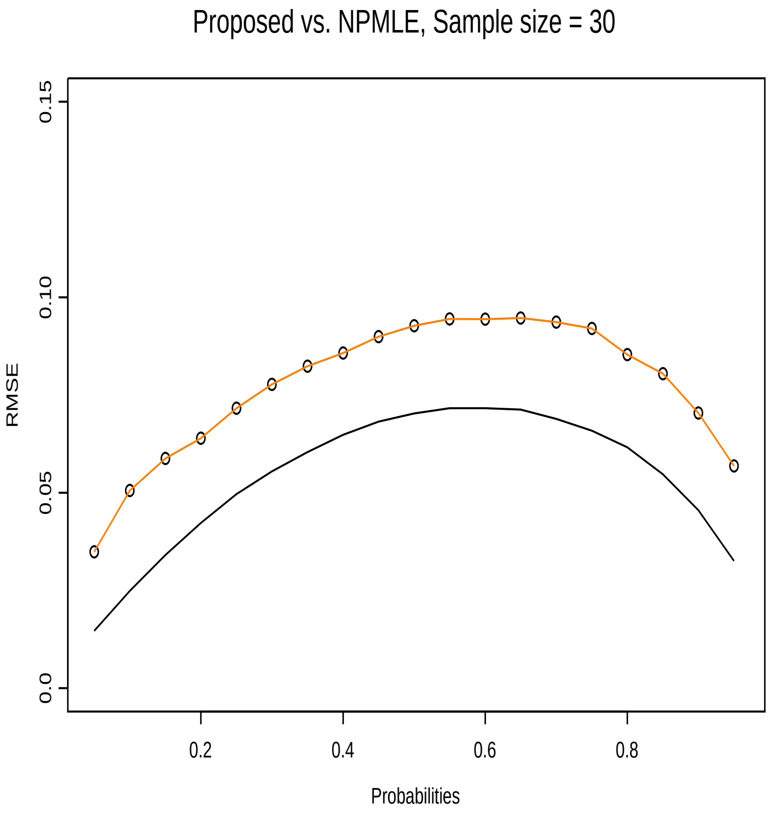
One-Sided Problem:

For the one-sided simulations, the group worked with the case above. We ran 10,000 simulations with a sample size of 100 and found that the reported results were not only accurate but higher than the results from the simulations. After running 10,000 simulations on our estimator for the one-sided problem, we found that from quantiles zero through .25 (with step of .05) our estimator had a smaller MSE than the NPMLE. From quantiles .25 through .8 the NPMLE had a smaller MSE and from .8 through 1 the two estimators were almost identical. These results were somewhat surprising because Rojo and Ma (1996) had concluded otherwise. Therefore, we ran both simulators 20,000 times, 10,000 simulations each for a sample size of 20 and 10,000 simulations each for a sample size of 30. As expected, we found that the proposed estimator had a smaller MSE than the NPMLE uniformly. In the case of sample size of 20, the greatest difference in the $\sqrt{MSE}$ is .015684286 and on average the proposed estimator has a smaller $\sqrt{MSE}$ by .013327239. In the case of sample size of 30, the greatest difference in $\sqrt{MSE}$ is .00864384 and on average the proposed estimator has a smaller $\sqrt{MSE}$ by .0073919. But in the case of sample size of 100, the greatest difference in $\sqrt{MSE}$ is .002010813, with the NPMLE being smaller. On average, for this case, the NPMLE has a smaller $\sqrt{MSE}$ by .00075748. The MSE of this greatest difference is $4.04 * 10^{-6}$ and the MSE of the average is $5.74 * 10^{-7}$. These numbers are so small that for all practical purposes, the proposed estimator is just as accurate as the NPMLE. In conclusion, for the one-sided problem the proposed estimator is as accurate as the NPMLE for large sample sizes and more accurate for small sample sizes.

