Implementation of flexible lifetime distributions in regression models to estimate survival times

Implementación de distribuciones flexibles de duración de vida en modelos de regresión para estimar tiempos de supervivencia

Jaime Mosquera Gutiérrez

Universidad Nacional de Colombia
Facultad de Ciencias, Escuela de Estadística
Medellín, Colombia
2023

Implementation of flexible lifetime distributions in regression models to estimate survival times

Jaime Mosquera Gutiérrez

Thesis presented as partial requirement for the degree of:
Master of Science - Statistics

Advisor:
Ph.D. Freddy Hernández Barajas

Line of research:
Generalized Models
Research Group:
Grupo de Investigación en Estadística UNAL sede Medellín

Universidad Nacional de Colombia
Facultad de Ciencias, Escuela de Estadística
Medellín, Colombia
2023

Implementación de distribuciones flexibles de duración de vida en modelos de regresión para estimar tiempos de supervivencia

Jaime Mosquera Gutiérrez

Tesis o trabajo de grado presentada(o) como requisito parcial para optar al título de:
Magister en Ciencias - Estadística

Director:
Ph.D. Freddy Hernández Barajas

Línea de Investigación:
Modelos Generalizados
Grupo de Investigación:
Grupo de Investigación en Estadística UNAL sede Medellín

Universidad Nacional de Colombia
Facultad de Ciencias, Escuela de Estadística
Medellín, Colombia
2023

The reason that, having started as a chemist, I became a statistician was that Statistics seemed to me of much greater importance. It was about the catalysis of scientific method itself.

George E.P. Box

Acknowledgements

I would like to express my sincere gratitude to my parents, Jaime Mosquera and Ruthbely Gutiérrez, for their emotional support throughout my Master's degree program. Without their constant encouragement, I would not have been able to achieve this significant milestone in my academic career. I am also deeply grateful to my grandmother Dora Ortiz, who has been a constant source of motivation and inspiration. This thesis and my Master's degree are a shared dream that we have finally been able to realize together.

I would like to extend my thanks to Leslie Avendaño, who played a small but crucial role in this process. It was during an eureka moment she caused when I was able to finish the implementation of `maxlogLreg` function, the central routine of the **EstimationTools** package. Additionally, Leslie's contribution allowed me to attend successfully the Portuguese Statistics Society Conference.

I would also like to thank my friend Laura Gómez for her unwavering support and encouragement when it seemed impossible to finish. She was an inspiration to me as we were both pursuing our Master's degrees simultaneously, and her love and dedication to her own thesis motivated me. How to forget Natalia Ortiz, Jessica Pulzara, and Catalina Burgos for being such supportive and collaborative classmates throughout our academic program. I will always cherish the memories of our academic discussions mixed with anecdotes and conversations. I especially value the enduring friendship of Natalia, which has stood the test of time.

A special thank you goes to Professor Russell Alpizar-Jara, who has helped me to explore new frontiers in the application of statistics. He also played a vital role in my participation in the Portuguese Statistics Society Conference for the second time, and his friendship has been invaluable. Professors Juan Carlos Salazar, Juan Carlos Correa and Isabel Ramirez not only helped me with important and interesting subjects but also inspired certain ideas that are reflected in this work. I wil have eternal gratitude and admiration for them.

Finally, I would like to express my deepest appreciation to Professor Freddy Hernandez for sharing his knowledge with me, inspiring me, and adding value to the development of this work. His patience and understanding were also greatly appreciated.

