2010

COMPARISON OF TWO SAMPLES BY A NONPARAMETRIC LIKELIHOOD-RATIO TEST

William H. Barton
University of Kentucky, phos@windstream.net

ABSTRACT OF DISSERTATION

William H. Barton

The Graduate School
University of Kentucky
2010

COMPARISON OF TWO SAMPLES BY A NONPARAMETRIC LIKELIHOOD-RATIO TEST

ABSTRACT OF DISSERTATION

A dissertation submitted in partial
fulfillment of the requirements for
the degree of Doctor of Philosophy
in the College of Arts and Sciences
at the University of Kentucky

By
William H. Barton
Lexington, Kentucky

Director: Dr. Mai Zhou, Professor of Statistics
Lexington, Kentucky 2010

ABSTRACT OF DISSERTATION

COMPARISON OF TWO SAMPLES BY A NONPARAMETRIC LIKELIHOOD-RATIO TEST

In this dissertation we present a novel computational method, as well as its software implementation, to compare two samples by a nonparametric likelihood-ratio test. The basis of the comparison is a mean-type hypothesis. The software is written in the R-language [4]. The two samples are assumed to be independent. Their distributions, which are assumed to be unknown, may be discrete or continuous. The samples may be uncensored, right-censored, left-censored, or doubly-censored. Two software programs are offered. The first program covers the case of a single mean-type hypothesis. The second program covers the case of multiple mean-type hypotheses. For the first program, an approximate p-value for the single hypothesis is calculated, based on the premise that -2log-likelihood-ratio is asymptotically distributed as $\chi^2_{(1)}$. For the second program, an approximate p-value for the p hypotheses is calculated, based on the premise that -2log-likelihood-ratio is asymptotically distributed as $\chi^2_{(p)}$. In addition we present a proof relating to use of a hazard-type hypothesis as the basis of comparison. We show that -2log-likelihood-ratio is asymptotically distributed as $\chi^2_{(1)}$ for this hypothesis. The R programs we have developed can be downloaded free-of-charge on the internet at the Comprehensive R Archive Network (CRAN) at `http://cran.r-project.org`, package name *emplik2*. The R-language itself is also available free-of-charge at the same site.

KEYWORDS: Likelihood, Ratio, Nonparametric, Hypothesis, Hazard

Author's signature: William H. Barton

Date: April 23, 2010

COMPARISON OF TWO SAMPLES BY A NONPARAMETRIC LIKELIHOOD-RATIO TEST

By
William H. Barton

Director of Dissertation: Mai Zhou

Director of Graduate Studies: Dr. William Griffith

Date: April 23, 2010

RULES FOR THE USE OF DISSERTATIONS

Unpublished dissertations submitted for the Doctor's degree and deposited in the University of Kentucky Library are as a rule open for inspection, but are to be used only with due regard to the rights of the authors. Bibliographical references may be noted, but quotations or summaries of parts may be published only with the permission of the author, and with the usual scholarly acknowledgments.

Extensive copying or publication of the dissertation in whole or in part also requires the consent of the Dean of the Graduate School of the University of Kentucky.

A library that borrows this dissertation for use by its patrons is expected to secure the signature of each user.

Name	Date

DISSERTATION

William H. Barton

The Graduate School
University of Kentucky
2010

COMPARISON OF TWO SAMPLES BY A NONPARAMETRIC LIKELIHOOD-RATIO TEST

DISSERTATION

A dissertation submitted in partial
fulfillment of the requirements for
the degree of Doctor of Philosophy
in the College of Arts and Sciences
at the University of Kentucky

By
William H. Barton
Lexington, Kentucky

Director: Dr. Mai Zhou, Professor of Statistics
Lexington, Kentucky 2010

To my wife Elaine, for her encouragement and support.

Many daughters have done nobly; but you excel them all.

ACKNOWLEDGMENTS

I would like to thank my advisor, Dr. Mai Zhou, who first introduced me to the field of empirical likelihood. My gratitude also goes to Drs. Arne Bathke, William Griffith, Richard Charnigo, and Robert Molzon, who served on my dissertation committee. I extend my thanks as well to the numerous professors whose classes I attended in the doctoral program at the University of Kentucky. Their enthusiasm for the field of statistics and their great learning are a continual inspiration to me.

Finally I would like to honor my parents, Francis William Barton, Jr. and Yorke Hargis Barton, who raised me and who funded my undergraduate education in the Chemical Engineering curriculum at the Virginia Polytechnic Institute.

TABLE OF CONTENTS

LIST OF FIGURES

Chapter 1 Outline of the Dissertation

In this dissertation we present a novel computational method, as well as its software implementation, to compare two samples by a nonparametric likelihood-ratio test. The basis of the comparison is a mean-type hypothesis. The software is written in the R-language [4].

The two samples are assumed to be independent. Their distributions, which are assumed to be unknown, may be discrete or continuous. The samples may be uncensored, right-censored, left-censored, or doubly-censored.

Two software programs are offered. The first program covers the case of a single mean-type hypothesis. The second program covers the case of multiple mean-type hypotheses.

For the first program, an approximate p-value for the single hypothesis is calculated, based on the premise that -2log-likelihood-ratio is asymptotically distributed as $\chi^2_{(1)}$. For the second program, an approximate p-value for the p hypotheses is calculated, based on the premise that -2log-likelihood-ratio is asymptotically distributed as $\chi^2_{(p)}$.

In addition we present a proof relating to use of a hazard-type hypothesis as the basis of comparison. We show that -2log-likelihood-ratio is asymptotically distributed as $\chi^2_{(1)}$.

The R programs we have developed can be downloaded free-of-charge on the inter-

net at the Comprehensive R Archive Network (CRAN), `http://cran.r-project.org` package name *emplik2*. The R-language itself is also available free-of-charge at the same site.

An outline of the development of the computational method and the two software programs is as follows:

- In Chapter 2 we introduce and discuss the nonparametric likelihood-ratio test.
- In Chapter 3 we derive the equations and algorithm that underlie the first program (single-hypothesis case).
- In Chapter 4 we derive the equations and algorithm that underlie the second program (multiple-hypothesis case).
- In Chapter 5 we give examples of how the programs can be used.
- In Chapter 6 we present a mathematical derivation for the case where a hazard-type hypothesis is used.
- In Chapter 7 we present a proof that the estimated probability jumps calculated by the two programs will be non-negative.
- In Chapter 8 we suggest ways that this dissertation could be extended in the future.
- In Appendix A we list the annotated R-code for the two programs.
- In Appendix B we list the annotated R-code for the simulations used in the dissertation.

Chapter 2 Introduction

This chapter briefly reviews the parametric likelihood-ratio test and the nonparametric likelihood-ratio test.

2.1 Parametric Likelihood-Ratio Test

This section describes the parametric likelihood-ratio test. It largely follows the discussion in Casella and Berger (2002) [1], pp. 374-375.

Suppose we have an uncensored random sample

$$\boldsymbol{x} = (x_1, x_2, \ldots, x_n) \tag{2.1}$$

with a common pdf or pmf $f(x|\theta)$, where the parameter θ (which can be a vector) is an element of the parameter space Θ. Then the likelihood function $L(\theta|\boldsymbol{x})$ is defined as

$$L(\theta|\boldsymbol{x}) = \prod_{i=1}^{n} f(x_i|\theta) \ . \tag{2.2}$$

The likelihood-ratio statistic $\lambda(\boldsymbol{x})$ for testing the set of hypotheses

$$H_o : \theta \in \Theta_o \tag{2.3}$$

$$H_A : \theta \in \Theta_o^c \tag{2.4}$$

is defined as

$$\lambda(\boldsymbol{x}) = \frac{\sup\limits_{\theta \in \Theta_o} L(\theta|\boldsymbol{x})}{\sup\limits_{\theta \in \Theta} L(\theta|\boldsymbol{x})} \ . \tag{2.5}$$

We can evaluate the hypothesis H_o in (2.3) using a likelihood-ratio test (LRT), which takes the following form:

$$\text{Reject } H_o \text{ when } \lambda(\boldsymbol{x}) \leq c, \text{ where } 0 \leq c \leq 1 \ . \tag{2.6}$$

Consider the right-hand side (RHS) of (2.5). Let us denote the unrestricted maximum-likelihood estimator of θ, where $\theta \in \Theta$, as $\widehat{\theta}$. Let us denote the restricted maximum-likelihood estimator of θ, where $\theta \in \Theta_o$, as $\widehat{\theta}_o$. Now the MLE is invariant, that is, if $\widehat{\theta}$ is the MLE of θ, then for any function $g(\theta)$, the MLE of $g(\theta)$ is $g(\widehat{\theta})$ (see Casella and Berger (2002) [1] p. 320). Therefore by the invariance property of the MLE , (2.5) can be written as,

$$\lambda(\boldsymbol{x}) = \frac{L(\widehat{\theta}_o|\boldsymbol{x})}{L(\widehat{\theta}|\boldsymbol{x})} \tag{2.7}$$

It is convenient to apply the (natural) log function to the inequality in (2.6). The log function preserves the inequality since log is a monotonic function. Then the test (2.6) takes the following form:

$$\text{Reject } H_o \text{ when } \log \lambda(\boldsymbol{x}) \leq \log c, \text{ where } -\infty < \log c \leq 0 \ . \tag{2.8}$$

And from (2.7) we see that $\log \lambda(\boldsymbol{x})$ has the following simple form:

$$\log \lambda(\boldsymbol{x}) = \log L(\widehat{\theta_o}|\boldsymbol{x}) - \log L(\widehat{\theta}|\boldsymbol{x}) \tag{2.9}$$

where from (2.2),

$$\log L(\widehat{\theta_o}|\boldsymbol{x}) = \sum_{i=1}^{n} \log f(x_i|\widehat{\theta_o}) \tag{2.10}$$

$$\log L(\widehat{\theta}|\boldsymbol{x}) = \sum_{i=1}^{n} \log f(x_i|\widehat{\theta}) \ . \tag{2.11}$$

Wilks (1938) [20] showed that if H_o is true, then $-2 \log \lambda(\boldsymbol{x})$ has an asymptotic $\chi^2_{(p)}$ distribution (under certain regularity conditions), where p is the number of restrictions imposed on the parameters by H_o. This $\chi^2_{(p)}$ distribution can be used to select a meaningful value for c in (2.6).

Advantages of the parametric LRT are as follows:

1. Confidence sets obtained by this method are transformation-invariant. That is, if a confidence set for a parameter θ is $\{\theta : -2 \log \lambda(\boldsymbol{x}) \leq c\}$, then a confidence set for $g(\theta)$ is $\{g(\theta) : -2 \log \lambda(\boldsymbol{x}) \leq c\}$, where $g(\cdot)$ is a function.

2. It is not necessary to construct a variance-covariance matrix in order to form a confidence set for a parameter θ.

3. The confidence set for a parameter $\theta \in \Theta$ always falls within Θ, even when θ is close to the boundary of Θ. (By contrast, for example, it's possible for a parametric t-test to place θ partially outside Θ in this situation.)

2.2 Nonparametric Likelihood-Ratio Test

This section describes the nonparametric likelihood-ratio test. A good reference for empirical likelihood is the book by Owen (2001) [12].

The parametric LRT described in the previous section requires that we know the distribution family of the data. But often it's not clear what distribution family is applicable. In such a case we risk misspecifying the distribution family. The nonparametric LRT avoids this problem.

The nonparametric LRT we will use is based on the empirical distribution of the data. For this reason it is sometimes called the empirical likelihood-ratio test (ELRT). The empirical LRT has all three advantages of the parametric LRT listed in the previous section. Plus it also has the advantage that we do not have to specify the distribution family of the data.

Owen (2001) [12] p. xiii describes the empirical likelihood method of inference as

follows:

> *Empirical likelihood is a nonparametric method of inference based on a data-driven likelihood ratio function. Like the bootstrap and jackknife, empirical likelihood inference does not require us to specify a family of distributions for the data. Like parametric likelihood methods, empirical likelihood makes an automatic determination of the shape of the confidence regions; it straightforwardly incorporates side information expressed through constraints or prior distributions; it extends to biased sampling and censored data, and it has very favorable asymptotic power properties. Empirical likelihood can be thought of as a bootstrap that does not resample, and as a likelihood without parametric assumptions.*

It is well-known that the bootstrap technique, mentioned by Owen in his quote above, yields a slightly different numerical result each time it is applied to the same data set. By contrast, the empirical likelihood technique yields the identical numerical result each time it is applied to the same data set. This reproducibility is another advantage of the empirical likelihood technique.

Recall that the empirical distribution of a data set $\boldsymbol{x}$ is defined as follows:

$$F_n(t) = \frac{1}{n} \sum_{i=1}^{n} I[X_i \leq t] \ . \tag{2.12}$$

The empirical distribution shown above, which applies to an uncensored and unconstrained data set, is the maximum-likelihood estimator of the distribution F from which the data set derives (see Owen (2001) [12] p. 29). From (2.12) we can see that the probability jump for each data point of the empirical distribution is $\frac{1}{n}$, which is

a very simple result.

On the other hand, if the data are censored, or if the data are under some kind of constraint so that their range is restricted, then the empirical distribution will not in general be the maximum-likelihood estimator of F. This dissertation addresses this situation, where the data are censored and under a constraint.

Chapter 3 derives the mathematical theory for the two-sample case where the data are censored and are constrained by a single hypothesis H_o. It describes how to construct the maximum-likelihood estimators for the two distributions and also how to construct an appropriate likelihood-ratio test for the single hypothesis H_o.

Chapter 4 derives the mathematical theory for the two-sample case where the data are censored and are constrained by a set of p simultaneous hypotheses. $H_{o1}, H_{o2}, \ldots,$ H_{op}. It describes how to construct the maximum-likelihood estimators for the two distributions and also how to construct an appropriate likelihood-ratio test for the p hypotheses.

The mathematical theory presented in Chapters 3 and 4 highlights one disadvantage of the ELRT, that it is computationally intensive. For this reason we have written R-software that will run the necessary computations for both the single-hypothesis case and the multiple-hypothesis case. As mentioned in Chapter 1, the software we have developed can be downloaded free-of-charge on the internet at the Comprehensive R Archive Network (CRAN), `http://cran.r-project.org`, package name emplik2. The R-language itself is also available free-of-charge at the same site.

Chapter 3 Single-Hypothesis Case

3.1 Introduction

In this chapter we derive the likelihood-ratio test for the case of two independent censored samples and a single mean-type hypothesis. The R-code that implements the derivation is listed in Appendix A.

First let us review the concept of censored data. We consider that a random variable X is observed if and only if it lies within the interval $[S_i, R_i]$, where S_i and R_i are random variables. Then we can express the X data by the pair of random variables $(Tx_i, \delta x_i)$ where for i in $1, \ldots, n$,

$$Tx_i = \begin{cases} X_i & \text{if } S_i \le X_i \le R_i \\ R_i & \text{if } R_i < X_i < \infty \\ S_i & \text{if } -\infty < X_i < S_i \end{cases}$$

$$\delta x_i = \begin{cases} 1 & \text{if } S_i \le X_i \le R_i \qquad \text{(uncensored)} \\ 0 & \text{if } R_i < X_i < \infty \qquad \text{(right-censored)} \\ 2 & \text{if } -\infty < X_i < S_i \qquad \text{(left-censored)} \end{cases} .$$

A "doubly-censored" data set is one that contains both right-censored and left-censored values.

We denote the two independent censored samples as $\boldsymbol{Tx} = (Tx_1, \ldots, Tx_n)$ and $\boldsymbol{Ty} = (Ty_1, \ldots, Ty_m)$. They are assumed to have the following properties:

- $\boldsymbol{Tx}$ and $\boldsymbol{Ty}$ are independent.

- $\boldsymbol{Tx}$ and $\boldsymbol{Ty}$ may be uncensored, right-censored, left-censored, or doubly-censored.
- For uncensored data, $Tx_1, \ldots, Tx_n \sim$ iid $F_X(\cdot)$ and $Ty_1, \ldots, Ty_n \sim$ iid $F_Y(\cdot)$.
- $F_X(\cdot)$ and $F_Y(\cdot)$ are unknown.
- $F_X(\cdot)$ and $F_Y(\cdot)$ may be continuous or discrete.

We denote the single mean-type hypothesis as follows:

$$H_o : E\big(\ g(X,Y)\ \big) = \theta \tag{3.1}$$

where g is a function and θ is a constant.

Two illustrative examples of H_o are as follows:

$$\textit{Example 1}: \qquad H_{o1} : E(X - Y) = \theta \tag{3.2}$$

where $g(X,Y) = X - Y$. This is equivalent to $H_{o1} : E(X) - E(Y) = \theta$.

$$\textit{Example 2}: \qquad H_{o2} : P(X > Y) = \theta \tag{3.3}$$

where $g(X,Y) = I[X > Y]$, hence $E\big(\ g(X,Y)\ \big) = P(X > Y)$.

The likelihood-ratio test (LRT) for H_o is derived in the next section.

3.2 Derivation of the LRT

In this section we derive the likelihood-ratio test for H_o.

The derivation below largely follows Zhou (2005)[23]. However, the derivation in Zhou is for a single sample, whereas the calculations here are for two samples. Since the calculations are involved it is convenient to organize them under several subsections.

We first establish the following notation:

- $\boldsymbol{tx} = (tx_1, \ldots, tx_n)$ and $\boldsymbol{ty} = (ty_1, \ldots, ty_m)$ are the distinct values observed.
- "Distinct value" means unique (tx_i, dx_i) or (ty_j, dy_j).
- $\boldsymbol{dx} = (dx_1, \ldots, dx_n)$ and $\boldsymbol{dy} = (dy_1, \ldots, dy_m)$ are the censoring-status values.
- A status is either 0 (right-censored) or 1 (uncensored) or 2 (left-censored).
- $\boldsymbol{wx} = (wx_1, \ldots, wx_n)$ and $\boldsymbol{wy} = (wy_1, \ldots, wy_m)$ are the weights.
- The weight is the number of occurrences of a distinct value.
- Later on (in the EM algorithm) we allow the weights to be fractions.
- A "hat" $(\wedge)$ over a variable indicates a maximum-likelihood estimation, e.g. $\widehat{\boldsymbol{\mu}}$.
- $\boldsymbol{\mu} = (\mu_1, \ldots, \mu_n)$ are the probability jumps for $\boldsymbol{x}$.
- $\boldsymbol{\nu} = (\nu_1, \ldots, \nu_m)$ are the probability jumps for $\boldsymbol{y}$.
- The probability jumps are non-negative.
- The probability jumps sum to 1, for both $\boldsymbol{\mu}$ and $\boldsymbol{\nu}$.
- *log* is the natural logarithm.

3.2.1 Calculation of the Unconstrained Log-Likelihood

Let Θ_μ denote the space for $\boldsymbol{\mu}$ and let $\ell_1(\widehat{\boldsymbol{\mu}}|\boldsymbol{tx})$ denote the supremum of the unconstrained log-likelihood of $\boldsymbol{\mu}|\boldsymbol{tx}$. Then by the definition of likelihood,

$$\ell_1(\widehat{\boldsymbol{\mu}}|\boldsymbol{x}) = \sup_{\boldsymbol{\mu}\in\Theta_{\boldsymbol{\mu}}} \log\Big(\prod_{dx_i=1} (\mu_i)^{wx_i} \prod_{dx_i=0} (\sum_{Tx_k>Tx_i} \mu_k)^{wx_i} \prod_{dx_i=2} (\sum_{Tx_k<Tx_i} \mu_k)^{wx_i}\Big) \tag{3.4}$$

$$\begin{aligned} = \sup_{\boldsymbol{\mu}\in\Theta_{\boldsymbol{\mu}}} \Big(&\sum_{dx_i=1} wx_i \ \log(\mu_i) + \sum_{dx_i=0} wx_i \ \log(\sum_{Tx_k>Tx_i} \mu_k) \\ &+ \sum_{dx_i=2} wx_i \ \log(\sum_{Tx_k<Tx_i} \mu_k) \Big) . \end{aligned} \tag{3.5}$$

In the RHS of equation (3.4)

- The first product involves the probability jumps of the uncensored data.
- The second product involves the survivals of the right-censored data.
- The third product involves the "left-survivals" of the left-censored data.

Calculation of the distribution $\widehat{\boldsymbol{\mu}}$ that maximizes (3.5) depends on the status of the data.

- If the data are uncensored, $\widehat{\boldsymbol{\mu}}$ is the empirical distribution.
- If the data are right-censored, $\widehat{\boldsymbol{\mu}}$ is the Kaplan-Meier estimator as described in Kaplan and Meier (1958) [8].
- If the data are left-censored, $\widehat{\boldsymbol{\mu}}$ is the "left-Kaplan-Meier-estimator" as described in Gomez et. al. (1992) [5].

- If the data are doubly-censored, $\widehat{\boldsymbol{\mu}}$ is found via a numerical procedure as described in Chang and Yang (1987) [2].

- In any case, non-zero jumps will occur only at uncensored points, as noted in Zhou (2005) [23].

Similarly let $\Theta_{\boldsymbol{\nu}}$ represent the space for $\boldsymbol{\nu}$ and let $\ell_1(\widehat{\boldsymbol{\nu}}|\boldsymbol{ty})$ denote the supremum of the unconstrained log-likelihood of $\boldsymbol{\nu}|\boldsymbol{ty}$. The expression for $\ell_1(\widehat{\boldsymbol{\nu}}|\boldsymbol{ty})$ is analogous to (3.5):

$$\ell_1(\widehat{\boldsymbol{\nu}}|\boldsymbol{y}) = \sup_{\boldsymbol{\nu}\in\Theta_{\boldsymbol{\nu}}} \Big(\sum_{dy_j=1} wy_j \ \log(\nu_j) + \sum_{dy_j=0} wy_j \ \log(\sum_{Ty_k>Ty_j} \nu_k) + \sum_{dy_j=2} wy_j \ \log(\sum_{Ty_k<Ty_j} \nu_k) \Big) \ . \tag{3.6}$$

We can calculate the maximum-likelihood estimator $\widehat{\boldsymbol{\nu}}$ for (3.6) using a procedure analogous to that used for $\widehat{\boldsymbol{\mu}}$. Then since Tx and Ty are independent we can write the unconstrained log-likelihood ℓ_1 as

$$\ell_1 = \ell_1(\widehat{\boldsymbol{\mu}}|\boldsymbol{tx}) + \ell_1(\widehat{\boldsymbol{\nu}}|\boldsymbol{ty}) \ . \tag{3.7}$$

3.2.2 Calculation of the Constrained Log-Likelihood

The constrained log-likelihood ℓ_o is found by solving a system of equations for $\widehat{\boldsymbol{\mu}}$ and $\widehat{\boldsymbol{\nu}}$. A direct solution of these equations is not available, hence numerical methods must be applied.

Peto (1973)[15] shows how the constrained log-likelihood for interval-censored data can be maximized using a constrained Newton-Raphson algorithm. However the Newton-Raphson algorithm can become unwieldy with large data sets. We therefore prefer to use an expectation-maximization (EM) algorithm as described in Turnbull

(1976)[18].

It is convenient to organize the description of the calculation of $\widehat{\boldsymbol{\mu}}$ and $\widehat{\boldsymbol{\nu}}$ via the EM algorithm under four headings as follows:

1. Equations for $\widehat{\boldsymbol{\mu}}$ and $\widehat{\boldsymbol{\nu}}$.
2. Expectation step of the EM algorithm.
3. Maximization step of the EM algorithm.
4. Evaluation of H_o

3.2.2.1 *Equations for $\widehat{\mu}$ and $\widehat{\nu}$*

First we rewrite (3.1) as

$$H_o : E\big(\ g(X,Y) - \theta\ \big) = 0\ . \tag{3.8}$$

Then we define an estimator $\widehat{E}(g(X,Y)-\theta)$ for the left-hand side (LHS) of (3.8) as follows:

$$\widehat{E}\big(\ g(X,Y) - \theta\ \big) = \sum_{dx_i=1} \sum_{dy_j=1} \big(\ g(tx_i, ty_j) - \theta\ \big)\, \widehat{\mu}_i\, \widehat{\nu}_j\ . \tag{3.9}$$

We restrict (3.9) and the subsequent equations to the uncensored data ($dx_i = 1, dy_j = 1$) with additional "shifted" weight, in accordance with the shifting procedure that is explained below. Then the restrictions on $\widehat{\boldsymbol{\mu}}$ and $\widehat{\boldsymbol{\nu}}$ can be rendered as follows:

$$\sum_{dx_i=1} \widehat{\mu}_i - 1 = 0 \tag{3.10}$$

$$\sum_{dy_j=1} \widehat{\nu}_j - 1 = 0 \tag{3.11}$$

$$\sum_{dx_i=1} \sum_{dy_j=1} \left(g(tx_i, ty_j) - \theta \right) \widehat{\mu}_i \, \widehat{\nu}_j = 0 \quad \text{from (3.8) and (3.9).} \tag{3.12}$$

Of course there is also the restriction that the entries in $\widehat{\boldsymbol{\mu}}$ and $\widehat{\boldsymbol{\nu}}$ must be non-negative. Chapter 7 contains a proof showing that these entries will indeed be non-negative if θ lies in the "feasible" range.

Let us define Θ_o as the set of $(\widehat{\boldsymbol{\mu}}, \widehat{\boldsymbol{\nu}})$ that are consistent with the constraint equations (3.10) - (3.12). We will use Lagrange multipliers γ, η, and λ to find $(\widehat{\boldsymbol{\mu}}, \widehat{\boldsymbol{\nu}})$ $\in \Theta_o$ that maximize the likelihood as in (3.5) and (3.6). Then ℓ_o can be expressed as follows:

$$\begin{aligned} \ell_o = \sup_{(\boldsymbol{\mu},\boldsymbol{\nu}) \in \Theta_o} \Big(& \sum_{dx_i=1} wx_i \, \log(\mu_i) + \sum_{dy_j=1} wy_j \, \log(\nu_j) - \gamma(1 - \sum_{dx_i=1} \mu_i) \\ & -\eta(1 - \sum_{dy_j=1} \nu_j) - \lambda \sum_{dx_i=1} \sum_{dy_j=1} \left(g(tx_i, ty_j) - \theta \right) \mu_i \, \nu_j \Big) . \end{aligned} \tag{3.13}$$

We maximize the RHS of (3.13) by taking partial derivatives with respect to $\mu_i, \nu_j, \gamma, \eta, \lambda$ and setting the partial derivatives equal to 0. This results in the following system of equations that must be solved for $\widehat{\mu}_i$ and $\widehat{\nu}_j$:

$$\sum_{dx_i=1} \widehat{\mu}_i - 1 = 0 \tag{3.14}$$

$$\sum_{dy_j=1} \widehat{\nu}_j - 1 = 0 \tag{3.15}$$

$$\widehat{\mu}_i = \frac{wx_i}{\sum_{dx_i=1} wx_i + \lambda \sum_{dy_j=1} \left(g(tx_i, ty_j) - \theta \right) \widehat{\nu}_j} \tag{3.16}$$

$$\widehat{\nu_j} = \frac{wy_j}{\sum\limits_{dy_j=1} wy_j + \lambda \sum\limits_{dx_i=1} \left(\ g(tx_i, ty_j) - \theta \ \right) \ \widehat{\mu_i}} \tag{3.17}$$

$$\sum_{dx_i=1} \sum_{dy_j=1} \left(\ g(tx_i, ty_j) - \theta \ \right) \ \widehat{\mu_i} \ \widehat{\nu_j} = 0 \ . \tag{3.18}$$

Taking a second partial derivative in the RHS of (3.13) with respect to μ_i yields a negative number, confirming that $\widehat{\mu}_i$ is indeed a maximizer (rather than a minimizer) of (3.13). Likewise taking a second partial derivative in the RHS of (3.13) with respect to ν_j yields a negative number, confirming that $\widehat{\nu}_j$ is also a maximizer (rather than a minimizer) of (3.13).

Expressions for $\widehat{\mu_i}$ and $\widehat{\nu_j}$ can be obtained by solving (3.16) - (3.18) simultaneously. A direct solution is not available. However, the equations can be solved numerically using the Expectation Maximization (EM) algorithm of Dempster, Laird, and Rubin (1977) [3].

As mentioned previously, application of the EM algorithm to this type problem is described in Turnbull (1976) [18]. Turnbull applies his algorithm to only a single sample. This dissertation extends the work of Turnbull by applying the EM algorithm to two-samples.

The EM algorithm is an iterative procedure. Each iteration of the EM algorithm runs as follows:

1. Establish initial estimates $\widehat{\boldsymbol{\mu}}^{(0)}, \widehat{\boldsymbol{\nu}}^{(0)}$ for $\widehat{\boldsymbol{\mu}}, \widehat{\boldsymbol{\nu}}$, respectively.