Two-Sided Problem:

In the following table, we list the $\sqrt{MSE}$**'s** that were calculated theoretically for our estimator, along with the $\sqrt{MSE}$**'s** for the NPMLE. Due to times constraint, we just run the simulation, using Lee, Yan, and Shi's algorithm, to estimate $\overline{H}(x) = exp(-x)$, given $\overline{F}(x) = exp(-x/0.8)$ and $\overline{G}(x) = exp(-x/1.2)$ with sample size of 30, and the size of the simulation is 4550 times. The results are given in table below where the differences are the discrepancies between $\sqrt{MSE}$**'s** of the estimator defined in (1.1) and the NPMLE.

Size 30									
	$t_{.10}$	$t_{.20}$	$t_{.30}$	$t_{.40}$	$t_{.50}$	$t_{.60}$	$t_{.70}$	$t_{.80}$	$t_{.90}$
Est. (Theo.)	.0249	.0423	.0554	.0648	.0703	.0716	.0689	.0616	.0455
NPMLE	.0506	.0640	.0777	.0857	.0927	.0944	.0936	.0853	.0704
Differences	-.0257	-.0217	-.0223	-.0209	-.0224	-.0228	-.0247	-.0237	-.0249



Proposed vs. NPMLE, Sample size = 30

The result shows that the estimator defines in (1.1) is uniformly better than the NPMLE in this case; however, we suspect that for a large sample size, our estimator converges to the empirical survival function, and thus might not preserve this property. Further investigation is needed to determine whether it is true.

V. Appendix

(a) One-Sided Problem Code:

Lee, Yan, and Shi's Algorithm:

```
 HR<- function(sample)
{
        MSE<-NULL
        EMSE<-NULL
        sumtotal1<-numeric(19)
        sumsqtotal1<-numeric(19)
        sumtotal2<-numeric(19)
        sumsqtotal2<-numeric(19)
        for(w in 1:1000)
        {

        vector<-rexp(sample)
        S<-sort(vector)
        a1<-NULL
        for(i in 1:length(vector))
        {
                a1[i]<- 1-(1-exp(-vector[i]*1.25))
        }
        a<-sort(a1)
        b<-a[length(a):1]
        c<-(length(a):1)
        d<-c((length(c)-1):1,0)
        e<-c(1,b[1:(length(b)-1)])
        f<-cbind(c,e,d,b) #up to here we are just ordering data
        x<-dim(f)[1]
        B<- matrix(,x,x) #gives us the yab matrix
        for (i in 1:x)
                for(j in 1:x)
                {B[i,j]<-(f[i,1]*f[j,4]-f[j,3]*f[i,2])/(f[i,2]-f[j,4])}
        totalystar<-NULL
        ystar<-NULL
        for(number in 1:nrow(B))
        {
        for(i in 1:number)
        {
                AVECTOR<-B[i,number:ncol(B)]
                theMax<-AVECTOR[1]
                for(i in 1:length(AVECTOR))
                {
                        if(theMax<=AVECTOR[i])
                        {
                                theMax<-AVECTOR[i]
                        }
                }
                ystar[i]<-theMax
        }

        #gives us the vector of all ystars
        totalystar[number]<-min(ystar)
}
        combo<- 1-(1/(c+totalystar))
        constants<-c(0.05129329,0.10536052,0.16251893,0.22314355,
0.28768207,0.35667494,0.43078292,0.51082562,0.59783700,
0.69314718,0.79850770,0.91629073,1.04982212,1.20397280,
1.38629436,1.60943791,1.89711998,2.30258509,2.99573227)
        theIndex<-NULL
        for(k in 1:length(constants))
        {
                for(i in 1:length(S))
                {
                        if(constants[k]>=S[i])
                        {theIndex[k]<-(i)}
                }
                theIndex
        }
        estimator<- NULL
        for(i in 1:length(theIndex))
        {
                if(theIndex[i]=='NA')
                {estimator[i]<-1}
                else
                {estimator[i]<-prod(combo[1:theIndex[i]])}
        }
                empirical<-NULL
                counter<-numeric(19)
                for (j in 1:length(constants))
                for (i in 1:length(S))
                {
                   if(S[i]<=constants[j]){counter[j]<-counter[j]+1}
                }
                p<-seq(.05,.95,.05)
                empirical<-counter/sample
                Fx<- 1-estimator
                sumtotal1<-sumtotal1+Fx
                sumsqtotal1<-sumsqtotal1+(Fx)^2
                sumtotal2<-sumtotal2+empirical
                sumsqtotal2<-sumsqtotal2+(empirical)^2
}
```

```
MSE<-(sumsqtotal1/1000)-(2*p*(sumtotal1/1000))+(p^2)
EMSE<-(sumsqtotal2/1000)-(2*p*(sumtotal2/1000))+(p^2)
list(EMSE,MSE)
```