Abstract

In the fields of reliability engineering and survival analysis, it is common to find experiments from which data characterized by non-monotonic hazard functions—such as bathtub-shaped or unimodal functions—can be obtained. To model datasets like those mentioned, flexible lifetime distributions are frequently proposed. However, many of these distributions are not yet implemented in statistical software for fitting regression models. In this context, we have developed the **EstimationTools** R package, which offers a general-purpose framework for fitting and evaluating distributional regression models. This framework employs a syntax that mirrors mathematical notation. We leveraged maximum likelihood estimation and computed the log-likelihood function just using the probability mass/density function implemented in the R global workspace. Our framework is particularly suited for datasets where the response variable follows a flexible lifetime distribution, thereby enabling users to estimate distribution parameters in relation to covariates, even with censored data. It also provides graphical diagnostic tools through Martingale, Cox-Snell, Deviance and Randomized Quantile Residuals. The software has been tested on well-known datasets from health sciences and reliability studies, demonstrating its potential to develop models for applications such as flood prediction, churn analysis, credit risk modeling, recidivism, and student dropout. Overall, our work represents a versatile alternative for fitting parametric time-to-event models.

Keywords: bathtub hazard, deviance residuals, distributional regression, flexible lifetime distributions, Martingale residuals, maximum likelihood estimation, randomized quantile residuals.

Resumen

En los campos de la ingeniería de confiabilidad y el análisis de supervivencia, es común encontrar experimentos de los cuales se pueden obtener datos caracterizados por funciones de riesgo no monótonas, tales como funciones en forma de bañera o unimodales. Para modelar conjuntos de datos como los mencionados, se proponen frecuentemente distribuciones de vida útil flexibles. Sin embargo, muchas de estas distribuciones aún no están implementadas en software estadístico para ajustar modelos de regresión. En este contexto, hemos desarrollado el paquete R **EstimationTools**, que ofrece un marco de trabajo de propósito general para ajustar y evaluar modelos de regresión distribucional. Este marco utiliza una sintaxis que refleja la notación matemática. Utilizamos la estimación de máxima verosimilitud y calculamos la función de log-verosimilitud basada en la función de masa/densidad de probabilidad implementada en el espacio de trabajo global de R. Nuestro marco es particularmente adecuado para conjuntos de datos donde la variable de respuesta sigue una distribución de vida útil flexible, lo que permite a los usuarios estimar parámetros de distribución en relación con covariables, incluso con datos censurados. También proporciona herramientas de diagnóstico gráfico a través de residuos de Martingala, Cox-Snell y Deviance. El software ha sido probado en conjuntos de datos bien conocidos de las ciencias de la salud y estudios de confiabilidad, demostrando su potencial para desarrollar modelos para aplicaciones como la predicción de inundaciones, análisis de abandono de clientes, modelado de riesgo crediticio, reincidencia y deserción estudiantil. En general, nuestro trabajo representa una alternativa versátil para ajustar modelos paramétricos de tiempo hasta el evento.

Palabras clave: función hazard en forma de bañera, residuos deviance, regresión distribucional, distribuciones de vida útil flexibles, residuos de Martingala, estimación de máxima verosimilitud, residuos cuantil aleatorizados.

Contents

List of Figures

List of Tables

List of acronyms and terms

AFT Accelerated Failure Time (models).

API Application Programming Interface.

BFGS Broyden-Fletcher-Goldfarb-Shanno.

CDF Cumulative Density Function.

CHF Cumulative hazard function.

EDA Exploratory data analysis.

EW Exponentiated Weibull.

IG Inverse Gaussian.

KM Kaplan-Meier.

MLE Weibull distribution, parametrization type II.

OW Odd Weibull.

PDF Probability Density Function.

PORT Portable Fortran programs for numerical computation.

TTT Total time on test transform.

Weibull I Weibull distribution, parametrization type I.

Weibull II Weibull distribution, parametrization type II.

ZIP Zero-inflated Poisson.

List of Symbols

β_{jk} Regression parameter k for distribution parameter j.

$\boldsymbol{\eta}_j$ Vector of linear predictors.

ε Relative error.

ϵ_i Residuals (noise term) for distributions used in AFT models.

Θ Parameter space of dimension p (number of parameters).

$\boldsymbol{\theta}$ Distribution parameter vector (actual values).

$\hat{\boldsymbol{\theta}}$ Distribution parameter vector (estimated values).