2. In the expectation step of the EM algorithm, shift the weights of the censored data onto the uncensored data. This shifting converts the weights $\boldsymbol{wx}$ and $\boldsymbol{wy}$ to their expected values, which we denote as $\boldsymbol{Wx}$ and $\boldsymbol{Wy}$, respectively.

3. In the maximization step of the EM algorithm, convert $\widehat{\boldsymbol{\mu}}^{(0)}$ and $\widehat{\boldsymbol{\nu}}^{(0)}$ into improved estimates, which we denote as $\widehat{\boldsymbol{\mu}}^{(1)}$ and $\widehat{\boldsymbol{\nu}}^{(1)}$, respectively.

4. Set $\widehat{\boldsymbol{\mu}}^{(0)} \leftarrow \widehat{\boldsymbol{\mu}}^{(1)}$ and $\widehat{\boldsymbol{\nu}}^{(0)} \leftarrow \widehat{\boldsymbol{\nu}}^{(1)}$ to complete the iteration.

3.2.2.2 *Expectation Step of the EM Algorithm*

As mentioned just above, the expectation step of the EM algorithm involves calculating the expectation of the "latent" (i.e., "present but of unknown value") variables $\boldsymbol{wx}$ and $\boldsymbol{wy}$. We denote these expectations as $\boldsymbol{Wx}$ and $\boldsymbol{Wy}$, respectively.

In calculating these expectations we reckon that a probability jump for a censored event cannot occur at its censoring value (since by definition the event occurs beyond the censoring value). Therefore per Zhou (2005) [23] the expectation step will involve dividing the weights of the censored data among the uncensored data that lie beyond them. The division will be done in proportion to the probability jumps of the uncensored data. We call this procedure "shifting." It is also known as "redistribution."

As a simple example of the shifting procedure, suppose we have the following situation for the $\boldsymbol{tx}$ data:

- The data (all unique, in ascending order) are $tx_1, \ldots, tx_{10}$.
- The corresponding weights are $wx_1, wx_2, \ldots, wx_{10}$.
- The corresponding probability jumps, in the vector $\widehat{\boldsymbol{\mu}}_{\mathbf{0}}$, are $\widehat{\mu}_1, \ldots, \widehat{\mu}_{10}$.

- The data are uncensored except for tx_8 which is right-censored.

Then we shift the weight of tx_8 onto tx_9 and tx_{10} as follows:

- Initialize $Wx_i = wx_i$ for $i = 1, \ldots, 10$.
- Increase Wx_9 by $(wx_8)(\widehat{\mu}_9)/(\widehat{\mu}_9 + \widehat{\mu}_{10})$.
- Increase Wx_{10} by $(wx_8)(\widehat{\mu}_{10})/(\widehat{\mu}_9 + \widehat{\mu}_{10})$.
- Set $Wx_8 = 0$.

An intuitive explanation of the above shifting procedure is as follows. We know the 8th observation must occur somewhere above its censoring value tx_8. The only practical choices for its expected occurrence are at tx_9 and tx_{10}, since we know events occurred at those two values. So we "shift" the weight of tx_8 to the right, splitting its weight between tx_9 and tx_{10} in proportion to their estimated probability jumps.

We express this shifting procedure in formal notation as follows:

$$Wx_i = \sum_{k=1}^{n} E_F\big(I_{[Tx_k = tx_i]} \mid tx_k, \delta x_k \big) \tag{3.19}$$

$$Wy_j = \sum_{k=1}^{m} E_F\big(I_{[Ty_k = ty_j]} \mid ty_k, \delta y_k \big) \ . \tag{3.20}$$

Equation (3.19) can be stated in words as follows: the expected weight at a location tx_i, given tx_k and δx_k, is its own weight plus all the shifted weight that it receives from the censored observations. Equation (3.20) can be stated in a similar manner.

The expectation in the RHS of (3.19) depends on the status of the data, as follows:

- For right-censored tx_k:

$$E_F\left(\; I_{[Tx_k=tx_i]} \mid tx_k, \delta x_k \; \right) = \frac{\Delta F(tx_i)}{1 - F(tx_k)} \qquad \text{for } tx_i > tx_k \tag{3.21}$$

$$E_F\left(\; I_{[Tx_k=tx_i]} \mid tx_k, \delta x_k \; \right) = 0 \qquad \text{for } tx_i \leq tx_k \tag{3.22}$$

- For left-censored tx_k:

$$E_F\left(\; I_{[Tx_k=tx_i]} \mid tx_k, \delta x_k \; \right) = \frac{\Delta F(tx_i)}{F(tx_k^-)} \qquad \text{for } tx_i < tx_k \tag{3.23}$$

$$E_F\left(\; I_{[Tx_k=tx_i]} \mid tx_k, \delta x_k \; \right) = 0 \qquad \text{for } tx_i \geq tx_k \tag{3.24}$$

- For uncensored tx_k:

$$E_F\left(\; I_{[Tx_k=tx_i]} \mid tx_k, \delta x_k \; \right) = 1 \qquad \text{for } tx_i = tx_k \tag{3.25}$$

$$E_F\left(\; I_{[Tx_k=tx_i]} \mid tx_k, \delta x_k \; \right) = 0 \qquad \text{for } tx_i \neq tx_k \tag{3.26}$$

The expectation in the RHS of (3.20) is similarly calculated, using equations analogous to (3.21) - (3.26).

As a result of the expectation step (3.16) - (3.18) are transformed into the following three equations:

$$\widehat{\mu}_i^{(1)} = \frac{W x_i}{\sum\limits_{dx_i=1} w x_i + \lambda \sum\limits_{dy_j=1} \left(\; g(tx_i, ty_j) - \theta \; \right) \widehat{\nu}_j^{(0)}} \tag{3.27}$$

$$\widehat{\nu}_j^{(1)} = \frac{W y_j}{\sum\limits_{dy_j=1} w y_j + \lambda \sum\limits_{dx_i=1} \left(\; g(tx_i, ty_j) - \theta \; \right) \widehat{\mu}_i^{(0)}} \tag{3.28}$$

$$\sum_{dx_i=1} \sum_{dy_j=1} \left(g(tx_i, ty_j) - \theta \right) \widehat{\mu}_i^{(1)} \widehat{\nu}_j^{(1)} = 0 \tag{3.29}$$

Note the following in (3.27) - (3.29) just above:

1. Positive probability jumps for $\widehat{\boldsymbol{\mu}}$ and $\widehat{\boldsymbol{\nu}}$ occur only at the uncensored data points.
2. $\widehat{\mu}_i^{(0)}$ and $\widehat{\mu}_i^{(1)}$ are not the same value. Rather $\widehat{\mu}_i^{(0)}$ is an initial estimate and $\widehat{\mu}_i^{(1)}$ is an improved estimate.
3. $\widehat{\nu}_j^{(0)}$ and $\widehat{\nu}_j^{(1)}$ are not the same value. Rather $\widehat{\nu}_j^{(0)}$ is an initial estimate and $\widehat{\nu}_j^{(1)}$ is an improved estimate.

3.2.2.3 *Maximization Step of the EM Algorithm*

The maximization step is an iterative procedure that accomplishes three things with reference to (3.27) - (3.29):

1. $\widehat{\mu}_i^{(0)}$ and $\widehat{\mu}_i^{(1)}$ become equal within some small tolerance.
2. $\widehat{\nu}_j^{(0)}$ and $\widehat{\nu}_j^{(1)}$ become equal within some small tolerance.
3. The LHS of (3.29) becomes equal to zero within some small tolerance.

Each iteration of the maximization runs as follows:

1. In (3.29) substitute the RHS of (3.27) for $\widehat{\boldsymbol{\mu}}^{(1)}$ and the RHS of (3.28) for $\widehat{\boldsymbol{\nu}}^{(1)}$, forming a single equation in λ.
2. Solve the substituted (3.29) for λ, using Rgui "uniroot" function.
3. Use λ to calculate $\widehat{\mu}^{(1)}$ and $\widehat{\nu}^{(1)}$ in (3.27) and (3.28).

4. Update $\widehat{\boldsymbol{\mu}}^{(0)} \leftarrow \widehat{\boldsymbol{\mu}}^{(1)}$ and $\widehat{\boldsymbol{\nu}}^{(0)} \leftarrow \widehat{\boldsymbol{\nu}}^{(1)}$.

5. Calculate the log-likelihood ℓ_0 as in RHS (3.13) using $\widehat{\boldsymbol{\mu}}^{(1)}$ and $\widehat{\boldsymbol{\nu}}^{(1)}$.

6. If ℓ_0 has not stabilized, return to the expectation step of the EM algorithm.

7. Otherwise, we are done, report $\widehat{\mu}^{(1)}, \widehat{\nu}^{(1)}$, ℓ_o.

3.2.3 Evaluation of Ho

Once we have found ℓ_1 and ℓ_0 we can calculate the log-likelihood ratio as,

$$LLR = \ell_0 - \ell_1. \tag{3.30}$$

Using LLR we can then calculate an approximate p-value for H_o, assuming that $-2LLR$ is asymptotically distributed as $\chi^2_{(1)}$ when H_o is true. The p-value is calculated as $1 - F(-2LLR)$ where F is a $\chi^2_{(1)}$ distribution. We reject H_o with 95% confidence if this p-value is less than 0.05. Otherwise we fail to reject H_o with 95% confidence.

It is also possible to calculate an approximate confidence interval for the parameter θ associated with H_o as in (3.1). For example, suppose the hypothesis is $H_o : P(X \geq Y) = 0.5$. By trial-and-error we can find a number $\alpha < 0.5$ such that $H_o : P(X \geq Y) = \alpha$ has a p-value of 0.05. Similarly we can find a number $\beta > 0.5$ such that $H_o : P(X \geq Y) = \beta$ has a p-value of 0.05. Then an approximate 95 % confidence interval for the estimate $\theta = 0.5$ is $[\alpha, \beta]$. An example of this procedure is given in section 5.1

As noted above, the calculation of the approximate p-value for H_o and the approximate confidence limits for θ are based on the assumption that $-2LLR$ is asymptotically distributed as $\chi^2_{(1)}$ when H_o is true. We end this section with an argument that this assumption is reasonable.

Wilks (1938) [20] demonstrated that the uncensored-single-sample likelihood-ratio-statistic for a composite hypothesis is asymptotically distributed as $\chi^2_{(1)}$. Pan and Zhou (1999) [13] demonstrated the same result for single-sample right-censored data with a mean-type constraint. Murphy and Van der Vaart (1997) [11] demonstrated the same result for single-sample doubly-censored data with a mean-type constraint.

Owen (2001) [12] pp. 223-227 demonstrated that the uncensored two-sample likelihood-ratio-statistic for a mean-type constraint is asymptotically distributed as $\chi^2_{(1)}$.

In light of this previous work one might reasonably expect that $-2LLR$ should similarly be asymptotically distributed as $\chi^2_{(1)}$ in the two-sample censored-data case that we are considering in this dissertation. This reasonable expectation is reinforced by the simulation work shown in the four figures 3.1 to 3.4 below.

The four figures 3.1 to 3.4 below show probability plots of $-2LLR$ versus chisq(1) quantiles for some two-sample simulated data. Each of the two samples contains 50 values. The two samples are both generated from the same distribution and the hypothesis is $H_o : P(X \geq Y) = 0.5$ as in (3.3). In this case Ho is true since the two samples are from the same distribution. We see that the plots all fall along the 45° solid line up to a quantile of about 6 (which corresponds to a percentile of 98.6). Therefore the four plots support the idea that $-2LLR$ asymptotically follows a $\chi^2_{(1)}$

distribution for two-sample uncensored, right-censored, left-censored, and doubly-censored data with a mean-type constraint.

Figure 3.5 is a plot of two-sample right-censored data for which the hypothesis $H_o : E(X - Y) = 0$ is not true, rather in fact $E(X - Y) = -155.4$. Each of the two samples contains 50 values. It is evident that the plot does not lie on the 45° solid line. This illustrates that $-2LLR$ is not expected to be distributed as $\chi^2_{(1)}$ when H_o is not true.

Figure 3.6 is a plot of the same data as Figure 3.5 but plotted against chisq(df=1, ncp=0.85) quantiles (a non-central $\chi^2_{(1)}$ distribution). The ncp of 0.85 was found by trial and error. The plot fits the 45° solid line reasonably well up to a quantile of about 7.5, which corresponds to a percentile of 96.5. This suggests that $-2LLR$ under the alternate hypothesis asymptotically follows a non-central $\chi^2_{(1)}$ distribution. Such an asymptotic non-central $\chi^2_{(1)}$ distribution for a univariate empirical likelihood test is described in Owen [12] p. 16.

The code used to generate Figures 3.1 to 3.6 is listed in Appendix B.

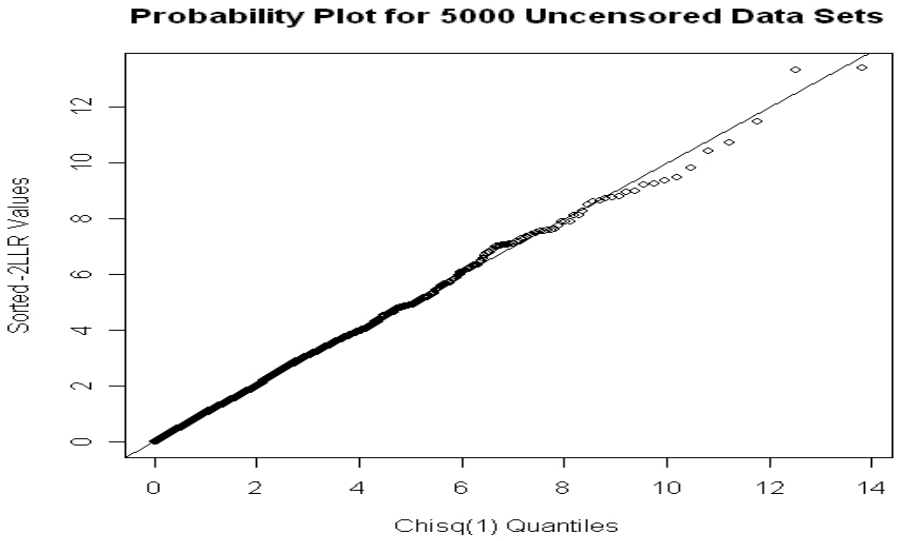

Figure 3.1: Probability Plot for Uncensored Data, Two Samples, Each 50 Values.

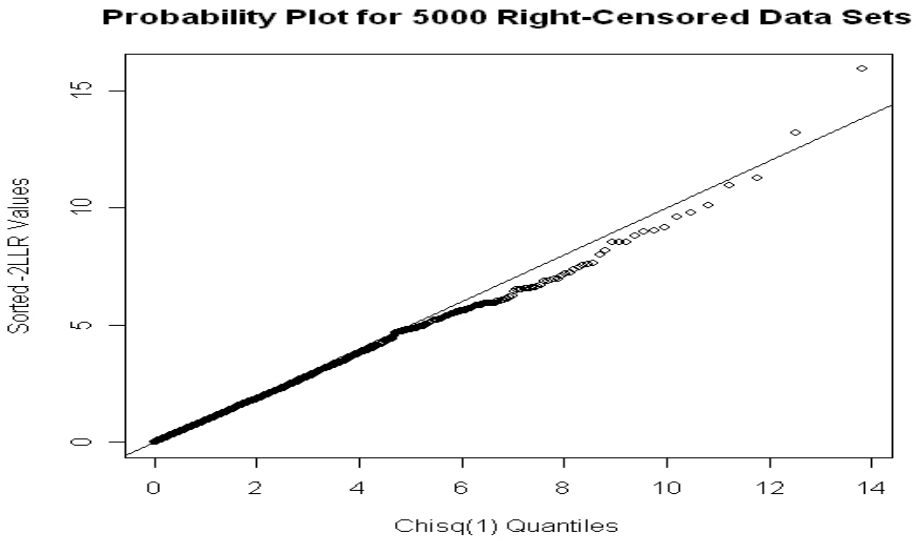

Figure 3.2: Probability Plot for Right-Censored Data, Two Samples, Each 50 Values.

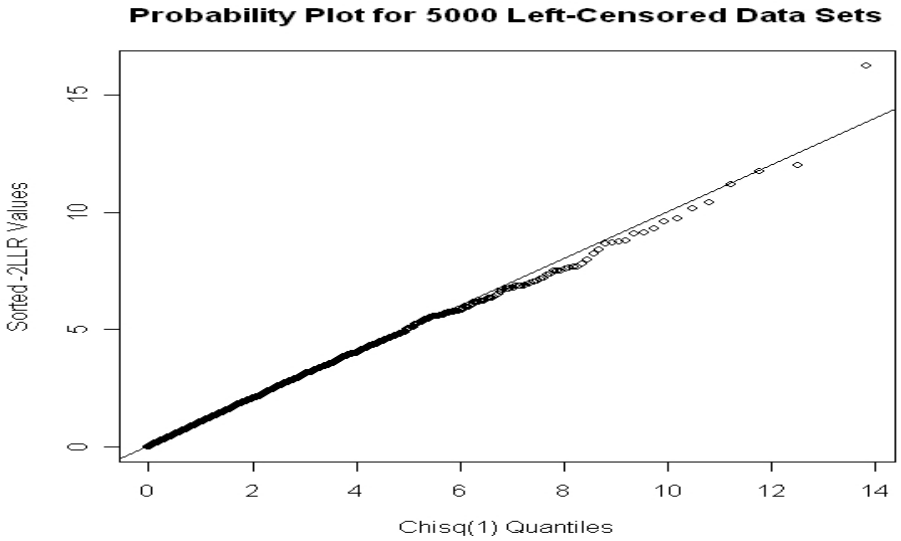

Figure 3.3: Probability Plot for Left-Censored Data, Two Samples, Each 50 Values.

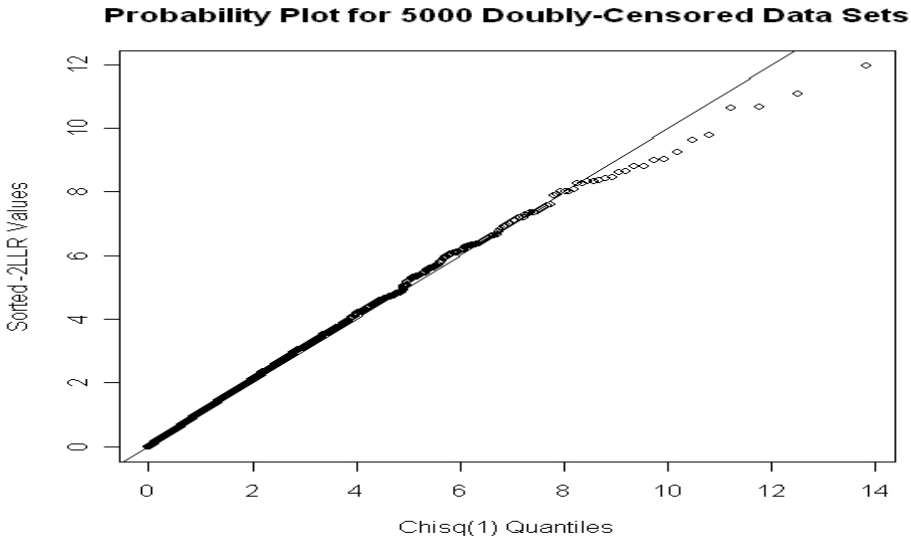

Figure 3.4: Probability Plot for Doubly-Censored Data, Two Samples, Each 50 Values.

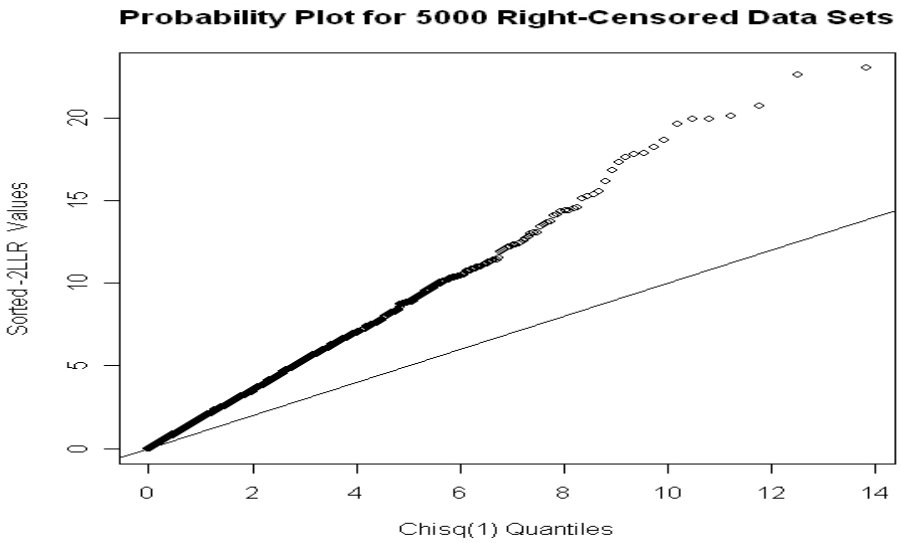

Figure 3.5: Probability Plot for Right-Censored Data, Two Samples, Each 50 Values, H_o False.

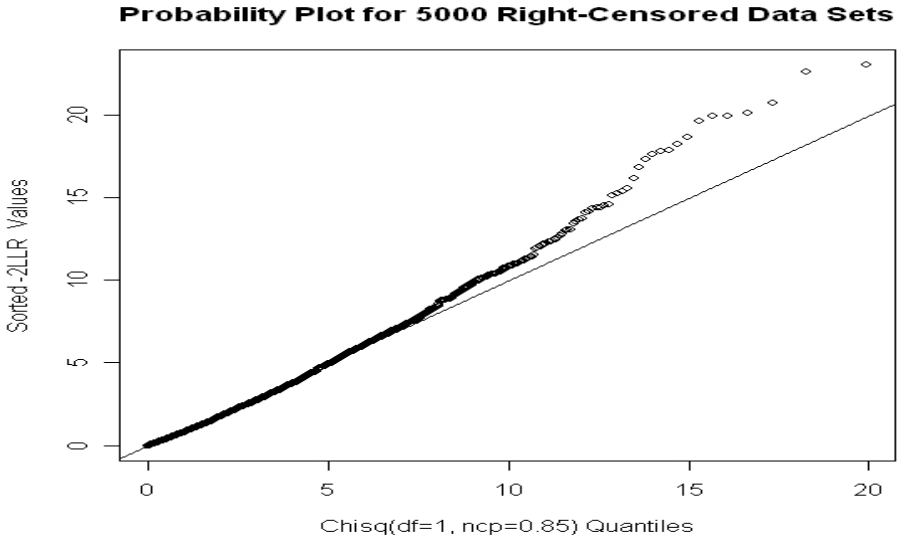

Figure 3.6: Probability Plot for Right-Censored Data, Two Samples, Each 50 Values, H_o False, Using Chisq(df=1, ncp=0.85) Quantiles.

Chapter 4 Multiple-Hypotheses Case

4.1 Introduction

In this chapter we derive the likelihood-ratio test for the case of two independent censored samples and p mean-type hypotheses. The R-code that implements the derivation is listed in Appendix A.

We denote the two independent censored samples as $\boldsymbol{Tx} = (Tx_1, \ldots, Tx_n)$ and $\boldsymbol{Ty} = (Ty_1, \ldots, Ty_m)$. They are assumed to have the following properties:

- $\boldsymbol{Tx}$ and $\boldsymbol{Ty}$ are independent.
- $\boldsymbol{Tx}$ and $\boldsymbol{Ty}$ may be uncensored, right-censored, left-censored, or doubly-censored.
- For uncensored data, $Tx_1, \ldots, Tx_n \sim$ iid $F_X(\cdot)$ and $Ty_1, \ldots, Ty_n \sim$ iid $F_Y(\cdot)$.
- $F_X(\cdot)$ and $F_Y(\cdot)$ are unknown.
- $F_X(\cdot)$ and $F_Y(\cdot)$ may be continuous or discrete.

We denote the p mean-type hypotheses as follows:

$$H_o : E\big(\ g_1(X,Y)\ \big) = \theta_1 \tag{4.1}$$

$$H_1 : E\big(\ g_2(X,Y)\ \big) = \theta_2 \tag{4.2}$$

$$\vdots$$

$$H_p : E\big(\ g_p(X,Y)\ \big) = \theta_p \tag{4.3}$$

where $g_1(X,Y),\ g_2(X,Y),\ldots,\ g_p(X,Y)$ are functions and $\theta_1,\ \theta_2,\ldots,$ θ_p are constants.

We can express the simultaneous equalities in (4.1) - (4.3) using vectors as follows:

$$E\big(\ \boldsymbol{g}(X,Y)\ \big) = \boldsymbol{\theta} \tag{4.4}$$

where

$$\boldsymbol{g}(X,Y) = \big(\ g_1(X,Y),\ g_2(X,Y),\ \ldots,\ g_p(X,Y)\ \big)^T \tag{4.5}$$

and

$$\boldsymbol{\theta} = (\theta_1,\ldots,\theta_p)^T \tag{4.6}$$

The likelihood-ratio test (LRT) for the simultaneous hypotheses in (4.1) - (4.3) is derived in the next section.

4.2 Derivation of the LRT

4.2.1 Calculation of the Unconstrained Log-Likelihood

The calculation of the unconstrained log-likelihood ℓ_1 is the same as the calculation of ℓ_1 in the single-hypothesis case (section 3.2.1) since ℓ_1 is not affected by the hypotheses.

4.2.2 Calculation of the Constrained Log-Likelihood

The calculation of the constrained log-likelihood is similar to the calculation for the single-hypothesis case in section 3.2.2, although it is somewhat more involved. As

in section 3.2.2 we use the EM algorithm to find the constrained-maximum-likelihood estimators of the probability jumps, $\widehat{\boldsymbol{\mu}} = (\widehat{\mu_1}, \ldots, \widehat{\mu_p})^T$ and $\widehat{\boldsymbol{\nu}} = (\widehat{\nu_1}, \ldots, \widehat{\nu_p})^T$.

It is convenient to organize the description of the calculation of $\widehat{\boldsymbol{\mu}}$ and $\widehat{\boldsymbol{\nu}}$ via the EM algorithm under four headings as follows:

1. Equations for $\widehat{\boldsymbol{\mu}}$ and $\widehat{\boldsymbol{\nu}}$.
2. Expectation step of the EM algorithm.
3. Maximization step of the EM algorithm.
4. Evaluation of H_o

4.2.2.1 *Equations for $\widehat{\boldsymbol{\mu}}$ and $\widehat{\boldsymbol{\nu}}$*

First we rewrite (4.4) as

$$E\big(\ \boldsymbol{g}(X,Y)\ -\boldsymbol{\theta}\big) = \mathbf{0} \tag{4.7}$$

where $\mathbf{0}$ is a vertical vector of zeros of length p.

Then we define an estimator $\widehat{E}\big(\ \boldsymbol{g}(X,Y) - \boldsymbol{\theta}\ \big)$ for the LHS of (4.7) as follows:

$$\widehat{E}\big(\ \boldsymbol{g}(X,Y)\ -\boldsymbol{\theta}\ \big) = \big(\ \widehat{\boldsymbol{\mu}}^T H_1\ \widehat{\boldsymbol{\nu}},\ \ldots,\ \widehat{\boldsymbol{\mu}}^T H_p\ \widehat{\boldsymbol{\nu}}\ \big)^T \tag{4.8}$$

where

$$H_k = [\ g_k(tx_i, ty_j) - mu_k\ ],\quad \text{for } dx_i = 1,\ dy_j = 1,\ k = 1, \ldots, p \tag{4.9}$$

and

$$\widehat{\boldsymbol{\mu}}^T H_k\ \widehat{\boldsymbol{\nu}} = \sum_{dx_i=1} \sum_{dy_j=1} \big(\ g_k(tx_i, ty_j) - \theta_k\ \big)\ \widehat{\mu}_i\ \widehat{\nu}_j \tag{4.10}$$

We restrict (4.8) - (4.10) and the subsequent equations to the uncensored data ($dx_i = 1, dy_j = 1$) with additional "shifted" weight, in accordance with the shifting procedure that is explained below.

Then the constraints on $\widehat{\boldsymbol{\mu}}$ and $\widehat{\boldsymbol{\nu}}$ can be rendered as follows:

$$\sum_{dx_i=1} \widehat{\mu}_i - 1 = 0 \tag{4.11}$$

$$\sum_{dy_j=1} \widehat{\nu}_j - 1 = 0 \tag{4.12}$$

$$\left(\widehat{\boldsymbol{\mu}}^T H_1 \, \widehat{\boldsymbol{\nu}}, \ \ldots, \ \widehat{\boldsymbol{\mu}}^T H_p \, \widehat{\boldsymbol{\nu}} \right)^T = \mathbf{0} \tag{4.13}$$

where (4.13) stems from (4.7) and (4.8).