Our Simulation Code:

```
Our2sample<-function(sample)
{
        sumtotal<-numeric(19)
        sumsqtotal<-numeric(19)
        for(w in 1:1000)
        {
        a<-rexp(sample)
        a<-sort(a)#gives random numbers in small to big
        xps<- c(.05129329,.10536052,.16251893,.22314355,.28768207,.35667494,
.43078292,.51082562,.59783700,.69314718,.79850770,.91629073,
1.04982212,1.20397280,1.38629436,1.60943791,1.89711998,
2.30258509,2.99573227)
        qxp<-NULL
        for(i in 1:length(xps))
        {
                qxp[i]<- exp(-xps[i]/.8)
        }
        empirical<-NULL
        counter<-numeric(19)
        for(j in 1:length(xps))
                for(i in 1:length(a))
                {
                        if(a[i]<=xps[j]){counter[j]<-counter[j]+1}
                }
        empirical<-(1-(counter/sample))
        S<-numeric(19)
        for(i in 1:length(qxp))
        {
                S[i]<-max(empirical[i],qxp[i])
        }
        p<-seq(.05,.95,.05)
        sumtotal<-sumtotal+S
        sumsqtotal<-sumsqtotal+(S^2)
}
        MSE<-(sumsqtotal/1000)-(2*(1-p)*(sumtotal/1000))+((1-p)^2)
        MSE
```

(b) Two-Sided Problem Code:

Lee, Yan, and Shi's Algorithm:

```
{
        # Enter the sample size here
        sampleSize <- 100
        # Enter how many run through here
        numSimulation <- 10

        MSE <- NULL
        EMSE <- NULL
        sumtotal1 <- 0
        sumtotal2 <- 0
        sumsqtotal1 <- 0
        sumsqtotal2 <- 0
        for(i in 1:numSimulation) {
        ############## Generate data and initialize ##########
        theQ <- NULL
        theR <- NULL
        theQr <- NULL
        theRr <- NULL
        theVector <- rexp(sampleSize)
        theVector <- sort(theVector)
        for(i in 1:sampleSize) {
                theQ[i] <- 1 - (1 - exp( - theVector[i] * 1.25))
        }
        for(i in 1:sampleSize) {
                theRr[i] <- theR[i] <-
                         1 - (1 - exp( - theVector[i] * (1/1.2)))
        }
        ############### Step 2r + 1 ##########################
        partLength <- 0
        thePartition <- c(0, sampleSize)
        totalystar <- numeric(sampleSize)
        while(partLength != length(thePartition)) {
           #update partLength
           partLength <- length(thePartition)
           for(index in 1:(length(thePartition) - 1)) {
               alpha1 <- thePartition[index]
               alpha2 <- thePartition[index + 1]
               for(i in (alpha1 + 1):alpha2) {
                     theQr[i] <- max(theQ[i], theR[alpha2])
               }
               C <- c(length(theQr):1)
               theQr <- sort(theQr)
               theQr <- theQr[length(theQr):1]
```

```
d <- c(1, theQr[1:length(theQr) - 1])
qst <- matrix(NULL, length(theQr), length(theQr))
#CALCULATE Y[s,t]
for(s in 1:length(theQr))
  for(t in s:length(theQr)) {
    theFunc <- function(y)
    {
       sum(log(1 - (1/(C[s:t] + y)))) - log(theQr[t]/d[s])
    }
      qst[s, t] <- uniroot(theFunc, c(((d[s]/(d[s] - theQr[t]) -
    }
            qst <- t(qst)
            #Now is the part where we solve the min max thing with
            #caveat of yi=max(yi,0) when alpha2=m
            theMax <- NULL
            ystar <- NULL
            for(i in (alpha1 + 1):alpha2) {
                    for(j in (alpha1 + 1):i) {
                            theMax[j] <-
                               max(qst[i:alpha2, j], na.rm = TRUE)
                    }
                    totalystar[i] <- min(theMax, na.rm = TRUE)
                    # reset ystar when alpha2 = m
                    if(alpha2 == sampleSize) {
                            totalystar[i] <- max(totalystar[i], 0)
                    }
            }
            # Add partition points
            for(i in 1:(length(totalystar) - 1)) {
                    if(totalystar[i] > totalystar[i + 1]) {
                            thePartition <- c(thePartition, i)
                    }
            }
            thePartition <- sort(thePartition)
            aVector <- c(thePartition[length(thePartition)])
            for(i in 1:(length(thePartition) - 1)) {
                    if(thePartition[i] != thePartition[i + 1])
                            aVector <- c(aVector, thePartition[i])
            }
            thePartition <- sort(aVector)
    }
    ###################### Step 2r + 2 ############################
    for(index in 1:(length(thePartition) - 1)) {
            alpha1 <- thePartition[index]
            alpha2 <- thePartition[index + 1]
            for(i in (alpha1 + 1):alpha2) {
```

```
        theRr[i] <- min(theR[i], theQr[alpha1])
}
C <- c(length(theRr):1)
theRr <- sort(theRr)
theRr <- theRr[length(theRr):1]
d <- c(1, theRr[1:length(theRr) - 1])
qst2 <- matrix(NULL, length(theRr), length(theRr))
for(s in 1:length(theRr))
        for(t in s:length(theRr)) {
                theFunc1 <- function(y)
                {
                  sum(log(1 - (1/(C[s:t] + y)))) -
                  log(theRr[t]/d[s])
                }
                qst2[s, t] <- uniroot(theFunc1,
                              c(((d[s]/(d[s]
                              - theRr[t]) - C[t]))
                              - 0.0001, 1000000))$root
        }
qst2 <- t(qst2)
#Now is the part where we solve the min max thing with
#caveat of yi=max(yi,0) when alpha2=m
theMin <- NULL
ystar <- NULL
for(i in (alpha1 + 1):alpha2) {
        for(j in (alpha1 + 1):i) {
           theMin[j] <-
              min(qst2[i:alpha2, j], na.rm = TRUE)
        }
        totalystar[i] <- max(theMin, na.rm = TRUE)
        # reset ystar when alpha2 = m
        if(alpha2 == sampleSize) {
                totalystar[i] <- min(totalystar[i], 0)
        }
}
for(i in 1:(sampleSize - 1)) {
        if(totalystar[i] > totalystar[i + 1]) {
                thePartition <- c(thePartition, i)
        }
}
thePartition <- sort(thePartition)
aVector <- c(thePartition[length(thePartition)])
for(i in 1:(length(thePartition) - 1)) {
        if(thePartition[i] != thePartition[i + 1])
                aVector <- c(aVector, thePartition[i])
}
```

```
                        thePartition <- sort(aVector)
                }
                #return values
                totalystar
        }
        c <- sampleSize:1
        combo <- 1 - (1/(c + totalystar))
        constants <- c(0.05129329, 0.10536052, 0.16251893, 0.22314355,
                      0.28768207, 0.35667494, 0.43078292, 0.51082562,
                      0.597837, 0.69314718, 0.7985077,0.91629073,
                      1.04982212, 1.2039728, 1.38629436, 1.60943791,
                      1.89711998, 2.30258509, 2.99573227)
        theIndex <- NULL
        for(k in 1:length(constants)) {
                for(i in 1:sampleSize) {
                        if(constants[k] >= theVector[i]) {
                                theIndex[k] <- (i)
                        }
                }
                theIndex
        }
        estimator <- NULL
        for(i in 1:length(theIndex)) {
                if(theIndex[i] == "NA") {
                        estimator[i] <- 1
                }
                else {
                        estimator[i] <- prod(combo[1:theIndex[i]])
                }
        }
        empirical <- NULL
        counter <- numeric(19)
        for(j in 1:length(constants))
                for(i in 1:sampleSize) {
                        if(theVector[i] <= constants[j]) {
                           counter[j] <- counter[j] + 1
                        }
                }
        p <- seq(0.05, 0.95, 0.05)
        empirical <- counter/sampleSize
        Fx <- 1 - estimator
        sumtotal1 <- sumtotal1 + Fx
        sumsqtotal1 <- sumsqtotal1 + (Fx)^2
        sumtotal2 <- sumtotal2 + empirical
        sumsqtotal2 <- sumsqtotal2 + (empirical)^2
}
```

```
   MSE <- (sumsqtotal1/numSimulation)
          - (2 * p * (sumtotal1/numSimulation)) + (p^2)
   EMSE <- (sumsqtotal2/numSimulation)
          - (2 * p * (sumtotal2/numSimulation)) + (p^2)
   list(MSE, EMSE)
}
```