Of course there is also the restriction that the entries in $\widehat{\boldsymbol{\mu}}$ and $\widehat{\boldsymbol{\nu}}$ must be non-negative. Chapter 7 contains a proof that these entries will indeed be non-negative if the entries in $\boldsymbol{\theta}$ all lie in their "feasible" ranges.

Let us define Θ_o as the set of $(\widehat{\boldsymbol{\mu}}, \widehat{\boldsymbol{\nu}})$ that are consistent with equations (4.11) - (4.13). Then similar to (3.13) we can express ℓ_o using Lagrange multipliers γ, η, and $\boldsymbol{\lambda}$ as follows:

$$\begin{aligned} \ell_o = \sup_{(\boldsymbol{\mu},\boldsymbol{\nu}) \in \Theta_o} \Big(& \sum_{dx_i=1} wx_i \ \log(\mu_i) + \sum_{dy_j=1} wy_j \ \log(\nu_j) \\ & -\gamma(1 - \sum_{dx_i=1} \mu_i) - \eta(1 - \sum_{dy_j=1} \nu_j) - \left(\boldsymbol{\mu}^T H_1 \, \boldsymbol{\nu}, \ \ldots \ , \boldsymbol{\mu}^T H_p \, \boldsymbol{\nu} \right) \boldsymbol{\lambda} \Big) \end{aligned} \tag{4.14}$$

where

$$\boldsymbol{\lambda} = (\lambda_1, \lambda_2, \ldots, \lambda_p)^T \tag{4.15}$$

and

$$\begin{aligned}(\ \widehat{\boldsymbol{\mu}}^T H_1 \ \widehat{\boldsymbol{\nu}}, \ \ldots \ , \widehat{\boldsymbol{\mu}}^T H_p \ \widehat{\boldsymbol{\nu}}) \ \boldsymbol{\lambda} \ &= \ \lambda_1 \sum_{dx_i=1} \sum_{dy_j=1} \left(\ g_1(tx_i, ty_j) - \theta_1 \ \right) \ \mu_i \ \nu_j \\ &+ \ \ldots + \ \lambda_p \sum_{dx_i=1} \sum_{dy_j=1} \left(\ g_p(tx_i, ty_j) - \theta_p \ \right) \ \mu_i \ \nu_j \end{aligned} \tag{4.16}$$

We maximize the RHS of (4.14) by taking partial derivatives with respect to $\mu_i, \nu_j, \gamma, \eta, \lambda$ and setting the partial derivatives equal to 0. This results in the following system of equations that must be solved for $\widehat{\boldsymbol{\mu}}$ and $\widehat{\boldsymbol{\nu}}$:

$$\sum_{dx_i=1} \widehat{\mu}_i - 1 = 0 \tag{4.17}$$

$$\sum_{dy_j=1} \widehat{\nu}_j - 1 = 0 \tag{4.18}$$

$$\widehat{\mu}_i = \frac{wx_i}{\sum_{dx_i=1} wx_i + \boldsymbol{\lambda}^T \sum_{dy_j=1} \left(\ \boldsymbol{g}(tx_i, ty_j) - \boldsymbol{\theta} \ \right) \ \widehat{\nu}_j} \tag{4.19}$$

$$\widehat{\nu}_j = \frac{wy_j}{\sum_{dy_j=1} wy_j + \boldsymbol{\lambda}^T \sum_{dx_i=1} \left(\ \boldsymbol{g}(tx_i, ty_j) - \boldsymbol{\theta} \ \right) \ \widehat{\mu}_i} \tag{4.20}$$

$$\left(\ \widehat{\boldsymbol{\mu}}^T H_1 \ \widehat{\boldsymbol{\nu}}, \ \ldots, \ \widehat{\boldsymbol{\mu}}^T H_p \ \widehat{\boldsymbol{\nu}} \ \right)^T = \mathbf{0} \tag{4.21}$$

where $\widehat{\boldsymbol{\mu}}$ and $\widehat{\boldsymbol{\nu}}$ are vertical vectors of the maximum-likelihood probability jumps.

Taking a second partial derivative in the RHS of (4.14) with respect to μ_i yields a negative number, confirming that $\widehat{\mu}_i$ is indeed a maximizer (rather than a minimizer) of (4.14). Likewise taking a second partial derivative in the RHS of (4.14) with respect to ν_j yields a negative number, confirming that $\widehat{\nu}_j$ is likewise a maximizer (rather than a minimizer) of (4.14).

Expressions for $\widehat{\mu}_i$ and $\widehat{\nu}_j$ can be obtained by solving (4.19) - (4.21) simultaneously. A direct solution is not available. However, the equations can be solved numerically using the Expectation Maximization (EM) algorithm of Dempster, Laird, and Rubin (1977) [3].

The EM algorithm is an iterative procedure. Each iteration of the EM algorithm runs as follows:

1. Establish initial estimates $\widehat{\boldsymbol{\mu}}^{(0)}, \widehat{\boldsymbol{\nu}}^{(0)}$ for $\widehat{\boldsymbol{\mu}}, \widehat{\boldsymbol{\nu}}$, respectively.

2. In the expectation step of the EM algorithm, shift the weights of the censored data onto the uncensored data using $\widehat{\boldsymbol{\mu}}^{(0)}$ and $\widehat{\boldsymbol{\nu}}^{(0)}$.

3. This shifting increases the weights $\boldsymbol{wx}$ and $\boldsymbol{wy}$ to new values which we denote as $\boldsymbol{Wx}$ and $\boldsymbol{Wy}$, respectively.

4. Apply the maximization step of the EM algorithm to convert $\widehat{\boldsymbol{\mu}}^{(0)}$ and $\widehat{\boldsymbol{\nu}}^{(0)}$ into improved estimates which we denote as $\widehat{\boldsymbol{\mu}}^{(1)}$ and $\widehat{\boldsymbol{\nu}}^{(1)}$, respectively.

5. Set $\widehat{\boldsymbol{\mu}}^{(0)} \leftarrow \widehat{\boldsymbol{\mu}}^{(1)}$ and $\widehat{\boldsymbol{\nu}}^{(0)} \leftarrow \widehat{\boldsymbol{\nu}}^{(1)}$

4.2.2.2 *Expectation Step of the EM Algorithm*

The expectation step of the EM algorithm is similar to the expectation step in the single-hypothesis case. It yields the following equations to be solved, analogous

to (4.19) - (4.21):

$$\widehat{\mu}_i^{(1)} = \frac{W x_i}{\sum\limits_{dx_i=1} w x_i + \boldsymbol{\lambda}^T \sum\limits_{dy_j=1} \left(\, \boldsymbol{g}(tx_i, ty_j) - \boldsymbol{\theta} \, \right) \widehat{\nu}_j^{(0)}} \tag{4.22}$$

$$\widehat{\nu}_j^{(1)} = \frac{W y_j}{\sum\limits_{dy_j=1} w y_j + \boldsymbol{\lambda}^T \sum\limits_{dx_i=1} \left(\, \boldsymbol{g}(tx_i, ty_j) - \boldsymbol{\theta} \, \right) \widehat{\mu}_i^{(0)}} \tag{4.23}$$

$$\left(\, (\widehat{\boldsymbol{\mu}}^{(1)})^T H_1 \, \widehat{\boldsymbol{\nu}}^{(1)}, \, \ldots, \, (\widehat{\boldsymbol{\mu}}^{(1)})^T H_p \, \widehat{\boldsymbol{\nu}}^{(1)} \, \right) = \mathbf{0} \tag{4.24}$$

$$\text{where} \qquad \widehat{\boldsymbol{\mu}}^{(1)} = \left(\, \widehat{\mu}_1^{(1)}, \, \widehat{\mu}_2^{(1)}, \, \ldots, \, \widehat{\mu}_n^{(0)} \, \right)^T \tag{4.25}$$

$$\widehat{\boldsymbol{\nu}}^{(1)} = \left(\, \widehat{\nu}_1^{(1)}, \, \widehat{\nu}_2^{(1)}, \, \ldots, \, \widehat{\nu}_m^{(1)} \, \right)^T \tag{4.26}$$

Note the following in (4.22) - (4.24) above:

1. Positive probability jumps occur only at the uncensored data points.
2. $\widehat{\mu}_i^{(0)}$ and $\widehat{\mu}_i^{(1)}$ are not the same value. Rather $\widehat{\mu}_i^{(0)}$ is an initial estimate and $\widehat{\mu}_i^{(1)}$ is an improved estimate.
3. $\widehat{\nu}_j^{(0)}$ and $\widehat{\nu}_j^{(1)}$ are not the same value. Rather $\widehat{\nu}_j^{(0)}$ is an initial estimate and $\widehat{\nu}_j^{(1)}$ is an improved estimate.

4.2.2.3 *Maximization Step of the EM Algorithm*

The maximization step is an iterative procedure that accomplishes three things with reference to (4.22) - (4.24):

1. $\widehat{\mu}_i^{(0)}$ and $\widehat{\mu}_i^{(1)}$ become equal within some small tolerance.
2. $\widehat{\nu}_j^{(0)}$ and $\widehat{\nu}_j^{(1)}$ become equal within some small tolerance.
3. The LHS of (4.24) becomes equal to $\mathbf{0}$ within some small tolerance.

Each iteration of the maximization runs as follows:

1. In (4.24) substitute the RHS of (4.22) for $\widehat{\boldsymbol{\mu}}^{(1)}$ and the RHS of (4.23) for $\widehat{\boldsymbol{\nu}}^{(1)}$, forming a single equation in $\boldsymbol{\lambda}$.
2. Solve the substituted (4.24) for $\boldsymbol{\lambda}$, using a Newton-Raphson routine.
3. Use $\boldsymbol{\lambda}$ to calculate $\widehat{\boldsymbol{\mu}}^{(1)}$ and $\widehat{\boldsymbol{\nu}}^{(1)}$ in (4.22) and (4.23).
4. Update $\widehat{\boldsymbol{\mu}}^{(0)} \leftarrow \widehat{\boldsymbol{\mu}}^{(1)}$ and $\widehat{\boldsymbol{\nu}}^{(0)} \leftarrow \widehat{\boldsymbol{\nu}}^{(1)}$.
5. Calculate the log-likelihood ℓ_0 as in RHS (3.13) using $\widehat{\boldsymbol{\mu}}^{(1)}$ and $\widehat{\boldsymbol{\nu}}^{(1)}$.
6. If ℓ_0 has not stabilized, return to the expectation step of the EM algorithm.
7. Otherwise, we are done, report $\widehat{\boldsymbol{\mu}}^{(1)}, \widehat{\boldsymbol{\nu}}^{(1)}$, ℓ_o.

4.2.3 Evaluation of Ho

Once we have found ℓ_1 and ℓ_0 we can calculate the log-likelihood ratio as,

$$LLR = \ell_0 - \ell_1. \tag{4.27}$$

Using LLR we can then calculate an approximate p-value for H_o, assuming that $-2LLR$ is asymptotically distributed as $\chi^2_{(p)}$ when H_o is true. The p-value is calculated as $1 - F(-2LLR)$ where F is a $\chi^2_{(p)}$ distribution. We reject H_o with 95% confidence if this p-value is less than 0.05. Otherwise we fail to reject H_o with 95% confidence.

We assume here that the number of degrees of freedom p of the $\chi^2_{(p)}$ distribution is equal to the number of hypotheses. More specifically, we assume that the hypotheses

$$\begin{gathered} H_{o1} : E\big(g_1(X,Y)\big) = \theta_1 \\ \vdots \\ H_{op} : E\big(g_p(X,Y)\big) = \theta_p \end{gathered}$$

are unrelated, so that

$$\sqrt{\frac{nm}{n+m}} \begin{pmatrix} \frac{1}{nm} \sum_{dx_i=1} \sum_{dy_j=1} \big(g_1(tx_i, ty_j) - \theta_1\big) \\ \vdots \\ \frac{1}{nm} \sum_{dx_i=1} \sum_{dy_j=1} \big(g_p(tx_i, ty_j) - \theta_p\big) \end{pmatrix} \xrightarrow{d} N(\mathbf{0}_p, \Sigma_{(pxp)})$$

where $\Sigma_{(pxp)}$ is non-singular.

Chapter 5 Examples of Use

5.1 Examples for el2.cen.EMs

We present four examples to illustrate how the el2.cen.EMs function can be used with right-censored data.

Example 1: Difference between two means

This example uses two simulated right-censored data sets. Both were generated from the same Weibull distribution using the Simulatr function listed in Appendix B. The mean of the distribution is 1060. The two data sets are as follows:

```
tx=c(3,50,70,92,233,263,353,466,490,493,605,705,744,764,784,918,1041,
1071,1107,1212,1227,1233,1336,1475,1491,1547,1667,1708,1816,1942)

dx=c(1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,0,1,0,0,1,0,0,0,0,0,0)

ty=c(31,37,73,78,105,125,237,251,261,277,280,293,295,419,515,637,756,
805,984,1016,1199,1210,1257,1344,1464,1473,1480,1516,1912,1994)

dy=c(0,1,0,1,0,0,1,1,1,0,1,1,1,1,1,0,0,1,1,0,1,0,1,1,1,1,1,1,1,1)
```

In this example we suppose that ***tx*** is a control data set and ***ty*** is a treatment data set. We wish to test whether the mean of the control ***tx*** is different from the mean of the treatment ***ty***. The null hypothesis can therefore be written as $H_o : E(X) - E(Y) = 0$ or equivalently as $H_o : E(X - Y) = 0$, as in (3.2). H_o is of

course true since both data sets were simulated from the same distribution.

We execute el2.cen.EMs as follows:

```
el2.cen.EMs(tx,dx,ty,dy,fun=function(tx,ty){tx-ty},mean=0)
```

The resultant nonparametric maximum-likelihood estimator (NPMLE) of the difference in means is 31. The resultant $Pval$ is 0.86, so we cannot with 95% confidence reject H_o.

Example 2: Confidence interval for the difference between two means

We can invert the test in Example 1 to find an equal-tails 95% confidence interval for the difference between the two means. Recall that $-2LLR$ is approximately distributed as $\chi^2_{(1)}$, which has only a single tail. Therefore we search by trial-and-error on either side of the NPMLE to find p-values of 0.05. The results are as follows:

```
el2.cen.EMs(tx,dx,ty,dy,fun=function(tx,ty){tx-ty},mean=-318)
```

gives a p-value of 0.05.

```
el2.cen.EMs(tx,dx,ty,dy,fun=function(tx,ty){tx-ty},mean=380)
```

gives a p-value of 0.05.

So the equal-tails 95% confidence interval for the difference between the two means is [-318, 380].

Example 3: Area under the ROC curve

Receiver-Operating-Characteristic (ROC) curves are often used to assess the accuracy of a test that discriminates between diseased and normal subjects. The area under the ROC curve falls between 0.5 and 1.0. A value of 0.5 indicates a useless test (no better than a coin-flip). A value of 1.0 indicates a perfect test. See Hadley and McNeil (1982)[6] and Krzanowski and Hand (2009)[9] for a discussion of the area under the ROC curve. See Wang et. al. (2009)[19] for a discussion of the area under the ROC curve with censored data.

The area under the ROC curve for subjects X and Y can be interpreted as $P(X > Y)$. The el2.cen.EMs function can be used to estimate $P(X > Y)$ with censored data. Consider the following two right-censored data sets:

```
tx=c(5,6,13,25,35,62,146,204,248,296,309,331,365,411,436,504,665,
851,979,1069,1102,1195,1257,1463,1563,1594,1688,1813,1849,1917)

dx=c(1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,0,0,1,1,1,0,0,0,0,0)

ty=c(7,20,41,48,57,82,83,84,106,113,133,134,153,153,172,185,194,
207,208,253,262,279,346,391,772,1086,1524,1578,1768,1792)

dy=c(1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,0,1,0)
```

Similar to (3.3) we execute el2.cen.EMs as follows (where mean=0.75 is simply a guess at the true mean, midway between 0.5 and 1.0):

```
el2.cen.EMs(tx,dx,ty,dy,fun=function(tx,ty){tx>ty},mean=0.75)
```

The resultant NPMLE of $P(X > Y)$ is 0.67. An equal-tails 95% confidence interval is [0.52, 0.80], found as in Example 2 above.

Example 4: Change-point analysis

Change point analysis is performed on time-ordered data to detect whether any changes have occurred. The number of changes is discerned and the times of the changes are estimated.

We offer here an example involving a control chart which monitors a certain process-characteristic versus time. There are 60 time points on the chart. All points are observed, hence this is uncensored data. The mean of the process has apparently shifted higher at some time point. We wish to infer the most likely value of k, where k is the last time-point of sample $\boldsymbol{tx}$ which follows distribution F_1. Subsequent time points of sample $\boldsymbol{ty}$ follow the distribution F_2 whose mean is higher. Based on our knowledge of the process we believe that k lies in the set 28, 29, 30, 31, 32. So these are the time points we wish to investigate as candidates for k.

A plot of the control chart is shown in Figure 5.1 below. It would be difficult to discern the most likely value of k merely by looking at the chart.

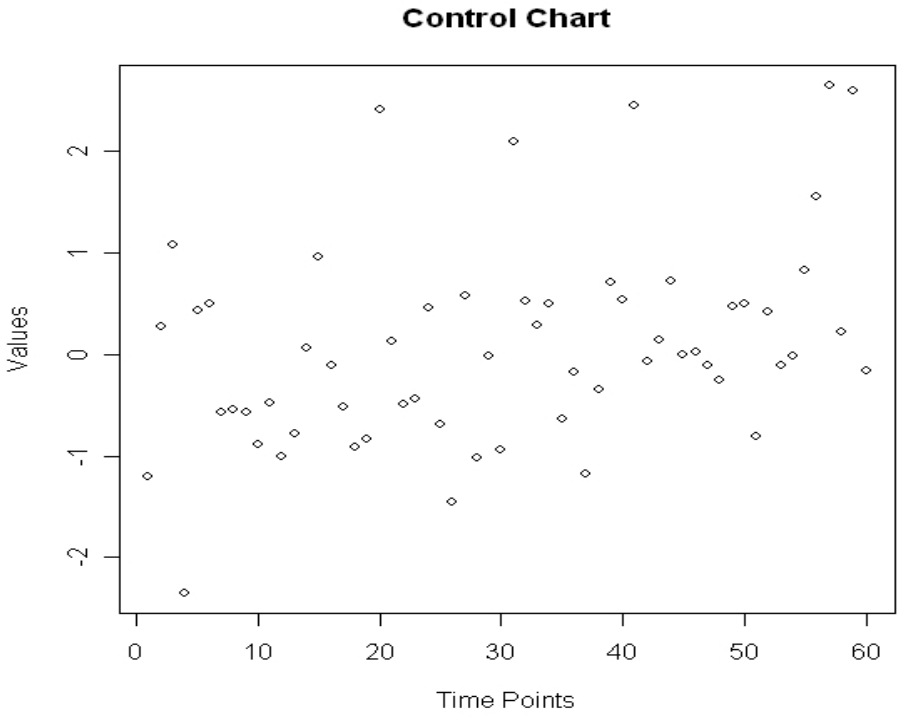

Figure 5.1: Control Chart for Change-Point Example

The R-code for the estimation of k using el2.cen.EMs is shown in Appendix B. The p-values for the 5 possible values of k are (0.03, 0.07, 0.79, 0.32, 0.19) corresponding to time-points (28, 29, 30, 31, 32) respectively. Therefore time-point 30 is the most likely value for k since it corresponds to the highest p-value of 0.79. In fact time-point 30 is the correct value, so the analysis is exactly correct in this example.

The NPMLE for the difference in means returned by the program, that is, $E(X - Y)$, is 0.58. This is reasonably close to the correct value of 1.

5.2 Example for el2.cen.EMm

We have a doubly-censored data set ***tx*** and a doubly-censored data set ***ty***. Both data sets were generated from the same Weibull distribution using the Simulatd function listed in Appendix B. The mean of the distribution is 1060. The two data sets are as follows:

```
tx<-c(10,80,209,273,279,324,391,415,566,85,852,881,895,954,1101,1133,
1337,1393,1408,1444,1513,1585, 1669,1823,1941)

dx<-c(1,2,1,1,1,1,1,2,1,1,1,1,1,1,1,0,0,1,0,0,0,0,1,1,0)

ty<-c(21,38,39,51,77,185,240,289,524,610,612,677,798,881,899,946,1010,
1074,1147,1154,1199,1269,1329,1484,1493,1559,1602,1684,1900,1952)

dy<-c(1,1,1,1,1,1,2,2,1,1,1,1,1,2,1,1,1,1,1,1,0,0,1,1,0,0,1,0,0,0)
```

We test the following set of hypotheses:

$$H_{o1} : P(X \geq Y) = 0.5$$
$$H_{o2} : E(X - 1060) = 0$$

In this case of course H_{o1} and H_{o2} are both true since both data sets were simulated from the same distribution with mean 1060.

We must first calculate arguments for the el2.cen.EMm function as follows:

```
nx<-length(tx)
ny<-length(ty)
```

```
mymaxit<-10
mymean<-c(0.5,0)
p<-2
H1<-matrix(NA,nrow=nx,ncol=ny)
H2<-matrix(NA,nrow=nx,ncol=ny)
for (i in 1:nx) {
  for (j in 1:ny) {
   H1[i,j]<-(tx[i]>ty[j])
   H2[i,j]<-(tx[i]-ty[j]) } }
H<-matrix(c(H1,H2),nrow=nx,ncol=p*ny)
```

Then we execute el2.cen.EMm as follows:

```
el2.cen.EMm(tx, dx, ty, dy, p, H, mean=mymean, maxit=mymaxit)
```

The resultant $Pval$ is 0.73, so we cannot with 95% confidence reject the two simultaneous hypotheses $Ho1$ and $Ho2$. The NPMLE for the mean is (0.51, 58).

Chapter 6 Hazard-Type Hypothesis

In this chapter we prove a theorem involving a hypothesis based on the cumulative hazard function.

Recall that the hypothesis treated in the previous chapters took the form

$$\iint H(t,s)\ dF_X(t)\ dF_Y(s) = \theta\ .$$

The hypothesis treated in this chapter is of an analogous form,

$$\iint H(t,s)\ d\Lambda_X(t)\ d\Lambda_Y(s) = \theta$$

where in place of the cumulative distribution functions $F_X(\cdot),\ F_Y(\cdot)$ we have employed the respective cumulative hazard functions $\Lambda_X(\cdot),\ \Lambda_Y(\cdot)$.

The relationship between the cumulative distribution function $F_X(t)$ and the cumulative hazard function $\Lambda_X(t)$ is as follows:

$$\Lambda_X(t) = \int_0^t \frac{dF_X(u)}{1 - F_X(u)}.$$

$\Lambda_Y(t)$ is defined similarly.

We believe the treatment of this hazard-type hypothesis is a valuable theoretical contribution in its own right. In addition we believe it represents a step toward establishing a relationship between mean-type hypotheses and hazard-type hypotheses for

right-censored data. Developing such a relationship is valuable because hazard-type hypotheses are typically more mathematically tractable than mean-type hypotheses, for censored data. In this scenario the object would be to translate a mean-type hypothesis into a corresponding hazard-type hypothesis; acquire the results for the hazard-type hypothesis; and then translate those results back into results for the mean-type hypothesis.

The proof of the theorem just below is collaborative effort of Ph.D. candidate Ms. Yanling Hu and myself, under the supervision of our advisor Dr. Mai Zhou. Integral signs without explicit limits are understood to encompass the entire support of the variable, here and elsewhere.

Theorem(Hu and Barton): *Suppose* $\boldsymbol{X} = X_1, \ldots, X_n \sim$ iid F_1 *and* $\boldsymbol{Y} = Y_1, \ldots, Y_m \sim$ iid F_2 *are continuous, independent, random variables subject to right-censoring. There is a data sample* $\boldsymbol{x} = (x_1, \ldots, x_n)$ *from* F_1 *and a data sample* $\boldsymbol{y} = (y_1, \ldots, y_m)$ *from* F_2*. Suppose further that there is a hypothesized constraint of the following form:*

$$H_o : \iint H(t,s)\ d\Lambda_X(t)\ d\Lambda_Y(s) = \theta \tag{6.1}$$

where $\Lambda_X(t)$ *and* $\Lambda_Y(s)$ *are the respective cumulative-hazard functions of* X *and* Y*;* $H(t,s)$ *is a continuous function; and* θ *is the true value (that is,* $\theta = \iint H d\Lambda_X d\Lambda_Y$*). If* $H(t,s)$ *satisfies certain regularity conditions, so that* $\iint H(t,s)\ d\Lambda_X(s)\ d\Lambda_Y(t) < \infty$*, then under* H_o*, as* $\min(n,m) \to \infty$*, the distribution of the log-likelihood ratio (LLR) has the limit*

$$-2LLR \xrightarrow{d} \chi^2_{(1)}$$

where LLR is as defined in (3.30)*.*

Hazard functions are typically monotone-increasing and unbounded. Therefore the function $H(t,s)$ in (6.1) must approach 0 as $s,t \to \infty$ so that the integral will be finite. Three examples of such an $H(t,s)$ are $e^{-|t+s|}$, $e^{-|ts|}$, and $g(t,s)I[t<n]I[s<m]$.

In order to evaluate H_o in the above theorem we will calculate a constraint, similar in form to (6.1), but based on the data. The expression for this constraint is as follows:

$$\sum_{i=1}^{n}\sum_{j=1}^{m} H_{ij}\widehat{w}_i\widehat{\nu}_j - \theta = 0 \tag{6.2}$$

where $\widehat{w}_i$ and $\widehat{\nu}_j$ are the estimated jumps in hazard at x_i and y_j, respectively; and H_{ij} is an abbreviated notation for $H(x_i,y_j)$.

If we take the integrals in (6.1) to be Stieltjes integrals then (6.1) and (6.2) are fundamentally similar, only (6.1) applies to continuous (t,s) whereas (6.2) applies to discrete (x_i,y_j).

We organize the proof of the above Theorem into five sections. In section 6.1 we calculate maximum-likelihood estimates (MLE's) of the hazard jumps for each of the two samples. In section 6.2 we calculate the log-likelihood-ratio (LLR) for the two samples. In section 6.3 we show that $-2LLR$ is asymptotically distributed as $\chi^2_{(1)}$ under H_o. In section 6.4 we demonstrate that the Taylor expansions used in the calculations are valid. In section 6.5 we prove two lemmas used in the previous sections.

6.1 MLE's for the Hazard Jumps

In this section we calculate the maximum-likelihood-estimators (MLE's) for the hazard jumps in (6.1) and (6.2).

The calculation of the MLE's for the hazard jumps generally follows the derivation in Owen (2001) [12] pp. 223-227. There are some differences, however, since Owen's derivation involves a mean-type hypothesis for uncensored data, whereas our derivation involves a hazard-type hypothesis for right-censored data. Since the calculations are based on likelihood, we first discuss the likelihood expression that we will employ.

The empirical likelihoods EL_X and EL_Y for the two samples $\boldsymbol{x}$ and $\boldsymbol{y}$ can be written as follows:

$$EL_X = \prod_{i=1}^{n} {w_i}^{dx_i} \exp\Big(-\sum_{r=1}^{n} w_r \; I[x_r \le x_i] \Big) \tag{6.3}$$

$$EL_Y = \prod_{j=1}^{m} {\nu_j}^{dy_j} \exp\Big(-\sum_{s=1}^{m} \nu_s \; I[y_s \le y_j] \Big) \tag{6.4}$$

where $\boldsymbol{dx} = (dx_1, \ldots, dx_n)$ and $\boldsymbol{dy} = (dy_1, \ldots, dy_m)$ are the respective censoring indicators for $\boldsymbol{x}$ and $\boldsymbol{y}$. For $\boldsymbol{x}$ we use the convention that $dx_i = 1$ for an uncensored datum and $dx_i = 0$ for a right-censored datum, and similarly for $\boldsymbol{y}$.

The likelihood expression as in (6.3) and (6.4) is described in Murphy (1995) [10]. It is a "Poisson extension" of the usual likelihood. Although it is not a true likelihood, it does yield a meaningful likelihood-ratio test and it has the advantage that the log-likelihood is easily formed from it. See also Pan and Zhou(2002) [14] for a brief discussion of this Poisson extension.

The log-empirical-likelihoods ($\log EL_X$ and $\log EL_Y$) for the two samples can then be expressed as

$$\log EL_X = \sum_{i=1}^{n} \Big(dx_i \log w_i - \sum_{r=1}^{n} w_r \; I[x_r \leq x_i] \Big) \quad \text{from (6.3)} \tag{6.5}$$

$$\log EL_Y = \sum_{j=1}^{m} \Big(dy_j \log \nu_j - \sum_{s=1}^{m} \nu_s \; I[y_s \leq y_j] \Big) \quad \text{from (6.4).} \tag{6.6}$$

Since the two samples are independent, therefore the log-empirical-likelihood for the set of both samples, denoted as $\log EL$, can be expressed as the sum of (6.5) and (6.6) as follows:

$$\log EL = \log EL_X + \log EL_Y. \tag{6.7}$$

The maximization of the unconstrained likelihood in (6.5) is easily accomplished by taking a partial derivative with respect to w_i and then equating the partial derivative to zero. The maximization of the unconstrained likelihood in (6.6) is accomplished similarly. Expressions for the respective unconstrained hazard-jump MLE's $\widetilde{w}_i$ and $\widetilde{\nu}_j$ are as follows:

$$\widetilde{w}_i = \frac{dx_i}{Rx_i}, \; i = 1, \ldots, n \tag{6.8}$$

$$\widetilde{\nu}_j = \frac{dy_j}{Ry_j}, \; j = 1, \ldots, m. \tag{6.9}$$

Rx_i and Ry_j are the number of survivors at x_i^- and y_j^- respectively, defined as,

$$Rx_i = \sum_{r=1}^{n} I[x_r \geq x_i] \quad \text{and} \quad Ry_j = \sum_{s=1}^{m} I[y_s \geq y_j]. \tag{6.10}$$

The estimators in (6.8) and (6.9) are commonly referred to as Nelson-Aalen estimators (e.g. Kalbfleisch and Prentice [7] p. 18). They are sometimes notated as $d\widehat{\Lambda}_X(x_i)$ and $d\widehat{\Lambda}_Y(y_j)$, respectively, and we will adopt this latter notation in section 6.3.

The calculation of the MLE's for the constrained hazard jumps, which is considerably more complicated, will comprise the rest of this section 6.1.

We use $\log EL$ in (6.7) and the constraint in (6.2) to construct a constrained-log-likelihood target-function G. The constraint in (6.2) is incorporated into G by means of a Lagrange multiplier λ. The result is as follows:

$$G = \log EL - \lambda\Big(\sum_{i=1}^{n}\sum_{j=1}^{m} H_{ij} w_i \nu_j - \theta\Big). \tag{6.11}$$

We calculate the constrained MLE's of w_i, ν_j, and λ by taking partial derivatives of G with respect to w_i, ν_j, and λ and then equating these partial derivatives to zero. We denote the resulting three constrained MLE's as $\widehat{w}_i, \widehat{\nu}_j$, and $\widehat{\lambda}$, respectively.

In the course of calculating the constrained MLE's of w_i, ν_j, and λ we will make use of the following two quantities:

$$\widetilde{H}_{i\cdot} = \sum_{j=1}^{m} \widehat{\nu}_j H_{ij} \qquad \text{and} \qquad \widetilde{H}_{\cdot j} = \sum_{i=1}^{n} \widehat{w}_i H_{ij}. \tag{6.12}$$

Note that $\widetilde{H}_{i\cdot}$ and $\widetilde{H}_{\cdot j}$ are uniformly bounded since H_{ij} is necessarily uniformly

bounded per (6.1) above.

We proceed as follows:

$$\frac{\partial G}{\partial w_i}\Big|_{w_i=\widehat{w}_i,\ \nu_j=\widehat{\nu}_j,\ \lambda=\widehat{\lambda}} \ = \ 0. \tag{6.13}$$

Combining (6.13), (6.5)-(6.7) and (6.11) gives

$$\frac{dx_i}{\widehat{w}_i} - \sum_{r=1}^{n} I[x_r \geq x_i] - \widehat{\lambda} \sum_{j=1}^{m} H_{ij}\widehat{\nu}_j \ = \ 0. \tag{6.14}$$

Rearranging (6.14) gives

$$\widehat{w}_i = \frac{dx_i}{\left(Rx_i + \widehat{\lambda}\widetilde{H}_{i.}\right)} \tag{6.15}$$

using (6.10) and (6.12).

Similarly we set

$$\frac{\partial G}{\partial \nu_j}\Big|_{w_i=\widehat{w}_i,\nu_j=\widehat{\nu}_j,\lambda=\widehat{\lambda}} \ = \ 0. \tag{6.16}$$

Combining (6.16), (6.5)-(6.7) and (6.11) gives

$$\frac{dy_j}{\widehat{\nu}_j} - \sum_{s=1}^{m} I[y_s \geq y_j] - \widehat{\lambda} \sum_{i=1}^{n} H_{ij}\widehat{w}_i \ = \ 0 \tag{6.17}$$

Rearranging (6.17) gives

$$\widehat{\nu}_j = \frac{dy_j}{\left(Ry_j + \widehat{\lambda}\widetilde{H}_{.j}\right)}. \tag{6.18}$$

using (6.10) and (6.12).

The second derivatives $\frac{\partial^2 G}{\partial w_i^2}$ and $\frac{\partial^2 G}{\partial \nu_j^2}$ are easily shown to be negative, confirming that $\widehat{w}_i$ and $\widehat{\nu}_j$ are indeed maxima.

Lastly we set

$$\frac{\partial G}{\partial \lambda}\Big|_{w_i=\widehat{w}_i,\nu_j=\widehat{\nu}_j,\lambda=\widehat{\lambda}} \;=\; 0. \tag{6.19}$$

Combining (6.19), (6.5)-(6.7) and (6.11) gives

$$\sum_{i=1}^{n}\sum_{j=1}^{m} H_{ij}\widehat{w}_i\widehat{\nu}_j = \theta \tag{6.20}$$

which is just the constraint as in (6.2).

Equations (6.15), (6.18), and (6.20) must be solved simultaneously to find explicit expressions for $\widehat{w}_i$, $\widehat{\nu}_j$, and $\widehat{\lambda}$. A precise solution would require numerical methods. We instead calculate an approximate solution using Taylor expansions. We begin by finding an approximate value of $\widehat{\lambda}$.

First we expand $\widehat{w}_i$ and $\widehat{\nu}_j$ in (6.15) and (6.18) in order to bring $\widehat{\lambda}$ into the numerator. This results in the following two expressions:

$$\widehat{w}_i = \frac{dx_i}{Rx_i}\Big[1 - \Big(\frac{\widehat{\lambda}}{Rx_i}\widetilde{H}_{i.}\Big) + \Big(\frac{\widehat{\lambda}}{Rx_i}\widetilde{H}_{i.}\Big)^2 - \dots\Big] \tag{6.21}$$

$$\widehat{\nu}_j = \frac{dy_j}{Ry_j}\Big[1 - \Big(\frac{\widehat{\lambda}}{Ry_j}\widetilde{H}_{.j}\Big) + \Big(\frac{\widehat{\lambda}}{Ry_j}\widetilde{H}_{.j}\Big)^2 - \dots\Big] \tag{6.22}$$

where we have used the Taylor expansion

$$\frac{1}{1+\epsilon} = 1 - \epsilon + \epsilon^2 - \ldots, \quad \text{valid for } |\epsilon| < 1. \tag{6.23}$$

Later in section 6.4 we will show that

$$\frac{\widehat{\lambda}}{Rx_i}\widetilde{H}_{i\cdot} \xrightarrow{p} 0 \quad \text{as } \min(n,m) \to \infty \tag{6.24}$$

$$\frac{\widehat{\lambda}}{Ry_j}\widetilde{H}_{\cdot j} \xrightarrow{p} 0 \quad \text{as } \min(n,m) \to \infty \tag{6.25}$$

so that the restriction $|\epsilon| < 1$ in (6.23) will hold when $\min(n,m)$ is sufficiently large. Also, (6.24) and (6.25) will allow us to drop the higher-order terms in (6.21) and (6.22) as asymptotically negligible.

To simplify the notation in (6.21) and (6.22) let

$$\eta_i = \frac{\widehat{\lambda}}{Rx_i}\widetilde{H}_{i\cdot} \tag{6.26}$$

$$\kappa_j = \frac{\widehat{\lambda}}{Ry_j}\widetilde{H}_{\cdot j}\ . \tag{6.27}$$

Then (6.21) and (6.22) can be written as

$$\widehat{w}_i = \frac{dx_i}{Rx_i}\Big(1 - \eta_i + \eta_i^2 - \ldots\Big) \tag{6.28}$$

$$\widehat{\nu}_j = \frac{dy_j}{Ry_j}\Big(1 - \kappa_j + \kappa_j^2 - \ldots\Big) \tag{6.29}$$

where

$$\eta_i \xrightarrow{p} 0 \quad \text{as } \min(n,m) \to \infty, \quad \text{from (6.24) and (6.26)} \tag{6.30}$$

$$\kappa_j \xrightarrow{p} 0 \quad \text{as } \min(n,m) \to \infty, \quad \text{from (6.25) and (6.27).} \tag{6.31}$$

Substituting (6.28) and (6.29) into (6.20) gives the following:

$$\theta = \sum_{i=1}^{n}\sum_{j=1}^{m} H_{ij}\frac{dx_i}{Rx_i}\frac{dy_j}{Ry_j}\Big(1-\eta_i+\eta_i^2-\ldots\Big)\Big(1-\kappa_j+\kappa_j^2-\ldots\Big) \tag{6.32}$$

$$= \overline{H}_{\cdot\cdot} \; - \; \sum_{i=1}^{n}\sum_{j=1}^{m} H_{ij}\frac{dx_i}{Rx_i}\frac{dy_j}{Ry_j}\Big(\eta_i+\kappa_j\Big)$$

$$+ \sum_{i=1}^{n}\sum_{j=1}^{m} H_{ij}\frac{dx_i}{Rx_i}\frac{dy_j}{Ry_j}\Big(\eta_i^2+\kappa_j^2+\eta_i\kappa_j\Big) \; - \; \ldots \tag{6.33}$$

$$\text{where} \quad \overline{H}_{\cdot\cdot} = \sum_{i=1}^{n}\sum_{j=1}^{m} H_{ij}\frac{dx_i}{Rx_i}\frac{dy_j}{Ry_j} \tag{6.34}$$

$$\doteq \overline{H}_{\cdot\cdot} - \sum_{i=1}^{n}\sum_{j=1}^{m} H_{ij}\frac{dx_i}{Rx_i}\frac{dy_j}{Ry_j}\Big(\eta_i+\kappa_j\Big), \tag{6.35}$$

dropping higher order terms per (6.30) and (6.31).

Now η_i and κ_j in the right-hand side (RHS) of (6.35) are themselves functions of $\widehat{\lambda}$, per (6.26) and (6.27) above. In order to facilitate the objective of finding an explicit approximation for $\widehat{\lambda}$ we introduce the terms $\overline{H}_{i\cdot}$ and $\overline{H}_{\cdot j}$, which are not functions of $\widehat{\lambda}$, as follows:

$$\overline{H}_{i\cdot} = \sum_{j=1}^{m} H_{ij}\frac{dy_j}{Ry_j} \quad \text{and} \quad \overline{H}_{\cdot j} = \sum_{i=1}^{n} H_{ij}\frac{dx_i}{Rx_i}. \tag{6.36}$$

We obtain (6.36) by substituting $\frac{dy_j}{Ry_j}$ for $\widehat{\nu_j}$ and $\frac{dx_i}{Rx_i}$ for $\widehat{\mu_i}$ in (6.12). Note that $\overline{H}_{i\cdot}$ and $\overline{H}_{\cdot j}$ are uniformly bounded since H_{ij} is assumed to be uniformly bounded per (6.1) above.

Now consider the following:

$$\widetilde{H}_{i.} = \overline{H}_{i.} + (\widetilde{H}_{i.} - \overline{H}_{i.}) \tag{6.37}$$

$$= \overline{H}_{i.} + \sum_{j=1}^{m} H_{ij}\Big(\widehat{\nu}_j - \frac{dy_j}{Ry_j}\Big) \quad \text{from (6.12) and (6.36)} \tag{6.38}$$

$$= \overline{H}_{i.} + \sum_{j=1}^{m} H_{ij}\Big[\ \frac{dy_j}{Ry_j}\big(1 - \kappa_j + \kappa_j^2 - \dots\big) - \frac{dy_j}{Ry_j}\ \Big] \quad \text{from (6.29)} \tag{6.39}$$

$$= \overline{H}_{i.} - \sum_{j=1}^{m} H_{ij}\frac{dy_j}{Ry_j}\big(\kappa_j - \kappa_j^2 + \dots\big) \tag{6.40}$$

$$= \overline{H}_{i.} - \sum_{j=1}^{m} H_{ij}\frac{dy_j}{Ry_j}\ o_p(1) \quad \text{from (6.31), assuming } \max_{1\le j\le m}(\kappa_j) = o_p(1) \tag{6.41}$$

$$= \big(\ 1 - o_p(1)\ \big)\ \overline{H}_{i.} \quad \text{from (6.36).} \tag{6.42}$$

Similarly we can calculate

$$\widetilde{H}_{.j} = \big(\ 1 - o_p(1)\ \big)\ \overline{H}_{.j} \qquad \text{assuming } \max_{1\le i\le n}(\eta_i) = o_p(1). \tag{6.43}$$

Then we can estimate $\widehat{\lambda}$ as follows:

$$\theta \doteq \overline{H}_{..} - \sum_{i=1}^{n}\sum_{j=1}^{m} H_{ij}\frac{dx_i}{Rx_i}\frac{dy_j}{Ry_j}\Big(\eta_i + \kappa_j\Big) \quad \text{from (6.35)} \tag{6.44}$$

$$= \overline{H}_{..} - \sum_{i=1}^{n}\sum_{j=1}^{m} H_{ij}\frac{dx_i}{Rx_i}\frac{dy_j}{Ry_j}\Big(\frac{\widehat{\lambda}}{Rx_i}\widetilde{H}_{i.} + \frac{\widehat{\lambda}}{Ry_j}\widetilde{H}_{.j}\Big) \quad \text{from (6.26), (6.27)} \tag{6.45}$$

$$= \overline{H}_{\cdot\cdot} - \sum_{i=1}^{n}\sum_{j=1}^{m} H_{ij}\frac{dx_i}{Rx_i}\frac{dy_j}{Ry_j}\left[\frac{\widehat{\lambda}}{Rx_i}\overline{H}_{i\cdot}\left(1-o_p(1)\right)+\frac{\widehat{\lambda}}{Ry_j}\overline{H}_{\cdot j}\left(1-o_p(1)\right)\right] \tag{6.46}$$

from (6.42), (6.43)

$$= \overline{H}_{\cdot\cdot} - \widehat{\lambda}\left(1-o_p(1)\right)\sum_{i=1}^{n}\sum_{j=1}^{m} H_{ij}\frac{dx_i}{Rx_i}\frac{dy_j}{Ry_j}\left(\frac{\overline{H}_{i\cdot}}{Rx_i}+\frac{\overline{H}_{\cdot j}}{Ry_j}\right) \tag{6.47}$$

$$\doteq \overline{H}_{\cdot\cdot} - \widehat{\lambda}\sum_{i=1}^{n}\sum_{j=1}^{m} H_{ij}\frac{dx_i}{Rx_i}\frac{dy_j}{Ry_j}\left(\frac{\overline{H}_{i\cdot}}{Rx_i}+\frac{\overline{H}_{\cdot j}}{Ry_j}\right). \tag{6.48}$$

Then from (6.48) the desired approximation of λ is,

$$\widehat{\lambda} \doteq (\overline{H}_{\cdot\cdot} - \theta)/D \tag{6.49}$$

$$\text{where} \quad D = \sum_{i=1}^{n}\sum_{j=1}^{m} H_{ij}\frac{dx_i}{Rx_i}\frac{dy_j}{Ry_j}\left(\frac{\overline{H}_{i\cdot}}{Rx_i}+\frac{\overline{H}_{\cdot j}}{Ry_j}\right) \tag{6.50}$$

The RHS of (6.49) does not involve $\widehat{w}_i, \widehat{\nu}_j$, or $\widehat{\lambda}$, rather it only involves the data. Therefore the RHS of (6.49) is an explicit approximation of $\widehat{\lambda}$.

By a similar argument we can substitute (6.42) and (6.43) into (6.21) and (6.22), respectively. This gives an approximation $\breve{w}_i$ for $\widehat{w}_i$ and an approximation $\breve{\nu}_j$ for $\widehat{\nu}_j$, as follows:

$$\breve{w}_i = \frac{dx_i}{Rx_i}\left(1-\frac{\widehat{\lambda}}{Rx_i}\overline{H}_{i\cdot}\right) \doteq \widehat{w}_i \tag{6.51}$$

$$\breve{\nu}_j = \frac{dy_j}{Ry_j}\left(1-\frac{\widehat{\lambda}}{Ry_j}\overline{H}_{\cdot j}\right) \doteq \widehat{\nu}_j. \tag{6.52}$$

6.2 LLR for the Two Samples

In this section we calculate an expression for the log-likelihood-ratio (LLR) for the two samples under the constraint in (6.1).

Let us define the following:

$$\breve{\eta}_i = \frac{\widehat{\lambda}}{Rx_i}\overline{H}_{i.}, \quad i = 1, \ldots, n \quad \text{similar to (6.26)} \tag{6.53}$$

$$\breve{\boldsymbol{\eta}} = (\breve{\eta}_1, \breve{\eta}_2, \ldots, \breve{\eta}_n) \tag{6.54}$$

$$Q(\breve{\boldsymbol{\eta}}) = \sum_{i=1}^{n}\left(dx_i \log\left(\frac{dx_i}{Rx_i}(1-\breve{\eta}_i)\right) - \sum_{r=1}^{n}\frac{dx_r}{Rx_r}(1-\breve{\eta}_r)I[x_r \le x_i]\right) \tag{6.55}$$

$$Q_o = Q(\mathbf{0}) = \sum_{i=1}^{n}\Big(dx_i \log(\frac{dx_i}{Rx_i}) - \sum_{r=1}^{n}\frac{dx_r}{Rx_r}I[x_r \le x_i]\Big). \tag{6.56}$$

From (6.5), (6.8), (6.51), and (6.53) we see that $Q(\breve{\boldsymbol{\eta}})$ in (6.55) is an estimator for the constrained log-likelihood for the sample $\boldsymbol{x}$ and we see that Q_o in (6.56) is an estimator for the unconstrained log-likelihood for the sample $\boldsymbol{x}$.

Similarly let us define the following:

$$\breve{\kappa}_j = \frac{\widehat{\lambda}}{Ry_j}\overline{H}_{.j}, \quad j = 1, \ldots, m \quad \text{similar to (6.27)} \tag{6.57}$$

$$\breve{\boldsymbol{\kappa}} = (\breve{\kappa}_1, \breve{\kappa}_2, \ldots, \breve{\kappa_m}) \tag{6.58}$$

$$P(\breve{\boldsymbol{\kappa}}) = \sum_{j=1}^{m}\left(dy_j \log\left(\frac{dy_j}{Ry_j}(1-\breve{\kappa}_j)\right) - \sum_{s=1}^{m}\frac{dy_s}{Ry_s}(1-\breve{\kappa_r}s)I[y_s \le y_j]\right) \tag{6.59}$$

$$P_o = P(\mathbf{0}) = \sum_{j=1}^{m}\Big(dy_j \log(\frac{dy_j}{Ry_j}) - \sum_{s=1}^{m}\frac{dy_s}{Ry_s}I[y_s \le y_j]\Big). \tag{6.60}$$

From (6.6), (6.9), (6.52), and (6.57) we see that $P(\breve{\boldsymbol{\kappa}})$ in (6.59)is an estimator for the constrained log-likelihood for the sample $\boldsymbol{y}$ and we see that P_o in (6.60)is an estimator

for the unconstrained log-likelihood for the sample $\boldsymbol{y}$.

It is convenient to rewrite $Q(\breve{\boldsymbol{\eta}})$ in (6.55) as follows:

$$Q(\breve{\boldsymbol{\eta}}) = \sum_{i=1}^{n} \left(dx_i \log\Big(\frac{dx_i}{Rx_i}(1-\breve{\eta}_i)\Big)\right) - \sum_{i=1}^{n}\sum_{r=1}^{n}\left(\frac{dx_r}{Rx_r}(1-\breve{\eta}_r)I[x_r \le x_i]\right) \tag{6.61}$$

$$= \sum_{i=1}^{n} \left(dx_i \log\Big(\frac{dx_i}{Rx_i}(1-\breve{\eta}_i)\Big)\right) - \sum_{r=1}^{n}\left(\frac{dx_r}{Rx_r}(1-\breve{\eta}_r)\sum_{i=1}^{n} I[x_r \le x_i]\right) \tag{6.62}$$

$$= \sum_{i=1}^{n} \left(dx_i \log\Big(\frac{dx_i}{Rx_i}(1-\breve{\eta}_i)\Big)\right) - \sum_{r=1}^{n}\left(dx_r(1-\breve{\eta}_r)\right) \quad \text{using (6.10)} \tag{6.63}$$

$$= \sum_{i=1}^{n} \left(dx_i \log\Big(\frac{dx_i}{Rx_i}(1-\breve{\eta}_i)\Big) - dx_i(1-\breve{\eta}_i)\right) \text{ changing index r to i} \tag{6.64}$$

$$= \sum_{i=1}^{n} Q_i(\breve{\eta}_i) \tag{6.65}$$

$$\text{where} \quad Q_i(\breve{\eta}_i) = \left(dx_i \log\Big(\frac{dx_i}{Rx_i}(1-\breve{\eta}_i)\Big) - dx_i(1-\breve{\eta}_i)\right). \tag{6.66}$$

By a similar calculation we can rewrite $P(\breve{\boldsymbol{\kappa}})$ in (6.59) as follows:

$$P(\breve{\boldsymbol{\kappa}}) = \sum_{j=1}^{m} P_j(\breve{\kappa}_j) \tag{6.67}$$

$$\text{where} \qquad P_i(\breve{\kappa}_j) = \left(dy_j \log\Big(\frac{dy_j}{Ry_j}(1-\breve{\kappa}_j)\Big) - dy_j(1-\breve{\kappa}_j)\right). \tag{6.68}$$

We will require the quantities $\sum_{i=1}^{n} \breve{\eta}_i Q_i'(0)$, $\sum_{j=1}^{m} \breve{\kappa}_j P_j'(0)$, $\sum_{i=1}^{n} \frac{\breve{\eta}_i{}^2}{2} Q_i''(0)$, and $\sum_{j=1}^{m} \frac{\breve{\kappa}_j{}^2}{2} P_j''(0)$ which we calculate as follows:

$$\sum_{i=1}^{n} \breve{\eta}_i Q_i'(0) = \sum_{i=1}^{n} \breve{\eta}_i \left(\frac{-dx_i \frac{dx_i}{Rx_i}}{\frac{dx_i}{Rx_i}(1-\breve{\eta}_i)} + dx_i \right) \Big|_{\breve{\eta}_i=0} \quad \text{using (6.66)} \tag{6.69}$$

$$= 0 \tag{6.70}$$

A similar calculation shows

$$\sum_{j=1}^{m} \breve{\kappa}_j P_j'(0) = 0 \quad \text{using (6.68)} \tag{6.71}$$

And,

$$\sum_{i=1}^{n} \frac{\breve{\eta}_i{}^2}{2} Q_i''(0) = \sum_{i=1}^{n} \frac{\breve{\eta}_i{}^2}{2} \Big(\frac{-dx_i}{(1-\eta_i)^2} \Big) \Big|_{\breve{\eta}_i=0} \quad \text{using (6.69)} \tag{6.72}$$

$$= -\sum_{i=1}^{n} \frac{\breve{\eta}_i{}^2}{2} dx_i \tag{6.73}$$

A similar calculation shows

$$\sum_{j=1}^{m} \frac{\breve{\kappa}_j{}^2}{2} P_j''(0) = -\sum_{j=1}^{m} \frac{\breve{\kappa}_j{}^2}{2} dy_j \tag{6.74}$$

Then we can calculate $-2LLR$ as follows:

$$-2LLR \doteq -2\Big(Q(\breve{\boldsymbol{\eta}}) - Q_o\Big) - 2\Big(P(\breve{\boldsymbol{\kappa}}) - P_o\Big) \tag{6.75}$$

from (6.5) − (6.7), (6.53) − (6.60)

$$= -2Q(\breve{\boldsymbol{\eta}}) + 2Q_o - 2P(\breve{\boldsymbol{\kappa}}) + 2P_o \tag{6.76}$$

$$\begin{aligned} = & -2\left(Q_o + \sum_{i=1}^{n}\left(\breve{\eta}_i Q_i'(0) + \frac{\breve{\eta}_i{}^2}{2}Q_i''(0)\right)\right) + 2Q_o + R_n(\breve{\boldsymbol{\eta}}) \\ & - 2\left(P_o + \sum_{j=1}^{m}\left(\breve{\kappa}_j P_j'(0) + \frac{\breve{\kappa}_j{}^2}{2}P_j''(0)\right)\right) + 2P_o + R_m(\breve{\boldsymbol{\kappa}}) \end{aligned} \tag{6.77}$$

by Taylor expansion of $Q(\breve{\boldsymbol{\eta}})$ and $P(\breve{\boldsymbol{\kappa}})$ about $\mathbf{0}$,

where $R_n(\breve{\boldsymbol{\eta}})$ and $R_m(\breve{\boldsymbol{\kappa}})$ are the remainder terms

$$\doteq -\sum_{i=1}^{n}\breve{\eta}_i{}^2 Q_i''(0) - \sum_{j=1}^{m}\breve{\kappa}_j{}^2 P_j''(0) \tag{6.78}$$

using (6.70), (6.71), dropping remainder terms

$$= \sum_{i=1}^{n}\breve{\eta}_i{}^2 dx_i + \sum_{j=1}^{m}\breve{\kappa}_j{}^2 dy_j \quad \text{using (6.73), (6.74)} \tag{6.79}$$

$$= \widehat{\lambda}^2\sum_{i=1}^{n}\left(\frac{\overline{H_{i\cdot}}}{Rx_i}\right)^2 dx_i \;+\; \widehat{\lambda}^2\sum_{j=1}^{m}\left(\frac{\overline{H_{\cdot j}}}{Ry_j}\right)^2 dy_j \;\text{ using (6.53), (6.57)} \tag{6.80}$$

$$= \widehat{\lambda}^2\sum_{i=1}^{n}\sum_{j=1}^{m} H_{ij}\frac{dy_j}{Ry_j}\frac{dx_i}{Rx_i}\frac{\overline{H_{i\cdot}}}{Rx_i} \;+\; \widehat{\lambda}^2\sum_{i=1}^{n}\sum_{j=1}^{m} H_{ij}\frac{dx_i}{Rx_i}\frac{dy_j}{Ry_j}\frac{\overline{H_{\cdot j}}}{Ry_j} \tag{6.81}$$

using (6.36)

$$= \widehat{\lambda}^2 D \quad \text{from (6.50)} \tag{6.82}$$

$$\doteq \frac{(\overline{H}_{\cdot\cdot} - \theta)^2}{D} \quad \text{from (6.49)} \tag{6.83}$$

6.3 Distribution of -2LLR

In this section we show that the distribution of -2LLR goes asymptotically to $\chi^2_{(1)}$ as $\min(n, m) \to \infty$, when the null hypothesis H_o is true.

Let us establish the following notation:

Summations are represented as Stieltjes integrals (6.84)

x_i is represented as t (6.85)

y_j is represented as s (6.86)

H_{ij} is represented as $g(t, s)$ (6.87)

$\frac{dx_i}{Rx_i}$ is represented as $d\widehat{\Lambda}_1(t)$, a Nelson-Aalen jump as in (6.8) (6.88)

$\frac{dy_j}{Ry_j}$ is represented as $d\widehat{\Lambda}_2(s)$, a Nelson-Aalen jump as in (6.9) (6.89)

H_o in (6.1) can then be denoted as $\iint g(t, s) d\Lambda_1(t) d\Lambda_2(s) = \theta.$ (6.90)

The derivation below uses Martingale theory and the reader will need to be familiar with this theory in order to follow the derivation. A good reference for Martingale theory is Kalbfleisch and Prentice (2002) [7], chapter 5.