Our Codes to Calculate the Proposed MSE Theoretically:

```
#Bias & MSE for Case3 B1
#Sample size=100

p<-seq(0.05,.95,.05)
p
length(p)
t<- -log(1-p)
t
fcdf<-pexp(t,1/.8)
fcdf
nfcdf<-100*fcdf
nfcdf
A<-c(1:100)
A
n<-numeric(length(nfcdf))
for (i in 1:length(nfcdf))
{
        n[i]=100-length(which(A>=nfcdf[i]))
        }
n
gcdf<-pexp(t,1/1.2)
gcdf
mgcdf<-100*gcdf
mgcdf
m<-numeric(length(mgcdf))
for (j in 1:length(mgcdf))
{
        m[j]=length(which(A<=mgcdf[j]))

}
m
EVHhat<-numeric(length(p))
for (i in 1:length(p))
{
        S<-seq(m[i]+1,(n[i]),1)
        partialsum<-sum(S*dbinom(S,100,p[i]))
        EVHhat[i]=fcdf[i]*(1-pbinom(n[i],100,p[i]))
                  +1/100*partialsum+gcdf[i]
                  *pbinom(m[i],100,p[i])

}
EVHhat
Bias=p-EVHhat
Bias
MSE<-numeric(length(p))
for (i in 1:length(p))
{
        Sprime<-seq(m[i]+1,(n[i]),1)
        partialsum<-sum(((Sprime-100*p[i])^2)
                    *dbinom(Sprime,100,p[i]))
        MSE[i]=((fcdf[i]-p[i])^2)*(1-pbinom(n[i],100,p[i]))
       +((1/100)^2)*partialsum+((gcdf[i]-p[i])^2)
       *pbinom(m[i],100,p[i])

}
MSE
SQRofMSE<-sqrt(MSE)
SQRofMSE
```

References

[1] Chung, K. L. (1974). A Course in Probability Theory. Academic Press, San Diego.

[2] Dykstra, R. L. (1982). ”Maximum likelihood estimation of the survival functions of two stochastically ordered random variables”, J. Amer. Stat. Assoc., 77, 621-628.

[3] Lee, C. C., Yan, X. and Shi, N. (1999). ”Nonparametric estimation of bounded survival functions with censored observations”, Lifetime Data Analysis, 5, 81-90.

[4] Lehmann, E. L. (1955). ”Ordered families of distributions”, Ann. Math. Statist., 26, 399-419.

[5] Rojo, J. (1995). ”On the weak convergence of certain estimators of stochastically ordered survival functions”, J. Nonparam. Stat., Vol 4, No 4, 349-363.

[6] Rojo, J. and Ma, Z. (1996). ”On the estimation of stochastically ordered survival functions”, J. Statist. Comput. Simulation, 55, 1-21.

[7] Shibata, T. and Takeyama, T. (1977). "Stochastic Theory of Pitting Corrosion". Corrosion, 33, No. 7, 243.