Now consider,

$$\sqrt{\frac{nm}{n+m}}(\overline{H}_{\cdot\cdot} - \theta) = \sqrt{\frac{nm}{n+m}}\Big(\sum_{i=1}^{n}\sum_{j=1}^{m} H_{ij}\frac{dx_i}{Rx_i}\frac{dy_j}{Ry_j} - \theta\Big) \tag{6.91}$$

from (6.34)

$$= \sqrt{\frac{nm}{n+m}}\Big(\iint g(t,s)d\widehat{\Lambda}_1(t)d\widehat{\Lambda}_2(s) - \iint g(t,s)d\Lambda_1(t)d\Lambda_2(s)\Big) \tag{6.92}$$

using (6.84) to (6.90)

$$= \sqrt{\frac{nm}{n+m}}\Big(\iint g(t,s)d\widehat{\Lambda}_1(t)d\widehat{\Lambda}_2(s) - \iint g(t,s)d\Lambda_1(t)d\Lambda_2(s)$$

$$+ \iint g(t,s)d\Lambda_1(t)d\widehat{\Lambda}_2(s) - \iint g(t,s)d\Lambda_1(t)d\widehat{\Lambda}_2(s)\Big) \tag{6.93}$$

add, subtract term

$$= \sqrt{\frac{m}{n+m}}\iint g(t,s)d\Lambda_2(s)\; d\sqrt{n}\Big(\widehat{\Lambda}_1(t) - \Lambda_1(t)\Big)$$

$$+ \sqrt{\frac{n}{n+m}}\iint g(t,s)d\Lambda_1(t)\; d\sqrt{m}\Big(\widehat{\Lambda}_2(s) - \Lambda_2(s)\Big) + o_p(1) \tag{6.94}$$

see explanation in (6.108) to (6.110) below

$$\xrightarrow{d} \sqrt{\frac{m}{n+m}}\int f_1(t)dB_1\Big(C_1(t)\Big) + \sqrt{\frac{n}{n+m}}\int f_2(s)dB_2\Big(C_2(s)\Big) \tag{6.95}$$

as $\min(n,m) \to \infty$, where

$$\int g(t,s)d\Lambda_2(s) \triangleq f_1(t) \tag{6.96}$$

$$\int g(t,s)d\Lambda_1(t) \triangleq f_2(s) \tag{6.97}$$

$$C_1(t) = \int_{\gamma=0}^{t} \frac{d\Lambda_1(\gamma)}{\big(1-F_1(\gamma^-)\big)\big(1-G_1(\gamma^-)\big)} \tag{6.98}$$

$$C_2(s) = \int_{\gamma=0}^{s} \frac{d\Lambda_2(\gamma)}{\big(1-F_2(\gamma^-)\big)\big(1-G_2(\gamma^-)\big)} \tag{6.99}$$

B_1 is a Brownian motion with a "clock"$C_1(t)$ (6.100)

B_2 is a Brownian motion with a "clock"$C_2(s)$ (6.101)

$$\sqrt{n}\Big(\widehat{\Lambda}_1(t) - \Lambda_1(t)\Big) \xrightarrow{d} B_1\big(C_1(t)\big) \quad \text{as in Kalbfleisch [7]} \tag{6.102}$$

$$\sqrt{m}\Big(\widehat{\Lambda}_2(s) - \Lambda_2(s)\Big) \xrightarrow{d} B_2\big(C_2(s)\big) \quad \text{as in Kalbfleisch [7].} \tag{6.103}$$

Therefore,

$$\sqrt{\frac{nm}{n+m}}(\overline{H}_{\cdot\cdot} - \theta) \xrightarrow{d} N(0,\sigma^2) \quad \text{as } \min(n,m) \to \infty, \quad \text{from (6.95)} \tag{6.104}$$

$$\begin{aligned} \text{where } \quad \sigma^2 = \alpha \int f_1^2(t) \frac{d\Lambda_1(t)}{(1-F_1(t^-))(1-G_1(t^-))} \\ + \ (1-\alpha) \int f_2^2(s) \frac{d\Lambda_2(t)}{(1-F_2(s^-))(1-G_2(s^-))} \end{aligned} \tag{6.105}$$

and α is defined as in (6.115) below.

Therefore,

$$\frac{(\overline{H}_{\cdot\cdot} - \theta)}{\sqrt{D}} \xrightarrow{d} N\left(0, \frac{\sigma^2}{\xi}\right) \quad \text{as } \min(n,m) \to \infty, \quad \text{using (6.104)} \tag{6.106}$$

where ξ is a constant such that

$$\frac{nm}{n+m} D \xrightarrow{p} \xi, \text{ as } \min(n,m) \to \infty. \tag{6.107}$$

We demonstrate that (6.107) is true in (6.112) to (6.129) below.

In (6.94) above we use the following relationships:

$$\sqrt{\frac{nm}{n+m}} \iint g(t,s)\ d[\widehat{\Lambda}_1(t) - \Lambda_1(t)]\ d[\widehat{\Lambda}_2(s) - \Lambda_2(s)] \tag{6.108}$$

$$= \sqrt{\frac{1}{n+m}} \iint g(t,s)\ d\sqrt{n}[\widehat{\Lambda}_1(t) - \Lambda_1(t)]\ d\sqrt{m}[\widehat{\Lambda}_2(s) - \Lambda_2(s)] \tag{6.109}$$

$$\xrightarrow{d} \sqrt{\frac{1}{n+m}} \iint g(t,s)\ dB_1(C_1(t))\ dB_2(C_2(s)) \tag{6.110}$$

$$\text{as} \min(n,m) \to \infty$$

$$= o_p(1) \tag{6.111}$$

We subtract (6.108) from (6.111) (thus obtaining zero), add that expression to (6.93), and simplify to obtain (6.94).

With reference to (6.107) above, we demonstrate that $\frac{nm}{n+m} D \xrightarrow{p} \xi$ as $\min(n,m) \to \infty$ as follows:

$$D = \sum_{i=1}^{n} \sum_{j=1}^{m} H_{ij} \frac{dx_i}{(Rx_i)^2} \frac{dy_j}{Ry_j} \overline{H}_{i\cdot} + \sum_{i=1}^{n} \sum_{j=1}^{m} H_{ij} \frac{dx_i}{Rx_i} \frac{dy_j}{(Ry_j)^2} \overline{H}_{\cdot j} \tag{6.112}$$

from (6.50)

$$\frac{nm}{n+m}D = \frac{m}{n+m}\sum_{i=1}^{n}\sum_{j=1}^{m} H_{ij}\frac{dx_i}{Rx_i}\frac{n}{Rx_i}\frac{dy_j}{Ry_j}\overline{H}_{i.} + \frac{n}{n+m}\sum_{i=1}^{n}\sum_{j=1}^{m} H_{ij}\frac{dx_i}{Rx_i}\frac{dy_j}{Ry_j}\frac{m}{Ry_j}\overline{H}_{.j} \tag{6.113}$$

$$\xrightarrow{p} \alpha \iint g(t,s)f_1(t)\frac{1}{(1-F_1(t^-))(1-G_1(t^-))}d\Lambda_1(t)d\Lambda_2(s) \tag{6.114}$$
$$+ (1-\alpha)\iint g(t,s)f_2(s)\frac{1}{(1-F_2(s^-))(1-G_2(s^-))}d\Lambda_1(t)d\Lambda_2(s)$$

as $\min(n,m) \to \infty$, by law of large numbers for Nelson-Aalen estimators, where

We assume that $\frac{m}{n+m} \xrightarrow{p} \alpha$ as $\min(n,m) \to \infty,\ 1 \geq \alpha > 0$ (6.115)

Therefore $\frac{n}{n+m} \xrightarrow{p} 1-\alpha$ as $\min(n,m) \to \infty$ (6.116)

H_{ij} is represented as $g(t,s)$ as in (6.87) (6.117)

$\overline{H}_{i.} \xrightarrow{p} f_1(t)$, where $f_1(t)$ is as in (6.96), per Lemma 1 in section 6.5 (6.118)

$\overline{H}_{.j} \xrightarrow{p} f_2(s)$, where $f_2(s)$ is as in (6.97), per Lemma 1 in section 6.5 (6.119)

$\frac{n}{Rx_i} \xrightarrow{p} \frac{1}{(1-F_1(t^-))(1-G_1(t^-))}$ per Lemma 2 in section 6.5 (6.120)

$\frac{m}{Ry_j} \xrightarrow{p} \frac{1}{(1-F_2(s^-))(1-G_2(s^-))}$ per Lemma 2 in section 6.5 (6.121)

$G_1(t)$ is the censoring distribution of sample 1 (6.122)

$G_2(s)$ is the censoring distribution of sample 2 (6.123)

$\frac{dx_i}{Rx_i}$ is represented as $d\widehat{\Lambda}_1$ as in (6.88) (6.124)

$\frac{dy_j}{Ry_j}$ is represented as $d\widehat{\Lambda}_2$ as in (6.89) (6.125)

$$= \alpha \int f_1^2(t) \frac{d\Lambda_1(t)}{(1-F_1(t^-))(1-G_1(t^-))} \tag{6.126}$$
$$+ (1-\alpha) \int f_2^2(s) \frac{d\Lambda_2(s)}{(1-F_2(s^-))(1-G_2(s^-))} \quad \text{from (6.96) and (6.97)}$$

$$= \sigma^2 \quad \text{from (6.105)} \tag{6.127}$$

Hence,

$$\frac{nm}{n+m} D \to \sigma^2 \text{ as} \min(n,m) \xrightarrow{p} \infty, \text{ from (6.112) to (6.127)} \tag{6.128}$$

Combining (6.106) and (6.127) gives,

$$\frac{(\overline{H}_{..} - \theta)}{\sqrt{D}} \xrightarrow{d} N(0,1) \quad \text{as} \quad \min(n,m) \to \infty \tag{6.129}$$

Squaring both sides of (6.129) gives,

$$\frac{(\overline{H}_{..} - \theta)^2}{D} \xrightarrow{d} \chi^2_{(1)} \quad \text{as } \min(n,m) \to \infty \tag{6.130}$$

Finally, combining (6.83) and (6.130) gives

$$-2LLR \xrightarrow{d} \chi^2_{(1)} \quad \text{as } \min(n,m) \to \infty \tag{6.131}$$

This concludes the main portion of the proof. We can use (6.131) to calculate an approximate p-value to test H_o based on $-2LLR$.

6.4 Validation of the Taylor Expansions

In this section we justify the Taylor expansions in (6.35) by showing that $\frac{\widehat{\lambda}}{Rx_i}\widetilde{H}_{i.} \xrightarrow{p} 0$ as $\min(n,m) \to \infty$, as in (6.24); and $\frac{\widehat{\lambda}}{Ry_j}\widetilde{H}_{.j} \xrightarrow{p} 0$ as $\min(n,m) \to \infty$, as in (6.25).

We will assume that $\widetilde{H}_{i.}$, $\widetilde{H}_{.j}$, $\overline{H}_{i.}$, and $\overline{H}_{.j}$ are all uniformly bounded. We will also assume that $\zeta \geq \frac{n}{m} \geq \frac{1}{\zeta}$ for some positive number ξ. To simplify the calculations below we will additionally assume, without loss of generality, that $\zeta = 1$, so that $n = m$.

First we show that $\widehat{\lambda}^*/n$ becomes small as $n \to \infty$, where $\widehat{\lambda}^*/n$ is the root of $\overline{H}_{..} - \theta - \lambda D = 0$.

From (6.106) and (6.107) we know that $\frac{(\overline{H}_{..}-\theta)}{\sqrt{D}} \xrightarrow{d} N\Big(0, \frac{\sigma^2}{\frac{nm}{n+m}D}\Big)$ and from (6.127) we know that $\frac{nm}{n+m}D \xrightarrow{p} \sigma^2$ as $\min(n,m) \to \infty$. Without loss of generatlity let us assume that $n = m$ so that $\frac{nm}{n+m} = \frac{n}{2}$. Then $\frac{nm}{n+m}D = \frac{n}{2}D$ so that

$$\frac{n}{2}D \xrightarrow{p} \sigma^2 \quad \text{from (6.128)} \tag{6.132}$$

We will show that $\widehat{\lambda}^*/n \in (-\epsilon, \epsilon)$ as $n \to \infty$ for any small $\epsilon > 0$.

Now,

$$\overline{H}_{\cdot\cdot} - \theta - \lambda D = \sqrt{D}\,\Big(\frac{\overline{H}_{\cdot\cdot} - \theta}{\sqrt{D}} - \lambda\sqrt{D}\Big) \tag{6.133}$$

$$\doteq \sqrt{D}\Big(Z - \lambda\sqrt{D}\Big) \qquad \text{where } Z \sim N(0,1) \text{ from (6.129)} \tag{6.134}$$

$$= \sqrt{D}\Big(Z - \lambda\sqrt{2/n}\sqrt{\frac{n}{2}D}\Big) \tag{6.135}$$

$$\doteq \sqrt{D}\big(Z - \frac{\lambda}{\sqrt{n}}\sqrt{2}\;\sigma\big) \qquad \text{from (6.132)} \tag{6.136}$$

Now let us choose a sequence indexed by the natural numbers $(1, 2, \ldots, n, \ldots)$ that goes slowly to ∞. Without loss of generality let us choose the sequence $(\log(1), \log(2), \ldots, \log(n), \ldots)$. Then let $\frac{\lambda}{\sqrt{n}} = \pm\log(n)$ and substitute this into (6.136) above. When $\frac{\lambda}{\sqrt{n}} = \log(n)$ then the RHS of (6.135) is less than 0 for large n, since Z is bounded in probability. And similarly when $\frac{\lambda}{\sqrt{n}} = -\log(n)$ then the RHS of (6.136) is greater than 0 for large n. This implies that $\frac{\widehat{\lambda}^*}{n} \in \big(-\frac{\log(n)}{\sqrt{n}}, \frac{\log(n)}{\sqrt{n}}\big)$ for large n with probability approaching 1, since $\overline{H}_{\cdot\cdot} - \theta - \lambda D$ is monotone in λ. This in turn implies that

$$\frac{\widehat{\lambda}^*}{n} = O_p\Big(\frac{\log(n)}{\sqrt{n}}\Big) \tag{6.137}$$

Now,

$$\lim_{n\to\infty}\Big(\frac{Rx_i}{n}\Big) = (1 - F_1(x_i))(1 - G_1(x_i)) \quad \text{similar to (6.173)} \tag{6.138}$$

which implies that

$$Rx_i = O_p(n). \tag{6.139}$$

Combining (6.137) and (6.139) gives

$$\frac{\widehat{\lambda}^*}{Rx_i} = O_p\Big(\frac{\log(n)}{\sqrt{n}}\Big) \tag{6.140}$$

Then since $\widetilde{H}_{i.}$ and $\widetilde{H}_{.j}$ are bounded, (6.140) implies

$$\frac{\widehat{\lambda}^*}{Rx_i}\widetilde{H}_{i.} = O_p\Big(\frac{\log(n)}{\sqrt{n}}\Big) \quad \text{and} \quad \frac{\widehat{\lambda}^*}{Ry_j}\widetilde{H}_{.j} = O_p\Big(\frac{\log(n)}{\sqrt{n}}\Big) \tag{6.141}$$

Then with reference to (6.33) of order $\widehat{\lambda}$ consider the following:

$$\sum_{i=1}^{n}\sum_{j=1}^{m} H_{ij}\frac{dx_i}{Rx_i}\frac{dy_j}{Ry_j}\Big(\eta_i + \kappa_j\Big)$$

$$= \sum_{i=1}^{n}\sum_{j=1}^{m} \frac{dx_i}{Rx_i}\frac{dy_j}{Ry_j}\Big(\frac{\widehat{\lambda}}{Rx_i}\widetilde{H}_{i.} + \frac{\widehat{\lambda}}{Ry_j}\widetilde{H}_{.j}\Big) \quad \text{using(6.26), (6.27)} \tag{6.142}$$

$$= O_p\Big(\frac{\log(n)}{\sqrt{n}}\Big)\sum_{i=1}^{n}\sum_{j=1}^{m} H_{ij}\frac{dx_i}{Rx_i}\frac{dy_j}{Ry_j} \qquad \text{using (6.141)} \tag{6.143}$$

$$= O_p\Big(\frac{\log(n)}{\sqrt{n}}\Big) \quad \text{using (6.34), since } \overline{H}_{..} \text{ is uniformly bounded} \tag{6.144}$$

Similarly with reference to (6.33) of order $\widehat{\lambda}^2$,

$$\sum_{i=1}^{n}\sum_{j=1}^{m} H_{ij}\frac{dx_i}{Rx_i}\frac{dy_j}{Ry_j}\Big[\eta_i^2 + \eta_i\kappa_j + \kappa_j^2\Big]$$

$$= \sum_{i=1}^{n}\sum_{j=1}^{m} H_{ij}\frac{dx_i}{Rx_i}\frac{dy_j}{Ry_j}\Big[\Big(\frac{\widehat{\lambda}\,\widetilde{H}_{i.}}{Rx_i}\Big)^2 + \frac{\widehat{\lambda}}{Rx_i}\widetilde{H}_{i.}\frac{\widehat{\lambda}}{Ry_j}\widetilde{H}_{.j} + \Big(\frac{\widehat{\lambda}\,\widetilde{H}_{.j}}{Ry_j}\Big)^2\Big] \tag{6.145}$$

$$= O_p\Big(\frac{\log(n)}{\sqrt{n}}\Big)^2 \tag{6.146}$$

Finally consider the following:

$$\widetilde{H}_{i.} = \overline{H}_{i.} + (\widetilde{H}_{i.} - \overline{H}_{i.}) \tag{6.147}$$

$$= \overline{H}_{i.} + \sum_{j=1}^{m} H_{ij}\Big(\widehat{\nu_j} - \frac{dy_j}{Ry_j}\Big) \quad \text{substituting (6.12) and (6.36)} \tag{6.148}$$

$$\doteq \overline{H}_{i.} + \sum_{j=1}^{m} H_{ij}\Big[\frac{dy_j}{Ry_j}\Big(1 - \frac{\widehat{\lambda}}{Ry_j}\widetilde{H}_{.j}\Big) - \frac{dy_j}{Ry_j}\Big] \quad \text{from (6.22) to order } \widehat{\lambda} \tag{6.149}$$

$$= \overline{H}_{i.} - \sum_{j=1}^{m} H_{ij}\frac{dy_j}{Ry_j}\Big(\frac{\widehat{\lambda}}{Ry_j}\ \widetilde{H}_{.j}\Big) \quad \text{rearranging (6.149)} \tag{6.150}$$

$$= \overline{H}_{i.} - \sum_{j=1}^{m} H_{ij}\frac{dy_j}{Ry_j}\ O_p\Big(\frac{\log(n)}{\sqrt{n}}\Big) \quad \text{using (6.141)} \tag{6.151}$$

$$= \overline{H}_{i.} - O_p\Big(\frac{\log(n)}{\sqrt{n}}\Big)\sum_{j=1}^{m} H_{ij}\frac{dy_j}{Ry_j} \quad \text{rearranging (6.151)} \tag{6.152}$$

$$= \overline{H}_{i.} - O_p\Big(\frac{\log(n)}{\sqrt{n}}\Big)\overline{H}_{i.} \quad \text{from (6.36)} \tag{6.153}$$

$$\doteq \overline{H}_{i.} \tag{6.154}$$

Therefore from (6.144), (6.146), and (6.154) we conclude that the Taylor expansion in (6.35) is valid.

6.5 Lemmas

Lemma 1

$$\text{For fixed } t, \int g(t,s)d\widehat{\Lambda_2}(s) \xrightarrow{p} \int g(t,s)d\Lambda_2(s) \quad \text{(Refer to (6.94) - (6.96))} \tag{6.155}$$

Proof:

We will assume that $\int g(t,s)d\Lambda_2(s) < \infty$. We first apply the law of large numbers (LLN) to the case where the integral in (6.155) runs from 0 up to any large (but finite) positive number τ.

$$\int_0^\tau g(t,s)d\widehat{\Lambda_2}(s) = \sum_{j=1}^m I[y_j < \tau]\ g(t,y_j)\frac{dy_j}{Ry_j} \tag{6.156}$$

$$= \frac{1}{m}\sum_{j=1}^m I[y_j < \tau]\ dy_j\Big(\frac{g(t,y_j)}{Ry_j/m} - \frac{g(t,y_j)}{\big(1-F_2(y_j^-)\big)\big(1-G_2(y_j^-)\big)}\Big)$$

$$+ \frac{1}{m}\sum_{j=1}^m I[y_j < \tau]\frac{g(t,y_j)\ dy_j}{\big(1-F_2(y_j^-)\big)\big(1-G_2(y_j^-)\big)} \quad \text{add, subtract term} \tag{6.157}$$

The first term in (6.157) is bounded as follows:

$$\frac{1}{m}\sum_{j=1}^m I[y_j < \tau]\ dy_j\ \Big|\frac{g(t,y_j)}{Ry_j/m} - \frac{g(t,y_j)}{\big(1-F_2(y_j^-)\big)\big(1-G_2(y_j^-)\big)}\Big| \tag{6.158}$$

$$\le \sup_{y<\tau}\Big|\frac{g(t,y_j)}{Ry_j/m} - \frac{g(t,y_j)}{\big(1-F_2(y_j^-)\big)\big(1-G_2(y_j^-)\big)}\Big| \tag{6.159}$$

$$= \sup_{y<\tau}\Big|g(t,y_j)\Big|\ \Big|\frac{1}{Ry_j/m} - \frac{1}{\big(1-F_2(y_j^-)\big)\big(1-G_2(y_j^-)\big)}\Big|. \tag{6.160}$$

Now $\sup_{y<\tau}\Big|g(t,y_j)\Big|$ is bounded by assumption. And in Lemma 2 below we show that $Ry_j/m \xrightarrow{p} \big(1-F_2(y_j^-)\big)\big(1-G_2(y_j^-)\big)$, hence its reciprocal is at least uniformly convergent on the set $\{y \le \tau\}$. Hence the entire term in (6.160) is uniformly convergent to 0.

The last term in (6.157) is the average of an independent and identically distributed (iid) sum, so it converges to its expectation by the LLN. Now,

$$E\Big(I[y_j < \tau]\frac{g(t,y_j)\ dy_j}{\big(1-F_2(y_j^-)\big)\big(1-G_2(y_j^-)\big)}\Big) = \int_0^\tau g(t,s)d\Lambda_2(s) \tag{6.161}$$

Therefore, for any finite τ,

$$\int_0^\tau g(t,s)d\widehat{\Lambda}_2(s) \xrightarrow{p} \int_0^\tau g(t,s)d\Lambda_2(s) \tag{6.162}$$

Now we address the tail $[\tau,\infty)$ in (6.155).

Consider the ratio,

$$\sup_{y_j\in[0,\infty)} \Big| \frac{\big(1-F_2(y_j^-)\big)\big(1-G_2(y_j^-)\big)}{Ry_j/m} \Big| \tag{6.163}$$

This ratio in (6.163) is bounded in probability (per Zhou(1991)[22]). Therefore, except on a set of probability η, for any $1>\eta>0$, we have,

$$\Big| \frac{\big(1-F_2(y_j^-)\big)\big(1-G_2(y_j^-)\big)}{Ry_j/m} \Big) \Big| < C, \quad \text{for some positive constant } C \tag{6.164}$$

Therefore,

$$\begin{aligned} &\sum_{j=1}^m I[y_j<\tau]|g(t,y_j)|\ \frac{dy_j}{Ry_j} \\ &\le C\frac{1}{m}\sum_{j=1}^m I[y_j<\tau]\frac{|g(t,y_j)|\ dy_j}{\big(1-F_2(y_j^-)\big)\big(1-G_2(y_j^-)\big)} \end{aligned} \tag{6.165}$$

The RHS of (6.165) is an iid sum. Therefore by the strong law of large numbers it converges to its mean, which is,

$$C\int_\tau^\infty |g(t,s)|d\Lambda_2(s) \tag{6.166}$$

Since $\int |g(t,s)|d\Lambda_2(s) < \infty$ by assumption, therefore $\int_\tau^\infty |g(t,s)|d\Lambda_2(s)$ in (6.166) can be made arbitrarily small by selecting a large value of τ, such that $\int g(t,s)d\Lambda_2(s) < \frac{\epsilon}{C}$, where ϵ is an arbitrarily small positive number. Therefore,

$$C\int_\tau^\infty g(t,x)d\Lambda_2(s) < \epsilon \tag{6.167}$$

$$\text{Thus} \qquad \int_\tau^\infty g(t,s)d\widehat{\Lambda_2}(s) \xrightarrow{p} 0 \tag{6.168}$$

$$\text{hence} \qquad \int_0^\infty g(t,s)d\widehat{\Lambda_2}(s) \xrightarrow{p} \int_0^\infty g(t,s)d\Lambda_2(s) \tag{6.169}$$

<u>*Lemma 2*</u>

Suppose $M < \infty$ is such that $(1 - F_2(y_j^-))(1 - G_2(y_j^-)) > 0$.

Then for $y_j < M$ we have,

$$\frac{m}{Ry_j} \xrightarrow{p} \frac{1}{(1 - F_2(y_j^-))(1 - G_2(y_j^-))} \quad \Big(\text{ Refer to (6.120), (6.121) } \Big) \tag{6.170}$$

<u>Proof:</u>

$$\lim_{m\to\infty}\Big(\frac{Ry_j}{m}\Big) = \lim_{m\to\infty}\Big(\frac{\sum_{r=1}^m I[y_r \geq y_j]}{m}\Big) \quad \text{from (6.10)} \tag{6.171}$$

$$= \mathbb{E}\Big(I[y_r \geq y_j]\Big) \quad \text{Glivenko-Cantelli, law of large numbers} \tag{6.172}$$

$$= P(y_r \geq y_j) \tag{6.173}$$

$$= (1 - F_1(y_j^-))(1 - G_1(y_j^-)) \tag{6.174}$$

Equation (6.174) involves the variable Y as well as the right-censoring variable (which we may call R) associated with Y. We only observe the variable $T = \min(Y, R)$. In order for T to exceed a value y_j it is necessary that both Y and

R exceed y_j. Hence the probability that T exceeds y_j is the product of the probability that Y exceeds y_j and the probability that R exceeds y_j, where a product applies since Y and R are assumed to be independent.

Then,

$$\lim_{m\to\infty}\left(\frac{m}{Ry_j}\right) = \lim_{m\to\infty}\left(\frac{1}{Ry_j/m}\right) \tag{6.175}$$

$$= \frac{1}{\lim_{m\to\infty}\left(Ry_j/m\right)} \quad \text{since } s(t) = 1/t \text{ is continuous for } t \neq 0 \tag{6.176}$$

$$= \frac{1}{(1-F_2^-(y_j))(1-G_2^-(y_j))} \quad \text{from (6.174)} \tag{6.177}$$

Similarly,

$$\lim_{n\to\infty}\left(\frac{n}{Rx_i}\right) = \frac{1}{(1-F_1(x_i^-))(1-G_1(x_i^-))} \tag{6.178}$$

Chapter 7 Non-Negative Probabilities

In this chapter we prove that the probabilities $\widehat{\boldsymbol{\mu}}$ and $\widehat{\boldsymbol{\nu}}$ obtained in Chapters 3 and 4 are non-negative for uncensored data, provided that θ is chosen in the "feasible" range as in (7.5) below. (There is no concern about the censored data since their weights are shifted to the uncensored data in the EM algorithm.) The proof we offer is similar to the proof in Owen (2001) [12], p. 22.

Recall equations (3.16) - (3.18) in chapter 3 of this dissertation:

$$\widehat{\mu}_i = \frac{wx_i}{\sum_{i=1}^{n} wx_i + \lambda \sum_{j=1}^{m} \left(\, g(x_i, y_j) - \theta \, \right) \, \widehat{\nu}_j} \tag{7.1}$$

$$\widehat{\nu}_j = \frac{wy_j}{\sum_{j=1}^{m} wy_j + \lambda \sum_{i=1}^{n} \left(\, g(x_i, y_j) - \theta \, \right) \, \widehat{\mu}_i} \tag{7.2}$$

$$\sum_{i=1}^{n} \sum_{j=1}^{m} \left(\, g(x_i, y_j) - \theta \, \right) \, \widehat{\mu}_i \, \widehat{\nu}_j = 0 \tag{7.3}$$

Let us denote $g(x_i, y_j)$ as g_{ij} for simplicity. Then we can rewrite (7.3) above as,

$$\sum_{i=1}^{n} \sum_{j=1}^{m} g_{ij} \, \widehat{\mu}_i \, \widehat{\nu}_j = \theta. \tag{7.4}$$

Equation (7.4) says that θ is an average of the values of $g_{i,j}$, where $i = 1, \ldots, n$ and $j = 1, \ldots, m$. Let us denote the lowest value of $g_{i,j}$ as $g_{(1)}$ and let us denote

the highest value of $g_{i,j}$ as $g_{(N)}$, where $N = n + m$. Since an average value must lie between the lowest and highest values, then a "feasible" value for θ must be such that,

$$g_{(1)} \leq \theta \leq g_{(N)}. \tag{7.5}$$

We shall assume in the analysis below that θ has been chosen in the feasible range.

Now (7.1) is monotone in λ (which can easily be seen by taking its first derivative with respect to λ). Therefore from (7.5) the limits of the allowable range for λ in (7.1) can be found by successively putting $\Big(g_{ij} = g_{(1)},\ \widehat{\mu_i} = 1,\ \widehat{\nu_j} = 1\Big)$ and $\Big(g_{ij} = g_{(N)},\ \widehat{\mu_i} = 1,\ \widehat{\nu_j} = 1\Big)$ into (7.1).

Putting $\Big(g_{ij} = g_{(1)},\ \widehat{\mu_i} = 1,\ \widehat{\nu_j} = 1\Big)$ into (7.1) gives,

$$1 = \frac{wx_{(1)}}{\sum_{i=1}^{n} wx_i + \lambda_H\big(\ g_{(1)} - \theta\ \big)} \tag{7.6}$$

which implies,

$$\lambda_H = \frac{wx_{(1)} - \sum_{i=1}^{n} wx_i}{\Big(g_{(1)} - \theta\Big)} \quad \text{for } \theta \neq g_{(1)} \tag{7.7}$$

where λ_H is the high end of the allowable range for λ and $wx_{(1)}$ is the weight corresponding to $g_{(1)}$. From (7.7) we see that λ_H is positive.

Similarly putting $\Big(g_{ij} = g_{(N)},\ \widehat{\mu_i} = 1,\ \widehat{\nu_j} = 1\Big)$ into (7.1) gives,

$$1 = \frac{wx_{(N)}}{\sum_{i=1}^{n} wx_i + \lambda_L\big(\ g_{(N)} - \theta\ \big)} \tag{7.8}$$

which implies,

$$\lambda_L = \frac{wx_{(N)} - \sum_{i=1}^{n} wx_i}{\Big(g_{(N)} - \theta\Big)} \quad \text{for } \theta \neq g_{(N)} \tag{7.9}$$

where λ_L is the low end of the allowable range for λ and $wx_{(N)}$ is the weight corresponding to $g_{(N)}$. From (7.9) we see that λ_L is negative.

Combining (7.7) and (7.9) gives the allowable range of λ as,

$$\frac{wx_{(N)} - \sum_{i=1}^{n} wx_i}{\Big(g_{(N)} - \theta\Big)} \leq \lambda \leq \frac{wx_{(1)} - \sum_{i=1}^{n} wx_i}{\Big(g_{(1)} - \theta\Big)}. \tag{7.10}$$

Since (7.1) is monotone in λ we can find the allowable range of $\widehat{\mu_i}$ by successively putting the LHS and the RHS of (7.10) into (7.1). Putting the LHS of (7.10) into (7.1) gives,

$$\widehat{\mu_i}(\lambda_L) = \frac{wx_i}{\sum_{i=1}^{n} wx_i + \big(wx_{(N)} - \sum_{i=1}^{n} wx_i\big)\frac{\sum_{j=1}^{m}\big(\ g_{ij} - \theta\ \big)\ \widehat{\nu_j}}{g_{(N)} - \theta}}. \tag{7.11}$$

In (7.11) if $\sum_{j=1}^{m}\big(\ g_{ij} - \theta\ \big)\ \widehat{\nu_j} \leq 0$, let $\frac{\sum_{j=1}^{m}\big(\ g_{ij} - \theta\ \big)\ \widehat{\nu_j}}{g_{(N)} - \theta} = -\xi$, where $-\xi \leq 0$. Then (7.11) can be rewritten as,

$$\widehat{\mu}_i(\lambda_L) = \frac{wx_i}{\sum_{i=1}^{n} wx_i + \big(wx_{(N)} - \sum_{i=1}^{n} wx_i\big)(-\xi)} \tag{7.12}$$

which is non-negative.

In (7.11) if $\sum_{j=1}^{m} \big(\ g_{ij} - mu \ \big) \ \widehat{\nu}_j > 0$, let $\frac{\sum_{j=1}^{m} \big(\ g_{ij} - \theta \ \big) \ \widehat{\nu}_j}{g_{(N)} - \theta} = \zeta$, where $1 \geq \zeta > 0$. Then (7.11) can be rewritten as,

$$\widehat{\mu}_i(\lambda_L) = \frac{wx_i}{(1-\zeta)\sum_{i=1}^{n} wx_i + \zeta \ wx_{(N)}} \tag{7.13}$$

which is non-negative.

Combining (7.12) and (7.13) gives,

$$\frac{wx_i}{(1+\xi)\sum_{i=1}^{n} wx_i - \xi wx_{(N)}} \leq \widehat{\mu}_i(\lambda_L) \leq \frac{wx_i}{(1-\zeta)\sum_{i=1}^{n} wx_i + \zeta \ wx_{(N)}} \tag{7.14}$$

where the LHS and RHS of (7.14) are both positive. In addition, the LHS of (7.14) is easily shown to be less than the RHS of (7.14).

Similarly, putting the RHS of (7.10) into (7.1) gives,

$$\widehat{\mu}_i(\lambda_H) = \frac{wx_i}{\sum_{i=1}^{n} wx_i + \big(wx_{(1)} - \sum_{i=1}^{n} wx_i\big) \frac{\sum_{j=1}^{m} \big(\ g_{ij} - \theta \ \big) \ \widehat{\nu}_j}{g_{(1)} - \theta}} \tag{7.15}$$

In (7.15) if $\sum_{j=1}^{m} \left(g_{ij} - \theta \right) \widehat{\nu_j} < 0$, let $\frac{\sum_{j=1}^{m} \left(g_{ij} - \theta \right) \widehat{\nu_j}}{g_{(1)} - \theta} = \epsilon$, where $1 \geq \epsilon > 0$. Then (7.15) can be rewritten as,

$$\widehat{\mu_i}(\lambda_H) = \frac{wx_i}{(1-\epsilon)\sum_{i=1}^{n} wx_i + \epsilon\ wx_{(1)}} \tag{7.16}$$

which is non-negative.

In (7.15) if $\sum_{j=1}^{m} \left(g_{ij} - \theta \right) \widehat{\nu_j} \geq 0$, let $\frac{\sum_{j=1}^{m} \left(g_{ij} - \theta \right) \widehat{\nu_j}}{g_{(1)} - \theta} = -\delta$, where $-\delta \leq 0$. Then (7.15) can be rewritten as,

$$\widehat{\mu_i}(\lambda_H) = \frac{wx_i}{(1+\delta)\sum_{i=1}^{n} wx_i - \delta wx_{(1)}} \tag{7.17}$$

which is non-negative.

Combining (7.16) and (7.17) gives,

$$\frac{wx_i}{(1+\delta)\sum_{i=1}^{n} wx_i - \delta wx_{(1)}} \leq \widehat{\mu_i}(\lambda_H) \leq \frac{wx_i}{(1-\epsilon)\sum_{i=1}^{n} wx_i + \epsilon\ wx_{(1)}} \tag{7.18}$$

where the LHS and RHS of (7.18) are both positive. In addition, the LHS of (7.18) is easily shown to be less than the RHS of (7.18).

Therefore from (7.14) $\widehat{\mu_i}(\lambda_L) \geq 0$ and from (7.18) $\widehat{\mu_i}(\lambda_H) \geq 0$, for $\{i : dx_i = 1\}$. Since $\widehat{\mu_i}$ is monotone in λ per (7.1), therefore $\widehat{\mu_i}(\lambda)$ is non-negative over the allowable

range of λ, which is $\{\lambda : \lambda_L \leq \lambda \leq \lambda_H\}$.

By an analogous argument, $\widehat{\nu}_j(\lambda)$ for $j : dy_j = 1$ is likewise non-negative over the allowable range of λ.

Therefore we conclude that the probabilities $\widehat{\boldsymbol{\mu}}$ and $\widehat{\boldsymbol{\nu}}$ obtained in Chapters 3 and 4 are non-negative for uncensored data, when θ is chosen in the feasible range as in (7.5).

Chapter 8 Future Work

Five ways that this dissertation could be extended in the future are as follows:

1. Prove that $-2LLR \xrightarrow{d} \chi^2_{(1)}$ for two right-censored, left-censored, and doubly-censored samples with a mean-type hypothesis.

2. Prove that $-2LLR \xrightarrow{d} \chi^2_{(1)}$ for two right-censored, left-censored, and doubly-censored samples with multiple mean-type hypotheses.

3. Write an algorithm, similar to that in chapter 3, for the case of two interval-censored samples and a single mean-type hypothesis.

4. Write an algorithm, similar to that in chapter 3, for the case of two truncated samples and a single mean-type hypothesis.

5. Rewrite the loops in the R-programs using the C-language, for greater speed.

Appendix A: Program Code

This appendix is divided into two sections. The first section provides documentation for the eight functions which comprise the emplik2 R-package. The second section provides a listing of the code for the eight functions.

The names of the eight functions are as follows:

el2.cen.EMs

el2.cen.EMm

el2.test.wts

el2.test.wtm

myWKM

myWCY

myWdataclean2

myWdataclean3

Much of the code for the eight functions has been adapted, with permission, from the *emplik* R-package written by my advisor Dr. Mai Zhou.

A.1 Documentation for the Functions

el2.cen.EMs

Computes p-value for a single mean-type hypothesis, based on two independent samples that may contain censored data.

Description: This function uses the EM algorithm to calculate a maximized empirical-log-likelihood ratio for the hypothesis

$$H_o : E(g(x, y) - \theta) = 0$$

where E indicates expected value; $g(x, y)$ is a user-defined function of the two samples $\boldsymbol{x}$ and $\boldsymbol{y}$; and θ is the hypothesized value of $E(g(x, y))$. The samples $\boldsymbol{x}$ and $\boldsymbol{y}$ are assumed independent. They may be uncensored, right-censored, left-censored, or left-and-right ("doubly") censored. A p-value for H_o is also calculated, based on the assumption that -2log-likelihood-ratio is approximately distributed as $\chi^2_{(1)}$.

Usage: The function is called as follows:

el2.cen.EMs(x,dx,y,dy,fun=function(x,y){x>=y}, mean=0.5)

Arguments:

x	vector of data for the first sample
dx	vector of censoring indicators for x
y	vector of data for the second sample
dy	vector of censoring indicators for y

fun	user-defined, continuous-weight function
$g(x, y)$	used to define the mean in H_o, default is $x >= y$
$mean$	hypothesized value of $\theta = E(g(x, y))$, default is 0.5

Details: The value of *mean* should be chosen between the maximum and minimum values of $g(x_i, y_j)$; otherwise there may be no distributions for $\boldsymbol{x}$ and $\boldsymbol{y}$ that will satisfy H_o. If *mean* is inside this interval, but the convergence is still not satisfactory, then the value of *mean* should be moved closer to the NPMLE for $E(g(x, y))$. (The NPMLE itself should always be a feasible value for *mean*.)

Value: $el2.cen.EMs$ returns a list of values as follows:

$xd1$	a vector of the unique, uncensored x-values in ascending order
$yd1$	a vector of the unique, uncensored y-values in ascending order
$temp3$	a list of values returned by the $el2.test.wts$ function
$mean$	the hypothesized value of $\theta = E(g(x, y))$
$funNPMLE$	the non-parametric-maximum-likelihood-estimator of θ
$logel00$	the log of the unconstrained empirical likelihood
$logel$	the log of the constrained empirical likelihood
"$-2LLR$"	$-2(logel - logel00)$
$Pval$	the estimated p-value for H_o
$logvec$	vector of successive values of $logel$ computed by EM algorithm
sum_muvec	sum of probability jumps for the uncensored x-values
sum_nuvec	sum of probability jumps for the uncensored y-values
$constraint$	realized value of constraint, should be close to 0

el2.cen.EMm

Computes p-value for multiple mean-type hypotheses, based on two independent samples that may contain censored data.

Description: This function uses the EM algorithm to calculate a maximized empirical likelihood ratio for a set of p hypotheses as follows:

$$H_o : E(g(x, y) - \theta) = 0$$

where E indicates expected value; $g(x, y)$ is a vector of user-defined functions $g_1(x, y), \ldots, g_p(x, y)$; and θ is a vector of p hypothesized values of $E(g(x, y))$. The two samples $\boldsymbol{x}$ and $\boldsymbol{y}$ are assumed independent. They may be uncensored, right-censored, left-censored, or left-and-right ("doubly") censored. A p-value for H_o is also calculated, based on the assumption that -2log-likelihood-ratio is asymptotically distributed as $\chi^2_{(p)}$.

Usage: The function is called as follows:

el2.cen.EMm(x, dx, y, dy, p, H, mean)

Arguments:

x	vector of data for the first sample
dx	vector of censoring indicators for x
y	vector of data for the second sample

dy	vector of censoring indicators for y
p	the number of hypotheses
H	matrix defined as $H = [H_1, H_2, \ldots, H_p]$
H_k	equals $[g_k(x_i, y_j) - mu_k]_{nxm},\ k = 1, \ldots, p$
$mean$	vector, hypothesized value of $\theta = E(g(x, y)$

Details: The value of $mean_k$ should be chosen between the maximum and minimum values of $g_k(x_i, y_j)$; otherwise there may be no distributions for $\boldsymbol{x}$ and $\boldsymbol{y}$ that will satisfy H_o. If $mean_k$ is inside this interval, but the convergence is still not satisfactory, then the value of $mean_k$ should be moved closer to the NPMLE for $E(g_k(x, y))$. (The NPMLE itself should always be a feasible value for $mean_k$.)

Value: $el2.cen.EMm$ returns a list of values as follows:

$xd1$	vector of unique, uncensored x-values in ascending order
$yd1$	vector of unique, uncensored y-values in ascending order
$temp3$	list of values returned by $el2.test.wtm$ function
$mean$	hypothesized value of $\theta = E(g(x, y))$
$NPMLE$	non-parametric-maximum-likelihood-estimator vector of θ
$logel00$	log of the unconstrained empirical likelihood
$logel$	log of the constrained empirical likelihood
"$-2LLR$"	$-2(logel - logel00)$
$Pval$	p-value for the p simultaneous hypotheses
$logvec$	vector of successive values of $logel$ computed by EM algorithm
sum_muvec	sum of probability jumps for the uncensored x-values
sum_nuvec	sum of probability jumps for the uncensored y-values

el2.test.wts

Computes maximum-likelihood probability jumps for a single mean-type hypothesis, based on two independent uncensored samples.

Description: This function computes the maximum-likelihood probability jumps for a single mean-type hypothesis, based on two samples that are independent, uncensored, and weighted. The target function for the maximization is the constrained log-likelihood which can be expressed as,

$$\sum_{dx_i=1} wx_i \log \mu_i + \sum_{dy_j=1} wy_j \log \nu_j - \eta(1 - \sum_{dx_i=1} \mu_i) - \delta(1 - \sum_{dy_j=1} \nu_j) \\ -\lambda \sum_{dx_i=1} \sum_{dy_j=1} (g(x_i, y_j) - \theta)\mu_i \nu_j$$

where the variables are defined as follows:

$\boldsymbol{x}$ is a vector of data for the first sample

$\boldsymbol{y}$ is a vector of data for the second sample

$\boldsymbol{wx}$ is a vector of weights for the first sample

$\boldsymbol{wy}$ is a vector of weights for the second sample

$\boldsymbol{\mu}$ is a vector of probability jumps for the first sample

$\boldsymbol{\nu}$ is a vector of probability jumps for the second sample

λ is a Lagrangian multiplier

θ is the hypothesized value of $E(g(u, v))$

Usage: The function is called as follows:

el2.test.wts(u,v,wu,wv,mu0,nu0,indicmat,mean)

Arguments:

u	vector of uncensored data for first sample
v	vector of uncensored data for second sample
wu	vector of weights for u
wv	vector of weights for v
$mu0$	vector of probability jumps for u
$nu0$	vector of probability jumps for v
$indicmat$	a matrix $[g(u_i, v_j) - mean]$ where $g(u, v)$ is a user-chosen function
$mean$	is the hypothesized value of $\theta = E(g(u, v))$

Details: This function is called by $el2.cen.EMs$ and would not typically be accessed by the user. It is listed here for reference.

The value of $mean$ should be chosen between the maximum and minimum values of $g(u_i, v_j)$; otherwise there may be no distributions for $\boldsymbol{u}$ and $\boldsymbol{v}$ that will satisfy the the mean-type hypothesis. If $mean$ is inside this interval, but the convergence is still not satisfactory, then the value of $mean$ should be moved closer to the NPMLE for $E(g(u, v))$. (The NPMLE itself should always be a feasible value for $mean$.)

Value: $el2.test.wts$ returns a list of values as follows:

u	vector of uncensored data for the first sample
wu	vector of weights for u
$jumpu$	vector of probability jumps for u
v	vector of uncensored data for the second sample
wv	vector of weights for v

$jumpv$	vector of probability jumps for v
lam	value of the Lagrangian multipler found by the calculations

el2.test.wtm

Computes maximum-likelihood probability jumps for multiple mean-type hypotheses, based on two independent uncensored samples.

Description: This function computes the maximum-likelihood probability jumps for multiple mean-type hypotheses, based on two samples that are independent, uncensored, and weighted. The target function for the maximization is the constrained log(empirical likelihood) which can be expressed as,

$$\sum_{dx_i=1} wx_i \log \mu_i + \sum_{dy_j=1} wy_j \log \nu_j - \eta(1 - \sum_{dx_i=1} \mu_i) - \delta(1 - \sum_{dy_j=1} \nu_j) \\ - (\mu^T H_1 \nu, \ldots, \mu^T H_p \nu)\ \boldsymbol{\lambda}$$

where the variables are defined as follows:

$\boldsymbol{x}$ is a vector of data for the first sample

$\boldsymbol{y}$ is a vector of data for the second sample

$\boldsymbol{wx}$ is a vector of weights for the first sample

$\boldsymbol{wy}$ is a vector of weights for the second sample

$\boldsymbol{\mu}$ is a vector of probability jumps for the first sample

$\boldsymbol{\nu}$ is a vector of probability jumps for the second sample

H_k = a matrix $[g_k(x_i, y_j) - mean_k]$, where $k = 1, \ldots, p$

$g_k(x, y)$ is a user-chosen function

mean is a vector of length p of hypothesized means

$\boldsymbol{\lambda}$ is a vector of Lagrangian multipliers

Usage: The function is called as follows:

el2.test.wtm(xd1,yd1,wxd1new, wyd1new, muvec, nuvec, Hu, Hmu, Hnu, p, mean)

Arguments:

$xd1$	vector of uncensored data for first sample
yd	vector of uncensored data for second sample
$wxd1new$	vector of weights for $xd1$
$wyd1new$	vector of weights for $yd1$
$muvec$	vector of probability jumps for $xd1$
$nuvec$	vector of probability jumps for $yd1$
Hu	equals $[H_1 - [mean_1], \ldots, H_p - [mean_p]], dx_i = 1, dy_j = 1$
Hmu	a matrix (see code listing for its calculation)
Hnu	a matrix (see code listing for its calculation)
p	the number of hypotheses
$mean$	vector of hypothesized values of $E(g_k(u,v)), k = 1, \ldots, p$

Details: This function is called by $el2.cen.EMm$ and is not intended to be accessed by the user. It is listed here for reference.

The value of $mean_k$ should be chosen between the maximum and minimum values of $g_k(xd1_i, yd1_j)$; otherwise there may be no distributions for $\boldsymbol{xd1}$ and $\boldsymbol{yd1}$ that will satisfy the the mean-type hypothesis. If $mean_k$ is inside this interval, but the convergence is still not satisfactory, then the value of $mean_k$ should be moved closer to the NPMLE for $E(g(xd1, yd1))$. (The NPMLE itself should always be a feasible value for $mean_k$.)

Value: $el2.test.wtm$ returns a list of values as follows:

$constmat$	a matrix whose kth row is $\mu^T H_k \nu, k = 1, \ldots, p$
lam	vector of Lagrangian mulipliers
$muvec1$	vector of probability jumps for $xd1$
$nuvec1$	vector of probability jumps for $yd1$
$mean$	vector of hypothesized means

myWCY, myWKM, myWdataclean2, myWdataclean3

These four functions are not intended to be accessed by the user. They are listed here with brief descriptions for reference.

Description:

myWCY calculates the weighted Chang-Yang self-consistent estimator for doubly-censored data. It is called by *el2.cen.EMs* and *el2.cen.EMm*.

myWKM calculates the weighted Kaplan-Meier estimator for right-censored data. It is called by *el2.cen.EMs* and *el2.cen.EMm*.

myWdataclean2 sorts the data, collapses the true ties, and puts the number of tied values as the weights. It is called by *el2.cen.EMs*.

myWdataclean3 sorts the data, collapses the true ties, and puts the number of tied values as the weights. It is called by *myWCY* and *myWKM*.

Usage: The four functions are called as follows:

myWCY(x, d, zc = rep(1, length(d)), wt = rep(1, length(d)))

myWKM(x, d, zc = rep(1, length(d)), w = rep(1, length(d)))

myWdataclean2(z, d, wt = rep(1, length(z)))

myWdataclean3(z, d, zc = rep(1, length(z)), wt = rep(1, length(z)))

A.2 Annotated Listing of Function Code

Annotated code for el2.cen.EMs

```
el2.cen.EMs<-function(x,dx,y,dy,fun=function(x,y) {x>=y}, mean=0.5,
  maxit=25, ...){

#x,y pairs can be any combination of uncensored, left-cens,
#right-cens. Data can be discrete or continuous. Note that x>y is
#not the same as x>=y for discrete data

#Store data x,y as vectors.
    xvec <- as.vector(x)
    yvec <- as.vector(y)

#Store length of xvec,yvec.
    nx <- length(xvec)
    ny <- length(yvec)

#Check that there are at least 2 data in x,y.
    if (nx <= 1)
        stop("need more observations in x")
    if (ny <= 1)
        stop("need more observations in y")

#Check that status and observations have same length in x,y.
    if (length(dx) != nx)
        stop("length of x and dx must agree")
```

```
    if (length(dy) != ny)
        stop("length of y and dy must agree")

#Check that status are only 0,1,2.
    if (any((dx != 0) & (dx != 1) & (dx != 2)))
        stop("dx must be 0(right-censored) or 1(uncensored) or 2(left-
         censored")
    if (any((dy != 0) & (dy != 1) & (dy != 2)))
        stop("dy must be 0(right-censored) or 1(uncensored) or 2(left-
         censored")

#Check that xvec,yvec are numeric (for example, no NA values).
    if (!is.numeric(xvec))
        stop("x must be numeric")
    if (!is.numeric(yvec))
        stop("y must be numeric")

#Check that mean has dimension 1.
    if (length(mean) != 1)
        stop("mean must have dimension 1")

#"Clean" data using fcn myWdataclean2="weighted dataclean 2"
    temp1x <- myWdataclean2(xvec, dx)
    temp1y <- myWdataclean2(yvec, dy)

#Redefine x,y as ascending, distinct data values.
    x <- temp1x$value
```

```
    y <- temp1y$value
#Redefine status dx,dy corresponding to redefined x,y.
    dx <- temp1x$dd
    dy <- temp1y$dd
#Define weights wx,wy as the number of data at each distinct value.
    wx <- temp1x$weight
    wy <- temp1y$weight
#Set highest of all status 1,0 to 1.
    xindex10 <- which(dx != 2)
    yindex10 <- which(dy != 2)
    dx[xindex10[length(xindex10)]] <- 1
    dy[yindex10[length(yindex10)]] <- 1
#Set lowest of all status 1,2 to 1.
    xindex12 <- which(dx != 0)
    yindex12 <- which(dy != 0)
    dx[xindex12[1]] <- 1
    dy[yindex12[1]] <- 1
#Put censored,uncensored data into respective vectors.
    xd0  <-  x[dx == 0]
    wxd0 <- wx[dx == 0]
    xd1  <-  x[dx == 1]
    wxd1 <- wx[dx == 1]
    xd2  <-  x[dx == 2]
    wxd2 <- wx[dx == 2]
    yd0  <-  y[dy == 0]
    wyd0 <- wy[dy == 0]
    yd1  <-  y[dy == 1]
```

```
    wyd1 <- wy[dy == 1]
    yd2  <-  y[dy == 2]
    wyd2 <- wy[dy == 2]
#Check that there are at least 2 uncensored data for x,y.
    if (length(xd1) <= 1)
        stop("need more distinct uncensored x obs.")
    if (length(yd1) <= 1)
        stop("need more distinct uncensored y obs.")
#Store vector lengths.
    nx0 <- length(xd0)
    ny0 <- length(yd0)
    nx1 <- length(xd1)
    ny1 <- length(yd1)
    nx2 <- length(xd2)
    ny2 <- length(yd2)
    nx  <- length(x)
    ny  <- length(y)

#LR denominator (unconstrained):
   #Doubly-censored case.
    if ( (nx0>0) & (nx2>0 ) ) {
      temp2x   <- myWCY(x = x, d = dx, wt = wx)
      logelx00 <- temp2x$logEL
      jumpxu   <- temp2x$jump
   #Adjust xd1, nx1 etc. to reflect repeat data with different
   #censoring.
      xd1<-temp2x$xd1
```

```
   nx1<-length(xd1)
   wxd1<-temp2x$wd1
   xd0<-temp2x$xd0
   nx0<-length(xd0)
   wxd0<-temp2x$wd0
   }
 if ( (ny0>0) & (ny2>0 ) ) {
   temp2y   <- myWCY(x = y, d = dy, wt = wy)
   logely00 <- temp2y$logEL
   jumpyu   <- temp2y$jump
#Adjust yd1, ny1 etc. to reflect repeat data with different
#censoring.
   yd1<-temp2y$xd1
   ny1<-length(yd1)
   wyd1<-temp2y$wd1
   yd0<-temp2y$xd0
   ny0<-length(yd0)
   wyd0<-temp2y$wd0
   }
#Right-censored case.
 if ( (nx0>0) & (nx2==0 ) ) {
   temp2x   <- myWKM(x = x, d = dx, w = wx)
   logelx00 <- temp2x$logel
   jumpxu   <- temp2x$jump[temp2x$jump>0]
#Adjust xd1, nx1 etc. to reflect repeat data with different
#censoring.
   xd1<-temp2x$times[temp2x$jump>0]
```

```
   nx1<-length(xd1)
   wxd1<-temp2x$weight[temp2x$jump>0]
   xd0<-temp2x$times[temp2x$jump==0]
   nx0<-length(xd0)
   wxd0<-temp2x$weight[temp2x$jump==0]
   }
 if ( (ny0>0) & (ny2==0 ) ) {
   temp2y   <- myWKM(x = y, d = dy, w = wy)
   logely00 <- temp2y$logel
   jumpyu   <- temp2y$jump[temp2y$jump>0]
#Adjust yd1, ny1 etc. to reflect repeat data with different
#censoring.
   yd1<-temp2y$times[temp2y$jump>0]
   ny1<-length(yd1)
   wyd1<-temp2y$weight[temp2y$jump>0]
   yd0<-temp2y$times[temp2y$jump==0]
   ny0<-length(yd0)
   wyd0<-temp2y$weight[temp2y$jump==0]
   }
#Left-censored case.
 if ( (nx0==0) & (nx2>0 ) ) {
   dlx <- dx
   dlx [dlx == 2] <- 0
   temp2x   <- myWKM(x = x, d = rev(dlx), w = rev(wx))
   logelx00 <- temp2x$logel
   jumpxu   <- rev(temp2x$jump[temp2x$jump>0])
#Adjust xd1, nx1 etc. to reflect repeat data with different
```

```
# censoring.
    xd1<-temp2x$times[rev(temp2x$jump)>0]
    nx1<-length(xd1)
    wxd1<-rev(temp2x$weight)[rev(temp2x$jump)>0]
    xd2<-temp2x$times[rev(temp2x$jump)==0]
    nx2<-length(xd2)
    wxd2<-rev(temp2x$weight)[rev(temp2x$jump)==0]
    }
  if ( (ny0==0) & (ny2>0 ) ) {
    dly <- dy
    dly[dly == 2] <- 0
    temp2y   <- myWKM(x = y, d = rev(dly), w = rev(wy))
    logely00 <- temp2y$logel
    jumpyu   <- rev(temp2y$jump[temp2y$jump>0])
 #Adjust yd1, ny1 etc. to reflect repeat data with different
 #censoring.
    yd1<-temp2y$times[rev(temp2y$jump)>0]
    ny1<-length(yd1)
    wyd1<-rev(temp2y$weight)[rev(temp2y$jump)>0]
    yd2<-temp2y$times[rev(temp2y$jump)==0]
    ny2<-length(yd2)
    wyd2<-rev(temp2y$weight)[rev(temp2y$jump)==0]
    }
 #Uncensored case.
  if ( (nx0==0) & (nx2==0 ) ) {
    logelx00 <- sum(wxd1 * log(wxd1/sum(wxd1)))
    jumpxu   <- wxd1/sum(wxd1)
```

```
        }
    if ( (ny0==0) & (ny2==0 ) ) {
      logely00 <- sum(wyd1 * log(wyd1/sum(wyd1)))
      jumpyu   <- wyd1/sum(wyd1)
        }
  #Calculate likelihood.
      logel00 <- logelx00 + logely00
  #Calculate NPMLE.
      indic <- matrix(NA,nrow=nx1,ncol=ny1)
      for (i in 1:nx1) {
       for (j in 1:ny1) {
         indic[i,j] <- fun(xd1[i],yd1[j]) } }
      indicmat <- indic - mean
      funNPMLE=as.vector(jumpxu %*% indic %*% jumpyu)

#LR numerator (constrained):
  #Initialize muvec,nuvec.
    muvec <- jumpxu
    nuvec <- jumpyu
  #Initialize kx,ky and kkx,kky.
    #For each rt-cens x,y value, kx,ky holds xd1,yd1 index of first
    #uncensored value to its right.
    #For each left-censored value, kkx,kky holds xd1,yd1 index of first
    #uncensored value to its left.
     if (nx0>0) {
       kx <- rep(NA, nx0)
        for (i in 1:nx0) {kx[i] <- 1 + nx1 - sum(xd1 > xd0[i])}
```

```
    }
   if (nx2>0) {
     kkx <- rep(NA, nx2)
     for (i in 1:nx2) {kkx[i] <- sum(xd1 < xd2[i])}
    }
   if (ny0>0) {
     ky <- rep(NA, ny0)
      for (j in 1:ny0) {ky[j] <- 1 + ny1 - sum(yd1 > yd0[j])}
    }
   if (ny2>0) {
     kky <- rep(NA, ny2)
     for (j in 1:ny2) {kky[j] <- sum(yd1 < yd2[j])}
    }
#Initialize iteration counter num and log-likelihood-holder
#logvec.
   num <- 1
   logvec <- rep(0,maxit)
#Repeat EM repeatedly for maxit iterations.
     while (num <= maxit) {
#Initialize weights wxd1new,wyd1new.
      wxd1new <- wxd1
      wyd1new <- wyd1
#Perform Expectation step on uncensored x,y weights.
      if (nx0>0) {
        surx <- rev(cumsum(rev(muvec)))
        for (i in 1:nx0) {wxd1new[kx[i]:nx1] <- wxd1new[kx[i]:nx1] +
          wxd0[i] * muvec[kx[i]:nx1]/surx[kx[i]] }
```

```
        }
      if (nx2>0) {
        cdfx <- cumsum(muvec)
        for (i in 1:nx2) {wxd1new[1:kkx[i]] <- wxd1new[1:kkx[i]] +
          wxd2[i] * muvec[1:kkx[i]]/cdfx[kkx[i]] }
        }
      if (ny0>0) {
        sury <- rev(cumsum(rev(nuvec)))
        for (j in 1:ny0) {wyd1new[ky[j]:ny1] <- wyd1new[ky[j]:ny1] +
          wyd0[j] * nuvec[ky[j]:ny1]/sury[ky[j]] }
         }
      if (ny2>0) {
        cdfy <- cumsum(nuvec)
        for (j in 1:ny2) {wyd1new[1:kky[j]] <- wyd1new[1:kky[j]] +
          wyd2[j] * nuvec[1:kky[j]]/cdfy[kky[j]] }
         }
#Perform Maximization step on uncensored x,y jumps.
      temp3 <- el2.test.wts(xd1, yd1, wxd1new, wyd1new, muvec,
        nuvec, indicmat, mean)
      muvec <- temp3$jumpu
      nuvec <- temp3$jumpv
 #Calculate loglikelihood so its convergence can be tracked.
      logelx <- sum(wxd1 * log(muvec))
       if (nx0>0) {
        surx <- rev(cumsum(rev(muvec)))
        logelx <- logelx + sum(wxd0 * log(surx[kx]))
       }
```

```
      if (nx2>0) {
       cdfx <- cumsum(muvec)
       logelx <- logelx + sum(wxd2 * log(cdfx[kkx]))
      }
    logely <- sum(wyd1 * log(nuvec))
      if (ny0>0) {
       sury <- rev(cumsum(rev(nuvec)))
       logely <- logely + sum(wyd0 * log(sury[ky]))
      }
      if (ny2>0) {
       cdfy <- cumsum(nuvec)
       logely <- logely + sum(wyd2 * log(cdfy[kky]))
      }
    logel <- logelx + logely
    logvec[num]<-logel
    num <- num + 1
    }
 #Store -2log(likelihood ratio) in tval.
    tval <- 2 * (logel00 - logel)
 #Calculate constraint.
    constraint <- as.vector(muvec %*% indicmat %*% t(nuvec))
#Return results of the el2.cen.EM function.
    list(xd1=xd1,yd1=yd1,temp3=temp3, mean=mean, funNPMLE=
     funNPMLE, logel00=logel00, logel=logel, "-2LLR"=tval,
     Pval=1 - pchisq(tval, df = 1), logvec=logvec,
     sum_muvec=sum(muvec), sum_nuvec=sum(nuvec),
     constraint=constraint)}
```

Annotated code for el2.cen.EMm

```
el2.cen.EMm<-function(x, dx, y, dy, p, H, xc=1:length(x),
  yc=1:length(y), mean, maxit=25, ...) {
#x,y pairs can be any combination of uncensored, left-cens,
#right-cens.
 #Check that p >= 2.
    if (p <= 1)
    stop("p must be 2 or greater")
#Check that p is integer.
    if (floor(p) != p)
    stop("p must be an integer")
#Store data as vectors.
    xvec <- as.vector(x)
    yvec <- as.vector(y)
    dx   <- as.vector(dx)
    dy   <- as.vector(dy)
    mean <- as.vector(mean)
    xc   <- as.vector(xc)
    yc   <- as.vector(yc)
    nx   <- length(xvec)
    ny   <- length(yvec)
#Check that status and observations have same length in x,y.
    if (length(dx) != nx)
        stop("length of x and dx must agree")
    if (length(dy) != ny)
        stop("length of y and dy must agree")
```

```
    if (length(xc) != nx)
        stop("length of xc and dx must agree")
    if (length(dy) != ny)
        stop("length of yc and dy must agree")
#Check that there are enough data in x,y.
    if (nx <= length(mean))
        stop("need more observations than length of mean in x")
    if (ny <= 2*length(mean) + 1)
        stop("need more observations than length of mean in y")
#Check that status are only 0,1,2.
    if (any((dx != 0) & (dx != 1) & (dx != 2)))
        stop("dx must be 0(right-censored) or 1(uncensored) or 2(left-
         censored")
    if (any((dy != 0) & (dy != 1) & (dy != 2)))
        stop("dy must be 0(right-censored) or 1(uncensored) or 2(left-
         censored")
#Check that xvec,yvec are numeric.
    if (!is.numeric(xvec))
        stop("x must be numeric")
    if (!is.numeric(yvec))
        stop("y must be numeric")
    if (!is.numeric(mean))
        stop ("mean must be numeric")
#Check that H dimensions are consistent with lengths of x,y.
    if (dim(H)[1] != length(x) )
        stop("dim(H)[1] must equal length(x)")
    if (dim(H)[2] != p*length(y) )
```

```
        stop("dim(H)[2] must equal p*length(y)")
#"Clean" data using code from Wdataclean5
    #("Weighted Dataclean5", found in emplik package by Dr. Mai Zhou).
    #Clean x data.
    niceorderx <- order(x, -dx)
    x   <-  x[niceorderx]
    dx  <- dx[niceorderx]
    wx  <- wx[niceorderx]
    xc  <- xc[niceorderx]
    t1  <- x[-1]  != x[-nx]
    t2  <- dx[-1] != dx[-nx]
    t3  <- xc[-1] != xc[-nx]
    t <- t1 | t2 | t3
    ind <- c(which(t | is.na(t)), nx)
    csumwx <- cumsum(wx)
    x  <- x[ind]
    dx <- dx[ind]
    wx <- diff(c(0, csumwx[ind]))
    H   <- as.matrix(H[niceorderx, ][ind,])
    #Clean y data.
    niceordery <- order(y, -dy)
    y   <-  y[niceordery]
    dy  <- dy[niceordery]
    wy  <- wy[niceordery]
    yc  <- yc[niceordery]
    t1  <- y[-1]  != y[-ny]
    t2  <- dy[-1] != dy[-ny]
```

```
    t3  <- yc[-1] != yc[-ny]
    t <- t1 | t2 | t3
    ind <- c(which(t | is.na(t)), ny)
    csumwy <- cumsum(wy)
    y  <- y[ind]
    dy <- dy[ind]
    wy <- diff(c(0, csumwy[ind]))
    for (k in 1:p) {
        H[, ((k-1)*ny+1):(k*ny)] <-
          H[, ((k-1)*ny+1):(k*ny)][,niceordery][,ind] }
#For data with status 0 or 1, set status of highest datum to 1.
    xindex10 <- which(dx != 2)
    yindex10 <- which(dy != 2)
    dx[xindex10[length(xindex10)]] <- 1
    dy[yindex10[length(yindex10)]] <- 1
# For data with status 2 or 1, set status of lowest datum to 1.
    xindex12 <- which(dx != 0)
    yindex12 <- which(dy != 0)
    dx[xindex12[1]] <- 1
    dy[yindex12[1]] <- 1
#Put censored,uncensored data into respective vectors.
    xd0  <-  x[dx == 0]
    wxd0 <- wx[dx == 0]
    xd1  <-  x[dx == 1]
    wxd1 <- wx[dx == 1]
    xd2  <-  x[dx == 2]
    wxd2 <- wx[dx == 2]
```

```
    yd0  <-  y[dy == 0]
    wyd0 <- wy[dy == 0]
    yd1  <-  y[dy == 1]
    wyd1 <- wy[dy == 1]
    yd2  <-  y[dy == 2]
    wyd2 <- wy[dy == 2]
#Check that there are at least 2 uncensored data for x,y.
    if (length(xd1) <= 1)
        stop("need more distinct uncensored x obs.")
    if (length(yd1) <= 1)
        stop("need more distinct uncensored y obs.")
#Store vector lengths.
    nx0 <- length(xd0)
    ny0 <- length(yd0)
    nx1 <- length(xd1)
    ny1 <- length(yd1)
    nx2 <- length(xd2)
    ny2 <- length(yd2)
    nx  <- length(x)
    ny  <- length(y)
#LR denominator (unconstrained):
   #Doubly-censored case.
    if ( (nx0>0) & (nx2>0 ) ) {
      temp2x   <- myWCY(x = x, d = dx, wt = wx)
      logelx00 <- temp2x$logEL
      jumpxu   <- temp2x$jump
      }
```

```
 if ( (ny0>0) & (ny2>0 ) ) {
   temp2y   <- myWCY(x = y, d = dy, wt = wy)
   logely00 <- temp2y$logEL
   jumpyu   <- temp2y$jump
   }
#Right-censored case.
 if ( (nx0>0) & (nx2==0 ) ) {
   temp2x   <- myWKM(x = x, d = dx, w = wx)
   logelx00 <- temp2x$logel
   jumpxu   <- temp2x$jump[temp2x$jump>0]
   }
 if ( (ny0>0) & (ny2==0 ) ) {
   temp2y   <- myWKM(x = y, d = dy, w = wy)
   logely00 <- temp2y$logel
   jumpyu   <- temp2y$jump[temp2y$jump>0]
   }
#Left-censored case.
 if ( (nx0==0) & (nx2>0 ) ) {
   dlx <- dx
   dlx [dlx == 2] <- 0
   temp2x   <- myWKM(x = x, d = rev(dlx), w = rev(wx))
   logelx00 <- temp2x$logel
   jumpxu   <- rev(temp2x$jump[temp2x$jump>0])
   }
 if ( (ny0==0) & (ny2>0 ) ) {
   dly <- dy
   dly[dly == 2] <- 0
```

```
   temp2y   <- myWKM(x = y, d = rev(dly), w = rev(wy))
   logely00 <- temp2y$logel
   jumpyu   <- rev(temp2y$jump[temp2y$jump>0])
   }
 #Uncensored case.
  if ( (nx0==0) & (nx2==0 ) ) {
   logelx00 <- sum(wxd1 * log(wxd1/sum(wxd1)))
   jumpxu   <- wxd1/sum(wxd1)
   }
  if ( (ny0==0) & (ny2==0 ) ) {
   logely00 <- sum(wyd1 * log(wyd1/sum(wyd1)))
   jumpyu   <- wyd1/sum(wyd1)
   }
 #Calculate likelihood.
  logel00 <- logelx00 + logely00
 #Calculate NPMLE.
   NPMLE=rep(NA,p)
   H1 <- as.matrix(H[which(dx==1),])
     for (j in 1:p) {
     H2 <- H1[,((j-1)*ny+1):(j*ny)]
     H2 <- H2[,which(dy==1)]
     NPMLE[j]=jumpxu%*%H2%*%jumpyu
     }
   #LR numerator (constrained):
#Create mean-centered H, denoted as Hmc.
Hmc <- matrix(NA, nrow=nx, ncol=p*ny)
for (k in 1:p) {
```

```
   M <- matrix(mean[k], nrow=nx, ncol=ny)
   Hmc[,((k-1)*ny+1):(k*ny)] <-
      H[,((k-1)*ny+1):(k*ny)] - M }

#Create uncensored version of Hmc, denoted as Hu.
 whichdx <- which(dx == 1)
 whichdy <- which(dy == 1)
 Hu <- matrix(NA, nrow=nx1,ncol=p*ny1)
   for (k in 1:p) {
     Hu[,((k-1)*ny1+1):(k*ny1)] <-
      Hmc[,((k-1)*ny+1):(k*ny)][whichdx,whichdy] }

#Calculate Hmu and Hnu matrices (mean-centered).
 Hmu <- matrix(NA,nrow=p, ncol=ny1*nx1)
 Hnu <- matrix(NA,nrow=p, ncol=ny1*nx1)
 for (i in 1:p) {
   for (k in 1:nx1) {
     Hmu[i, ((k-1)*ny1+1):(k*ny1)] <-
       Hu[k,((i-1)*ny1+1):(i*ny1)] } }
 for (i in 1:p) {
   for (k in 1:ny1) {
     Hnu[i,((k-1)*nx1+1):(k*nx1)] <- Hu[(1:nx1),(i-1)*ny1+k]} }

#Initialize muvec,nuvec using unconstrained jumps.
 muvec <- jumpxu
 nuvec <- jumpyu
#Initialize kx,ky and kkx,kky.
```

```
  #For each rt-cens x,y value, kx,ky holds xd1,yd1 index of first
  #uncensored value to its right.
  #For each left-censored value, kkx,kky holds xd1,yd1 index of
  #first uncensored value to its left
   if (nx0>0) {
     kx <- rep(NA, nx0)
      for (i in 1:nx0) {kx[i] <- 1 + nx1 - sum(xd1 > xd0[i])}
    }
   if (nx2>0) {
     kkx <- rep(NA, nx2)
     for (i in 1:nx2) {kkx[i] <- sum(xd1 < xd2[i])}
    }
   if (ny0>0) {
     ky <- rep(NA, ny0)
      for (j in 1:ny0) {ky[j] <- 1 + ny1 - sum(yd1 > yd0[j])}
    }
   if (ny2>0) {
     kky <- rep(NA, ny2)
     for (j in 1:ny2) {kky[j] <- sum(yd1 < yd2[j])}
    }
#Initialize iteration counter num and log-likelihood-holder
#logvec.
   num <- 1
   logvec <- rep(0,maxit)
#Repeat EM repeatedly for maxit iterations.
     while (num <= maxit) {
#Initialize weights wxd1new,wyd1new.
```

```
      wxd1new <- wxd1
      wyd1new <- wyd1
#Perform Expectation step on uncensored x,y weights.
      if (nx0>0) {
        surx <- rev(cumsum(rev(muvec)))
        for (i in 1:nx0) {wxd1new[kx[i]:nx1] <- wxd1new[kx[i]:nx1] +
          wxd0[i] * muvec[kx[i]:nx1]/surx[kx[i]] }
        }
      if (nx2>0) {
        cdfx <- cumsum(muvec)
        for (i in 1:nx2) {wxd1new[1:kkx[i]] <- wxd1new[1:kkx[i]] +
          wxd2[i] * muvec[1:kkx[i]]/cdfx[kkx[i]] }
        }
      if (ny0>0) {
        sury <- rev(cumsum(rev(nuvec)))
        for (j in 1:ny0) {wyd1new[ky[j]:ny1] <- wyd1new[ky[j]:ny1] +
          wyd0[j] * nuvec[ky[j]:ny1]/sury[ky[j]] }
         }
      if (ny2>0) {
        cdfy <- cumsum(nuvec)
        for (j in 1:ny2) {wyd1new[1:kky[j]] <- wyd1new[1:kky[j]] +
          wyd2[j] * nuvec[1:kky[j]]/cdfy[kky[j]] }
         }
#Perform Maximization step on uncensored x,y jumps.
      temp3 <- el2.test.wtm(wxd1new, wyd1new, muvec,
        nuvec, Hu, Hmu, Hnu, p)
      muvec <- temp3$muvec1
```

```
     nuvec <- temp3$nuvec1
#Calculate loglikelihood so its convergence can be tracked.
     logelx <- sum(wxd1 * log(muvec))
      if (nx0>0) {
       surx <- rev(cumsum(rev(muvec)))
       logelx <- logelx + sum(wxd0 * log(surx[kx]))
      }
      if (nx2>0) {
       cdfx <- cumsum(muvec)
       logelx <- logelx + sum(wxd2 * log(cdfx[kkx]))
      }
    logely <- sum(wyd1 * log(nuvec))
      if (ny0>0) {
       sury <- rev(cumsum(rev(nuvec)))
       logely <- logely + sum(wyd0 * log(sury[ky]))
      }
      if (ny2>0) {
       cdfy <- cumsum(nuvec)
       logely <- logely + sum(wyd2 * log(cdfy[kky]))
      }
    logel <- logelx + logely
    logvec[num]<- logel
    num <- num + 1
    }
 #Store -2log(likelihood ratio) in tval.
    tval <- 2 * (logel00 - logel)
 #Return results of the el2.cen.EM function.
```

```
        list(xd1=xd1,yd1=yd1,temp3=temp3, mean=mean, NPMLE=
         NPMLE, logel00=logel00, logel=logel, "-2LLR"=tval,
         Pval=1 - pchisq(tval, df = length(mean)), logvec=logvec,
         sum_muvec=sum(muvec), sum_nuvec=sum(nuvec) )
}
```

Annotated code for el2.test.wts

```
el2.test.wts <- function (u,v,wu,wv,mu0,nu0,indicmat,mean) {

#If mean is not a scalar then stop.
    if (length(mean) != 1)
        stop("mean must be a scalar")

#Calculate scalars to be used in calculations.
    sumwu <- sum(wu)
    sumwv <- sum(wv)
    nu <- length(u)
    nv <- length(v)

#Calculate matrix and vectors to be used in calculations.
    indic4mu <- nu0 %*% t(indicmat)
    indic4nu <- mu0 %*% indicmat

#Calculate the delta for lam search.
    du <- 0.02 * sumwu/abs(sum(indic4mu))
    dv <- 0.02 * sumwv/abs(sum(indic4nu))
    dd <- min(du,dv)

#Define lamfun, where lamfun(true lam) = 0.
    lamfun <- function(lam, wu, wv, sumwu, sumwv, indic4mu,
      indic4nu, indicmat) {
    mu <- wu/(sumwu+lam*indic4mu)
```

```
    nu <- wv/(sumwv+lam*indic4nu)
    return(mu %*% indicmat %*% t(nu))
      }

#Find upper and lower bounds on lam, for uniroot.
    if (lamfun(0, wu, wv, sumwu, sumwv, indic4mu,
      indic4nu,indicmat) == 0)
      lam0 <- 0 else

    {if (lamfun(0, wu, wv, sumwu, sumwv, indic4mu,
      indic4nu,indicmat) > 0)
       {lo <- 0
       up <- dd
       while (lamfun(up, wu, wv, sumwu, sumwv, indic4mu,
         indic4nu,indicmat) > 0)
         {up <- up + dd} } else

           {up <- 0
           lo <- -dd
           while (lamfun(lo, wu, wv, sumwu, sumwv, indic4mu,
             indic4nu,indicmat) < 0)
             {lo <- lo - dd}}}

#Find lam using uniroot.
 lam <- uniroot(lamfun, lower = lo, upper = up,
    tol = 1e-09, wu=wu, wv=wv, sumwu=sumwu, sumwv=sumwv,
    indic4mu=indic4mu, indic4nu=indic4nu, indicmat=indicmat)$root
```

```
#Calculate updated mu1,nu1 using the lagrangian lam.
    mu1 <- wu/(sumwu + lam * nu0 %*% t(indicmat))
    nu1 <- wv/(sumwv + lam * mu0 %*% indicmat)

#List the original data & weights plus the p_i ,lam0, and mean.
    list(u=u, wu=wu, jumpu=mu1, v=v, wv=wv, jumpv=nu1, lam=lam, mean)
}
```

Annotated code for el2.test.wtm

```
el2.test.wtm <- function (wxd1new, wyd1new, muvec, nuvec, Hu, Hmu,
  Hnu, p, maxit=10) {

#Initialize vectors and scalars
  lam <- rep(0,p)
  nx1 <- length(muvec)
  ny1 <- length(nuvec)
  swxd1 <- sum(wxd1new)
  swyd1 <- sum(wyd1new)
  constmat <- matrix(NA, nrow=maxit, ncol=p)

for (r in 1:maxit) {
#Calculate muvec1
  muvec1 <- rep(NA, nx1)
  for (k in 1:nx1) {
    muvec1[k] <- wxd1new[k] / abs(swxd1 +  lam %*%
     Hmu[,((k-1)*ny1+1):(k*ny1)] %*% nuvec)
  }
#Calculate nuvec1
  nuvec1 <- rep(NA, ny1)
  for (k in 1:ny1) {
    nuvec1[k] <- wyd1new[k] / abs(swyd1 +  lam %*%
     Hnu[,((k-1)*nx1+1):(k*nx1)] %*% muvec)
  }
#Calculate constraint vector
```

```
  constraint <- rep(NA, p)
  for (k in 1:p) {
    constraint[k] <- muvec1 %*% Hu[,((k-1)*ny1+1):(k*ny1)] %*% nuvec1
  }
#Calculate constraintp matrix (derivative of constraint vector wrt
#lam)
 constraintp <- matrix(0, nrow=p, ncol=p) for (b in 1:p) {
  for (k in 1:p) {
    Hb <- Hu[, ((b-1)*ny1+1):(b*ny1)]
    Hk <- Hu[, ((k-1)*ny1+1):(k*ny1)]
    fact1 <- as.vector((Hk%*%nuvec)*(muvec1^2)/wxd1new)
    fact2 <- as.vector((muvec%*%Hk)*(nuvec1^2)/wyd1new)
    for (i in 1:nx1) {
      for (j in 1:ny1) {
        constraintp[k,b] <- constraintp[k,b] -
        Hb[i,j] * (nuvec1[j]*fact1[i] +
        muvec1[i]*fact2[j]) } } } }
constmat[r,] <- constraint
#Run Newton-Raphson routine
lam1 <- lam - constraint %*% solve(constraintp)
lam <- lam1 }
list(constmat=constmat, lam=lam1, muvec1=muvec1,
nuvec1=nuvec1) }
```

Code for myWKM

```
myWKM<-function (x, d, zc = rep(1, length(d)), w = rep(1,
length(d))) {
    if (any((d != 0) & (d != 1)))
        stop("d must be 0(right-censored) or 1(uncensored)")
    temp <- myWdataclean3(x, d, zc, w)
    dd <- temp$dd
    ww <- temp$weight
    dd[length(dd)] <- 1
    allrisk <- rev(cumsum(rev(ww)))
    survP <- cumprod(1 - (dd * ww)/allrisk)
    jumps <- -diff(c(1, survP))
    logel <- sum(ww[dd == 1] * log(jumps[dd == 1])) + sum(ww[dd ==
        0] * log(survP[dd == 0]))
    list(times = temp$value, jump = jumps, surv = survP, logel = logel,
        weight=ww)
 }
```

Code for myWCY

```
myWCY <- function (x, d, zc = rep(1, length(d)), wt = rep(1,
length(d)), maxit = 25, error = 1e-09) {
    xvec <- as.vector(x)
    nn <- length(xvec)
    if (nn <= 1)
        stop("Need more observations")
    if (length(d) != nn)
        stop("length of x and d must agree")
    if (any((d != 0) & (d != 1) & (d != 2)))
        stop("d must be 0(right-censored) or 1(uncensored)
        or 2(left-censored)")
    if (!is.numeric(xvec))
        stop("x must be numeric")
    temp <- myWdataclean3(z = xvec, d = d, zc = zc, wt = wt)
    x <- temp$value
    d <- temp$dd
    w <- temp$weight
    INDEX10 <- which(d != 2)
    d[INDEX10[length(INDEX10)]] <- 1
    INDEX12 <- which(d != 0)
    d[INDEX12[1]] <- 1
    xd1 <- x[d == 1]
    if (length(xd1) <= 1)
        stop("need more distinct uncensored obs.")
    xd0 <- x[d == 0]
```

```
xd2 <- x[d == 2]
wd1 <- w[d == 1]
wd0 <- w[d == 0]
wd2 <- w[d == 2]
m <- length(xd0)
mleft <- length(xd2)
if ((m > 0) && (mleft > 0)) {
    pnew <- wd1/sum(wd1)
    n <- length(pnew)
    k <- rep(NA, m)
    for (i in 1:m) {
        k[i] <- 1 + n - sum(xd1 > xd0[i])
    }
    kk <- rep(NA, mleft)
    for (j in 1:mleft) {
        kk[j] <- sum(xd1 < xd2[j])
    }
    num <- 1
    while (num < maxit) {
        wd1new <- wd1
        sur <- rev(cumsum(rev(pnew)))
        cdf <- 1 - c(sur[-1], 0)
        for (i in 1:m) {
            wd1new[k[i]:n] <- wd1new[k[i]:n] + wd0[i]
            * pnew[k[i]:n]/sur[k[i]]
        }
        for (j in 1:mleft) {
```

```
                wd1new[1:kk[j]] <- wd1new[1:kk[j]] + wd2[j] *
                  pnew[1:kk[j]]/cdf[kk[j]]
            }
            pnew <- wd1new/sum(wd1new)
            num <- num + 1
        }
        sur <- rev(cumsum(rev(pnew)))
        cdf <- 1 - c(sur[-1], 0)
        logel <- sum(wd1 * log(pnew)) + sum(wd0 * log(sur[k])) +
            sum(wd2 * log(cdf[kk]))
    }
    return(list(logEL = logel, time = xd1, jump = pnew, surv = 1 -
        cdf, prob = cdf, xd1 = xd1, xd0 = xd0, xd2 = xd2, wd1 = wd1,
        wd0 = wd0, xd2 = wd2))
}
```

Annotated code for myWdataclean2

```
myWdataclean2<-function (z, d, wt = rep(1, length(z))) {
#sort z,d,wt (ascending) wrt z, using -d to order ties
    niceorder <- order(z, -d)
    sortedz <- z[niceorder]
    sortedd <- d[niceorder]
    sortedw <- wt[niceorder]
#store length of sortedd in n
    n <- length(sortedd)
#y1 checks for jumps in sortedz using offsets of sortedz
    y1 <- sortedz[-1] != sortedz[-n]
#y2 checks for "jumps" in sortedd
    y2 <- sortedd[-1] != sortedd[-n]
#y checks for jumps (in sortedz or sortedd) using y1 and y2
    y <- y1 | y2
#ind stores jump indices (final index will be  n)
    ind <- c(which(y | is.na(y)), n)
#csum is cumulative sum of the weights
    csumw <- cumsum(sortedw)
#value contains the (unique) obs in sortedz
#dd contains the status of obs in sortedz
#weight has the weights of the obs in sortedz
    list(value = sortedz[ind], dd = sortedd[ind],
    weight = diff(c(0,csumw[ind])))
}
```

Annotated code for myWdataclean3

```
myWdataclean3<-function (z, d, zc = rep(1, length(z)), wt = rep(1,
length(z))) {
#sort z,d,wt (ascending) wrt z, using -d to order ties
    niceorder <- order(z, -d)
    sortedz <- z[niceorder]
    sortedd <- d[niceorder]
    sortedw <- wt[niceorder]
    sortedzc <- zc[niceorder]
#store length of sortedd in n
    n <- length(sortedd)
#y1 checks for jumps in sortedz using offsets of sortedz
    y1 <- sortedz[-1] != sortedz[-n]
#y2 checks for "jumps" in sortedd
    y2 <- sortedd[-1] != sortedd[-n]
#y3 checks for "jumps" in sortedzc
    y3 <- sortedzc[-1] != sortedzc[-n]
#y checks for jumps (in sortedz or sortedd or sortedzc) using
#y1,y2,y3
    y <- y1 | y2 | y3
#ind stores jump indices (final index will be  n)
    ind <- c(which(y | is.na(y)), n)
#csum is cumulative sum of the weights
    csumw <- cumsum(sortedw)
#value contains the (unique) obs in sortedz
#dd contains the status of obs in sortedz
```

```
#weight has the weights of the obs in sortedz
    list(value = sortedz[ind], dd = sortedd[ind], weight = diff(c(0,
        csumw[ind])))
}
```

Appendix B: Simulation Code

The annotated code used to generate the probability plots in Figures 3.1 to 3.6 and the change-point simulation in section 5.1 is listed below.

In Figures 3.1 to 3.4 we generate 5000 sets of $(\boldsymbol{x}, \boldsymbol{y})$ pairs, where x and y have the same Weibull distribution and $\boldsymbol{x}$ and $\boldsymbol{y}$ both have length 50. For each $(\boldsymbol{x}, \boldsymbol{y})$ pair we also calculate the corresponding value of $-2ELLR$ using $el.cen.EMs$. We then plot $-2ELLR$ against the corresponding 5000 quantiles of the $\chi^2_{(1)}$ distribution.

In all cases the hypothesis is that $P(x \geq y) = 0.5$, which is of course true since x and y have the same distribution. We therefore expect that $-2ELLR$ based on H_o will asymptotically follow a $\chi^2_{(1)}$ distribution, as surmised in Chapter 3. This is seen to be true in all four plots since all four plots closely follow the 45-degree straight line (except for the extreme-right tail). As noted in chapter 3, this (informally) validates the use of the $\chi^2_{(1)}$ distribution to calculate an approximate p-value for H_o.

In Figure 3.5 we generate 5000 sets of $(\boldsymbol{x}, \boldsymbol{y})$ pairs, where x and y have Weibull distributions with different means and $\boldsymbol{x}$ and $\boldsymbol{y}$ both have length 50. For each $(\boldsymbol{x}, \boldsymbol{y})$ pair we also calculate the corresponding value of $-2ELLR$ using $el.cen.EMs$. We then plot $-2ELLR$ against the corresponding 5000 quantiles of the $\chi^2_{(1)}$ distribution.

In this case the hypothesis is that $E(X - Y) = 0$ which is not true. We therefore do *not* expect that $-2ELLR$ based on H_o will asymptotically follow a $\chi^2_{(1)}$ distribution. And in fact $-2ELLR$ does not follow a $\chi^2_{(1)}$ distribution since it deviates significantly from the 45-degree straight line.

In Figure 3.6 we plot the same 5000 values of $-2LLR$ from Figure 3.5 against the corresponding 5000 quantiles of the $\chi^2_{(df=1,ncp=0.85)}$ distribution, where ncp=0.85 is a non-centrality parameter found by trial and error. The plot follows the 45-degree straight line up to a quantile of about 7.5. which corresponds to a percentile of 96.5. This suggests that $-2LLR$ follows a non-central $\chi^2_{(1)}$ distribution when H_o is false and the alternate hypothesis is true.

R-Code for Uncensored Probability Plot, Fig. 3.1

```
#Produce a Probability Plot, Uncensored Data
 N=50
 M=1

#Calculate chisq(1) quantiles for sample of N
orderx=qchisq(1:N/(N+1),1)

#Calculate N values of -2LLR
 mymatrixu2=matrix(0,nrow=M,ncol=N)
 dx=rep(1,50)
 dy=rep(1,50)
 for (i in 1:M){
 for (j in 1:N) {
 dummy=round(rweibull(100,shape=0.8662,scale=985.9))
 x=dummy[1:50]
 y=dummy[51:100]
 dummy2=el2.cen.EMs(x=x,dx=dx,y=y,dy=dy)
 mymatrixu2[i,j]=dummy2$"-2LLR" } }

 for (i in 1:M) {
 mymatrixu2[i,]=sort(mymatrixu2[i,])}
 ordery=apply(mymatrixu2,2,mean)

#Plot -2LLR values vs chisq(1) values
 plot(orderx,ordery,xlab="Chisq(1) Quantiles", ylab="Sorted
 -2LLR Values", main="Probability Plot for 5000
```

```
Uncensored Data Sets")
abline(0,1)
```

R-Code for Right-Censored Probability Plot, Fig. 3.2

```
simulatr=function(nx=50, ny=50, weib=T, expon=T,
shape=0.8662, scale=985.9, mu=6.6, sigma2=1) {

#The simulatr function simulates a two-sample data set from a
#survival study.  Both samples have the same Weibull(shape,scale)
#distribution (or alternatively the same lognormal(mu,sigma^2)
#distribution).  Both are right-censored.
#The default settings give an avg censoring rate of about 23%.

#nx is the number of data in the x-sample
#ny is the number of data in the y-sample
#weib is indicator whether to use Weibull to generate durations
#shape is the Weibull shape parameter
#scale is the Weibull scale parameter
#mu is the mean of the log of the durations
#sigma2 is the variance of the log of the durations

#Distributions are parameterized as follows:
#Weibull cdf     F(t) = 1 - exp(-(t/scale)^shape)
#Lognormal pdf   f(t)=(1/(x*sqrt(2*pi*sigma2))*
#exp(-(log(t)-mu)^2/(2*sigma2))

#For reference, calculate the true population mean
if (weib==T) {truemean=scale*gamma((shape+1)/shape)
} else {truemean= exp(mu+0.5*sigma2)}
```

```
#Generate nx+ny random Weibull(shape,scale) data (or
#lognormal(mu,sigma2) data if weib=F).
n=nx+ny
if (weib==T) {mydata=round(rweibull(n,shape,scale))
} else {mydata=round(rlnorm(n,meanlog=mu,sdlog=sigma2^0.5))}

#myend is the length of the study
myend=2000

#We allow subjects to enter the study up to the half-way point
#Calculate cutoff (which is the last day to enter the study).
cutoff=round(0.5*myend)

#Weight contains probabilities associated with entering on each day
#Use an exponential weight if expon=T
if (expon==T) {weight= exp(-(1:cutoff)/cutoff)
#Otherwise use the same weight for all days
} else {weight=rep(1,cutoff)}

#days contains the allowable entry days
days=1:cutoff

#Generate nx+ny random times of entry into study.
#The entry-day weights are exponential if expon=T.
#The entry-day weights are uniform if expon=F.
entry=sample(days, size=n, prob=weight, replace=T)
```

```
#Generate nx+ny random Weibull(shape,scale) durations (or
#lognormal(mu,sigma2) durations if weib=F)
if (weib==T) {dur=round(rweibull(n,shape,scale))
} else {dur=round(rlnorm(n,meanlog=mu,sdlog=sigma2^0.5))}

#Add the entry times to the Weibull durations to get event-dates
#An event date is the date when 1) the subject dies or 2) the
#subject is censored by exceeding the end of the study before dying
date=entry+dur

#Generate status vectors with ones where date <=myend,
#and with zeros where date > myend
status = as.numeric(date <= myend)

#Wherever the status is zero, replace the date with myend
date[which(status==0)]=myend

#Recalculate the durations
dur=date-entry

#Define the x and y durations and status
durx=dur[1:nx]
statusx=status[1:nx]
dury=dur[(nx+1):(n)]
statusy=status[(nx+1):(n)]

#Order the status and durations according to ascending durations
```

```
ordx=order(durx)
statusx=status[ordx]
durx=durx[ordx]
ordy=order(dury)
statusy=status[ordy]
dury=dury[ordy]

#List durx, statusx, dury, statusy, truemean
list("durx"=durx, "statusx"=statusx, "dury"=dury,
"statusy"=statusy, "truemean"=truemean)
}

#Produce a Probability Plot, Right-Censored Data
N=5000
M=1

#Calculate chisq(1) quantiles for sample of N
orderx=qchisq(1:N/(N+1),1)

#Calculate N values of -2LLR
mymatrixr=matrix(0,nrow=M,ncol=N)
for (i in 1:M) {
for (j in 1:N) {
dummy=simulatr(nx=50,ny=50)
x=dummy$durx
y=dummy$dury
dx=dummy$statusx
```

```
dy=dummy$statusy
dummy2=el2.cen.EMs(x=x,dx=dx,y=y,dy=dy)
mymatrixr[i,j]=dummy2$"-2LLR" } }

for (i in 1:M) {
mymatrixr[i,]=sort(mymatrixr[i,]) }
ordery=apply(mymatrixr,2,mean)

#Plot -2LLR values vs chisq(1) values
plot(orderx,ordery,xlab="Chisq(1) Quantiles", ylab="Sorted
-2LLR Values", main="Probability Plot for 5000
Right-Censored Data Sets")
abline(0,1)
```

R-Code for Left-Censored Probability Plot, Fig. 3.3

```
simulatl=function(nx=50, ny=50, weib=T, expon=T,
shape=0.8662, scale=985.9, mu=6.6, sigma2=1) {

#The simulatl function simulates a two-sample data set from a
#survival study. Both samples have the same Weibull(shape,scale)
#distribution (or alternatively the same lognormal(mu,sigma^2)
#distribution). Both samples are left-censored.
#The default settings give an avg censoring rate of about 18%.

#The protocol of the study is as follows. Two groups of children
#are admitted into a study to record at what age they experience
#a certain event. Their ages are measured in days. Children
#can enter the study until a specified cutoff date. The child's age
#at entry into the study must fall within a specified range. The
#children are monitored daily for certain characteristics as they
#age. The children all remain in the study until they experience the
#event, that is, no child is "right-censored." However some children
#on their first day in the study are found to have experienced the event
#at some unknown earlier age. These children who have already
#experienced the event when they are admitted are "left-censored."

#nx is the number of data in the x-sample
#ny is the number of data in the y-sample
#weib is indicator whether to use Weibull to generate durations
#shape is the Weibull shape parameter
#scale is the Weibull scale parameter
```

```
#mu is the mean of the log of the durations
#sigma2 is the variance of the log of the durations

#Distributions are parameterized as follows:
#Weibull cdf     F(t) = 1 - exp(-(t/scale)^shape)
#Lognormal pdf   f(t)=(1/(x*sqrt(2*pi*sigma2))*
#exp(-(log(t)-mu)^2/(2*sigma2))

#For reference, calculate the true population mean
if (weib==T) {truemean=scale*gamma((shape+1)/shape)
} else {truemean= exp(mu+0.5*sigma2)}

#We allow subjects to enter the study on days 1-600
cutoff=600

#Weight contains probabilities associated with entering on each day
#Use an exponential weight if expon=T
#Otherwise use the same weight for all days
if (expon==T) {weight=exp(-(1:cutoff)/cutoff)
} else {weight=rep(1,cutoff)}

#days contains the allowable entry days
days=1:cutoff

#Generate nx+ny random days of entry into study.
#The entry-day weights are exponential if expon=T.
#The entry-day weights are uniform if expon=F.
```

```
n=nx+ny
entryday=sample(days, size=n, prob=weight, replace=T)

#Assign random entry ages to each subject.
#We allow subjects to enter the study at ages 100-200.
entryage=sample(100:200, 100, replace=T)

#Generate nx+ny random Weibull(shape,scale) ages (or
#lognormal(mu,sigma2) ages if weib=F).
#These ages are the ages at which the children
#experience the event.
if (weib==T) {eventage=round(rweibull(n,shape,scale))
} else {eventage=round(rlnorm(n,meanlog=mu,sdlog=sigma2^0.5))}

#Define the observed event-age at which the event occurs.
obsage=eventage
indexleft=which(eventage<entryage)
obsage[indexleft]=entryage[indexleft]

#Define the status
status=rep(1,n)
status[indexleft]=2

#Define the x and y observed ages and status
obsagex=obsage[1:nx]
statusx=status[1:nx]
obsagey=obsage[(nx+1):(nx+ny)]
```

```
statusy=status[(nx+1):(nx+ny)]

#Order the status and durations according to ascending durations
ordx=order(obsagex)
statusx=status[ordx]
obsagex=obsagex[ordx]
ordy=order(obsagey)
statusy=status[ordy]
obsagey=obsagey[ordy]

#List obsagex, statusx, obsagey, statusy, truemean
list("obsagex"=obsagex, "statusx"=statusx, "obsagey"=obsagey,
"statusy"=statusy, "truemean"=truemean)
}

#Produce a Probability Plot, Left-Censored Data
N=5000
M=1

#Calculate chisq(1) quantiles for sample of N
orderx=qchisq(1:N/(N+1),1)

#Calculate N values of -2LLR
mymatrixl=matrix(0,nrow=M,ncol=N)
for (i in 1:M) {
for (j in 1:N) {
dummy=simulatl(nx=50,ny=50)
```

```
x=dummy$obsagex

y=dummy$obsagey

dx=dummy$statusx

dy=dummy$statusy

dummy2=el2.cen.EMs(x=x,dx=dx,y=y,dy=dy)

mymatrixl[i,j]=dummy2$"-2LLR" } }


for (i in 1:M) {

mymatrixl[i,]=sort(mymatrixl[i,]) }

ordery=apply(mymatrixl,2,mean)


#Plot -2LLR values vs chisq(1) values

plot(orderx,ordery,xlab="Chisq(1) Quantiles", ylab="Sorted

-2LLR Values", main="Probability Plot for 5000

Left-Censored Data Sets")

abline(0,1)
```

R-Code for Doubly-Censored Probability Plot, Fig. 3.4

```
simulatd=function(nx=50, ny=50, weib=T, expon=T,
shape=0.8662, scale=985.9, mu=6.6, sigma2=1) {

#The simulatd function simulates a two-sample data set from a
#survival study.  Both samples have the same Weibull(shape,scale)
#distribution (or alternatively the same lognormal(mu,sigma^2)
#distribution).  Both samples are doubly-censored.
#The default settings give an avg censoring rate of about 26%.

#The protocol of the study is as follows. Two groups of children
#are admitted into a study to record at what age they experience
#a certain event. Their ages are measured in days. Children
#can enter the study until a specified cutoff date. The child's age
#at entry into the study must fall within a specified range. The
#children are monitored daily for certain characteristics as they
#age. The children all remain in the study until they experience
#the event or until the study ends. Any child who has not
#experienced the event by the end of the study is "right-censored."
#Some children on their first day in the study are found to have
#experienced the event at some unknown earlier age. These children
#who have already experienced the event when they are admitted are
#"left-censored."

#nx is the number of data in the x-sample
#ny is the number of data in the y-sample
#weib is indicator whether to use Weibull to generate event-ages
```

```
#shape is the Weibull shape parameter
#scale is the Weibull scale parameter
#mu is the mean of the log of the event-ages
#sigma2 is the variance of the log of the event-ages

#Distributions are parameterized as follows:
#Weibull cdf     F(t) = 1 - exp(-(t/scale)^shape)
#Lognormal pdf   f(t)=(1/(x*sqrt(2*pi*sigma2))*
#exp(-(log(t)-mu)^2/(2*sigma2))

#For reference, calculate the true population mean
if (weib==T) {truemean=scale*gamma((shape+1)/shape)
} else {truemean= exp(mu+0.5*sigma2)}

#end is the length of the study
myend=2500

#We allow subjects to enter the study on days 1-250
cutoff=250

#Weight contains probabilities associated with entering on each day
#Use an exponential weight if expon=T
#Otherwise use the same weight for all days
if (expon==T) {weight= exp(-(1:cutoff)/cutoff)
} else {weight=rep(1,cutoff)}
```

```
#days contains the allowable entry days
days=1:cutoff

#Generate random times of entry into study to the subjects.
#The entry-day generation is exponential if expon=T.
#The entry-day generation is uniform if expon=F.
n=nx+ny
entryday=sample(days, size=n, prob=weight, replace=T)

#Assign random entry ages to the subjects.
#We allow subjects to enter the study at ages 10-250.
entryage=sample(10:250, 100, replace=T)

#Generate nx+ny random Weibull(shape,scale) ages (or
#lognormal(mu,sigma2) ages if weib=F).
#These ages are the ages at which the children
#experience the event.
if (weib==T) {eventage=round(rweibull(n,shape,scale))
} else {eventage=round(rlnorm(n,meanlog=mu,sdlog=sigma2^0.5))}

#obsage is the age at which the event is observed.
obsage=eventage
indexleft=which(eventage<entryage)
obsage[indexleft]=entryage[indexleft]
endage=entryage+(myend-entryday)
indexrt=which(endage<eventage)
obsage[indexrt]=endage[indexrt]
```

```
#Define the status
status=rep(1,n)
status[indexleft]=2
status[indexrt]=0

#Define the x and y observed ages and status
obsagex=obsage[1:nx]
statusx=status[1:nx]
obsagey=obsage[(nx+1):(nx+ny)]
statusy=status[(nx+1):(nx+ny)]

#Order the status and durations according to ascending durations
ordx=order(obsagex)
statusx=statusx[ordx]
obsagex=obsagex[ordx]
ordy=order(obsagey)
statusy=statusy[ordy]
obsagey=obsagey[ordy]

#List obsagex, statusx, obsagey, statusy, truemean
list("obsagex"=obsagex, "statusx"=statusx, "obsagey"=obsagey,
"statusy"=statusy, "truemean"=truemean)
}

#Produce a Probability Plot, Doubly-Censored Data
N=5000
```

```
M=1

#Calculate chisq(1) quantiles for sample of N
orderx=qchisq(1:N/(N+1),1)

#Calculate N values of -2LLR
 mymatrixd=matrix(0,nrow=M,ncol=N)
 for (i in 1:M) {
 for (j in 1:N) {
 dummy=simulatd(nx=50,ny=50)
 x=dummy$obsagex
 y=dummy$obsagey
 dx=dummy$statusx
 dy=dummy$statusy
 dummy2=el2.cen.EMs(x=x,dx=dx,y=y,dy=dy)
 mymatrixd[i,j]=dummy2$"-2LLR" } }

for (i in 1:M) {
mymatrixd[i,]=sort(mymatrixd[i,]) }
ordery=apply(mymatrixd,2,mean)

#Plot -2LLR values vs chisq(1) values
plot(orderx,ordery,xlab="Chisq(1) Quantiles", ylab="Sorted
-2LLR Values", main="Probability Plot for 5000
Doubly-Censored Data Sets")
abline(0,1)
```

R-Code for Right-Censored Probability Plot with Unequal Means, Figs. 3.5 and 3.6

```
simulatr.unequal=function(nx=50, ny=50, expon=T,
shape=c(0.8662,0.9709), scale=c(985.9,1200)) {

 #The simulatr.unequal function simulates a two-sample data
 #set from a survival study.  Both samples have a
 #Weibull(shape,scale) distribution.  Both are right-censored.
 #Mean of X is 1060.3, mean of Y is 1215.7

 #nx is the number of data in the x-sample
 #ny is the number of data in the y-sample
 #shape is the Weibull shape parameter
 #scale is the Weibull scale parameter

 #Weibull distribution is parameterized as follows:
 #Weibull cdf F(t) = 1 - exp(-(t/scale)^shape)

 #For reference, calculate the true population mean
 truemean=scale*gamma((shape+1)/shape)

 #myend is the length of the study myend=2000

 #We allow subjects to enter the study up to the half-way point
 #Calculate cutoff (which is the last day to enter the study).
 cutoff=round(0.5*myend)
```

```
#Weight contains probabilities associated with entering on each day
#Use an exponential weight if expon=T
if (expon==T) {weight= exp(-(1:cutoff)/cutoff)
#Otherwise use the same weight for all days
} else {weight=rep(1,cutoff)}

#days contains the allowable entry days days=1:cutoff

#Generate nx+ny random times of entry into study.
#The entry-day weights are exponential if expon=T.
#The entry-day weights are uniform if expon=F.
entry=sample(days, size=n, prob=weight, replace=T)

#Generate nx+ny random Weibull(shape,scale) durations
mydurx=round(rweibull(nx,shape[1],scale[1]))
mydury=round(rweibull(ny,shape[2],scale[2]))
dur=c(mydurx,mydury)

#Add the entry times to the Weibull durations to get event-dates
#An event date is the date when 1) the subject dies or 2) the
#subject is censored by exceeding the end of the study before dying
date=entry+dur

#Generate status vectors with ones where date <=myend,
#and with zeros where date > myend
status = as.numeric(date <= myend)
```

```
#Wherever the status is zero, replace the date with myend
date[which(status==0)]=myend

#Recalculate the durations
dur=date-entry

#Define the x and y durations and status
durx=dur[1:nx]
statusx=status[1:nx]
dury=dur[(nx+1):(nx+ny)]
statusy=status[(nx+1):(nx+ny)]

#Order the status and durations according to ascending durations
ordx=order(durx)
statusx=status[ordx]
durx=durx[ordx]
ordy=order(dury)
statusy=status[ordy]
dury=dury[ordy]

#List durx, statusx, dury, statusy, truemean
list("durx"=durx, "statusx"=statusx, "dury"=dury, "statusy"=statusy,
"truemean"=truemean) }

#Produce a Probability Plot, Right-Censored Data
#with Unequal Means
N=5000
```

```
M=1

#Calculate chisq(1) quantiles for sample of N
orderx=qchisq(1:N/(N+1),1)

#Calculate N values of -2LLR
mymatrixr=matrix(0,nrow=M,ncol=N)
for(i in 1:M) {
for (j in 1:N) {
dummy=simulatr.unequal(nx=50,ny=50)
x=dummy$durx
y=dummy$dury
dx=dummy$statusx
dy=dummy$statusy
dummy2=el2.cen.EMs(x=x,dx=dx,y=y,dy=dy,fun=function(x,y){x-y},mean=0)
mymatrixr[i,j]=dummy2$"-2LLR" } }

for (i in 1:M) {
mymatrixr[i,]=sort(mymatrixr[i,]) }
ordery=apply(mymatrixr,2,mean)

#Plot -2LLR values vs chisq(1) values, Figure 3.5
plot(orderx,ordery,xlab="Chisq(df=1) Quantiles",
ylab="Sorted -2LLR  Values",
main="Probability Plot for 5000 Right-Censored Data Sets")
abline(0,1)
```

```
#Plot -2LLR values vs chisq(df=1, ncp=0.85) values, Figure 3.6
 plot(orderx,ordery,xlab="Chisq(df=1, ncp=0.85) Quantiles",
 ylab="Sorted -2LLR  Values",
 main="Probability Plot for 5000 Right-Censored Data Sets")
 abline(0,1)
```

R-Code for Change-Point Analysis in 5.1

```
#Set.seed is used to reproduce example results
 set.seed(1234)
#Generate 30 X-data using mean 0, stdev 1
 x=rnorm(30,0,1)
#Generate 30 Y-data using mean 1, stdev 1
 y=rnorm(30,1,1)
#Initialize 30 X-status values as uncensored
 dx=rep(1,30)
#Initialize 30 Y-status values as uncensored
 dy=rep(1,30)
#Randomly choose 4 of X-data
 censx=sample(1:30,4)
#Randomly choose 4 of Y-data
 censy=sample(1:30,4)
#Censor the 4 chosen X-data
 dx[censx]=0
#Censor the 4 chosen Y-data
 dy[censy]=0
#Concatenate X and Y data
 data=c(x,y)
#Concatenate X and Y status
 status=c(dx,dy)
#We know points 1-28 must be from X
#We know points 33-60 must be from Y
#So we can estimate the diff in means from
#them using the Kaplan_Meier estimator
```

```
 dummy1=myWKM(data[1:28],status[1:28])
 meanx=sum(dummy1$times*dummy1$jump)
 dummy2=myWKM(data[33:60],status[33:60])
 meany=sum(dummy2$times*dummy2$jump)
 estdiff=meany-meanx
#Plot the control chart
 plot(1:n,c(x,y),xlab="Time Points",
 ylab="Values", main="Control Chart")
#Initialize the p-value and NPMLE vectors
 myvec=rep(0,5)
 mynpmle=rep(0,5)
#Run loop using five possible last-X values of 28-32
 for(j in 28:32) {
#Extract 10 values X[j-9]...X[j]
 xvec=data[(j-9):j]
#Extract 10 values Y[(j+1)]...Y[j+10]
 yvec=data[(j+1):(j+10)]
#Set corresponding X status values
 statx=status[(j-9):j]
#Set corresponding Y status values
 staty=status[(j+1):(j+10)]
#Check p-value for Ho: mean(y)-mean(X)=estdiff
 dummy=el2.cen.EMs(xvec,statx,yvec,staty,
 fun=function(x,y){y-x},mean=estdiff)
#Record p-value and NPMLE
 myvec[j-27]=dummy$Pval
 mynpmle[j-27]=dummy$funNPMLE
```

```
 }
#List estimated p-values and NPMLEs
 myvec
 mynpmle
```

Bibliography

[1] Casella, G. and Berger, R. L. (2002). *Statistical Inference*, Duxbury Press, Pacific Grove, California.

[2] Chang, M. and Yang, G. (1987). "Strong Consistency of a Nonparametric Estimator of the Survival Function with Doubly Censored Data," *Ann. Stat.*, **15**, pp. 1536-1547.

[3] Dempster, A., Laird, N., and Rubin, D. (1977). "Maximum Likelihood from Incomplete Data via the *EM* Algorithm." *J. Roy. Statist. Soc.*, Series B, **39**, pp. 1-38.

[4] Gentleman, R. and Ihaka, R. (1996) "R: A Language for Data Analysis and Graphics." *J. Comput. Graph. Stat.*, **5**, pp. 299-314.

[5] Gomez, G., Julia, O., and Utzet, F. (1992). "Survival Analysis for Left-Censored Data." In Klein, J. and Goel, P. (eds.), *Survival Analysis: State of the Art.* Kluwer Academic Publishers, Boston, pp. 269-288.

[6] Hadley, J. A. and McNeil, B. J. (1982). "The Meaning and Use of the Area under a Receiver Operating Characteristic (ROC) Curve." *Radiology*, **143**, pp. 29-36.

[7] Kalbfleisch, J. D. and Prentiss, R. L. (2002). *The Statistical Analysis of Failure Time Data.* Wiley-Interscience, Hoboken, New Jersey.

[8] Kaplan, E. L. and Meier, P. (1958) "Nonparametric Estimation from Incomplete Observations." *J. Amer. Stat. Assoc.*, **53**, pp. 457-481.

[9] Krzanowski, W. J. and Hand, D. J. (2009). *ROC Curves for Continuous Data*, Chapman & Hall/CRC Press, Boca Raton, Florida.

[10] Murphy, S.A. (1995). "Likelihood Ratio-Based Confidence Intervals in Survival Analysis." *J. Amer. Stat. Assoc.*, **90**, pp. 1399-1405.

[11] Murphy, S. A. and Van der Vaart, A. W. (1997) "Semiparametric Likelihood Ratio Inference." *Ann. Stat.* **25**, pp. 1471-1509.

[12] Owen, A. B. (2001). *Empirical Likelihood*. Chapman & Hall/CRC, Boca Raton.

[13] Pan, X. R. and Zhou, M. (1999). "Using one-parameter sub-family of distributions in empirical likelihood with censored data." *J. Stat. Plan. Infer.*, **75**, pp. 379-392.

[14] Pan, X. R. and Zhou, M. (2002). "Empirical likelihood ratio in terms of cumulative hazard function for censored data." *J. Multivar. Anal.*, **80**, pp. 166-188.

[15] Peto, R. (1973). "Experimental Survival Curves for Interval-Censored Data." *J. Roy. Statist. Soc.*, Series C, **22**, pp. 86-91.

[16] Qin, G. and Zhou, X. H. (2006). "Empirical Likelihood Inference for the Area Under the ROC Curve." *Biometrics*, **62**, pp. 613-622.

[17] Turnbull, B. W. (1974). "Nonparametric Estimation of a Survivorship Function with Doubly Censored Data." *J. Amer. Stat. Assoc.*, **69**, pp. 169-173.

[18] Turnbull, B. W. (1976). "The Empirical Distribution Function with Arbitrarily Grouped, Censored and Truncated Data." *J. Roy. Statist. Soc.*, Series B, **38**, pp. 290-295.

[19] Wang, Q., Yao, Lili, and Lai, Peng (2009) "Estimation of the area under ROC curve with censored data." *J. Stat. Plan. Infer.*, **139**, pp. 1033-1044.

[20] Wilks, S. S. (1938). "The large-sample distribution of the likelihood ratio for testing composite hypotheses." *Ann. Math. Statist.*, **9**, pp. 60-62.

[21] Wood, A. T. A., Do, A., and Broom, B. M. (1996). "Sequential Linearization of Empirical Likelihood Constraints with Application to U-Statistics." *J. Comput. Graph. Stat.*, **5**, pp. 365-385.

[22] Zhou, M. (1991). "Some Properties of the Kaplan-Meier Estimator for Independent Nonidentically Distributed Random Variables." *Ann. Stat.*, **19**, pp. 2266-2274.

[23] Zhou, M. (2005). "Empirical likelihood ratio with arbitrarily censored/truncated data by EM algorithm." *J. Comput. Graph. Stat.*, **14**, pp. 643-656.

Vita

Name

• William H. Barton

Date of Birth

• 1951

Place of Birth

• Plainfield, New Jersey, U.S.A.

Education

• 1973, B.S. Chemical Engineering, Virginia Polytechnic Institute, Blacksburg, VA

• 2004, M.S. Statistics, University of Kentucky, Lexington, KY

• 2010 (expected), Ph.D. Statistics, University of Kentucky, Lexington, KY

Employment

• 1973-1991, Engineer, IBM, Austin, TX

• 1991-1995, Engineer, IBM, Endicott, NY

• 1995-2010, Engineer, Lexmark International, Lexington, KY

Honors

• Holder of 3 U.S. Patents