ϕ Empirical TTT transform.

b_k Number of events that happened at time k (Kaplan-Meier estimator).

c_k Number of subjects at risk (known to have survived) at time k (Kaplan-Meier estimator).

d_i Unit deviance contributed by data point i.

D Deviance goodness-of-fit statistic.

$f(\cdot)$ Probability Density function.

$F(\cdot)$ Cumulative Density Function.

$h(\cdot)$ Hazard function.

$\mathbf{H}(\cdot)$ Hessian matrix.

n Sample size.

R_{il} An element of the $\boldsymbol{R}$ matrix to indicate the survival status (censoring direction).

$S(\cdot)$ Survival function.

t Value (realisation) of a random variable denoting the time-to-event.

T_i Response (random) variable for data point i denoting the time-to-event.

$H_{F^{-1}}$ TTT transform.

x_i Covariate for data point i.

$\mathbf{X}_j$ Design matrix with explanatory variable columns for the distribution parameter j.

$\boldsymbol{y}$ Generic symbol denoting a response (random) variable in vector notation.

y_i Generic symbol denoting the response variable for data point i in a regression model.

Y_i Generic symbol denoting the response (random) variable for data point i.

$\mathcal{D}$ An arbitrary distribution.

$\mathcal{I}(\cdot)$ Expected Fisher information matrix.

$\mathcal{J}(\cdot)$ Observed Fisher information matrix.

$l(\cdot)$ Log-likelihood function.

$\mathcal{L}(\cdot)$ Likelihood function.

Part I

Motivation and First Steps

1. Introduction

> ” *Begin challenging your own assumptions. Your assumptions are your windows on the world. Scrub them off every once in a while, or the light won't come in.*
>
> — **Alan Alda**
>
> (Actor, director and writer)

In survival analysis and reliability engineering, it is usual to perform estimation with a parametric approach in order to model duration times. For this purpose, the Weibull distribution is the most popular and the most widely used in reliability and lifetime data analysis (Almalki and Nadarajah, 2014). However, this and other traditional distributions in such areas (such as normal, exponential, gamma, log-normal, and log-logistic) are not flexible because they may only generate new non-monotonic and/or unimodal curves for the hazard function. This is inadequate to interpret data that do not have such characteristics (Ortega et al., 2011), for example, those that are modelled with the bathtub-shaped hazard function, among others. Therefore, distributions that cover other families and particular cases have been proposed, which give them flexibility and allow them to have curves with different shapes.

Additionally, many flexible lifetime distributions are not yet available in the aforementioned packages or in any statistical software, which is an obstacle making estimations quickly and reliably. In addition, there are regression models only for some of them, so they are not available to model the heterogeneity of a population or to study lifetime behaviour regarding covariates (Lawless, 2002) in survival and reliability applications.

1.1. Illustrative examples

This section presents parameter estimation with two datasets of duration times that are effectively modelled with flexible lifetime distributions. These application examples will be found throughout this thesis.

1.1.1. Time to failure on electronic equipment

This application example consists of an experiment in which data represent the time to fail (in hours) of 18 electronic devices and they are showed in table **1-1**. The results of this test are analysed by Wang (2000) and Cooray (2015) in order to model failure times.

Table 1-1: Failure times of 18 electronic devices (Wang, 2000; Cooray, 2015).

5	11	21	31	46	75
98	122	145	165	195	224
245	293	321	330	350	420

As we all know, a positive-valued distribution is necessary to model this type of dataset. Therefore, we chose the Weibull I[1]distribution, whose probability density function (PDF) is shown in equation 1-1

$$f(t|\mu,\sigma) = \mu\sigma t^{\sigma-1}\exp\left(-\mu t^{\sigma}\right), \qquad t>0, \tag{1-1}$$

with μ, $\sigma > 0$ and t denoting the time. We also employed the odd Weibull distribution (OW) proposed by Cooray (2015). Its PDF is given by the following expression:

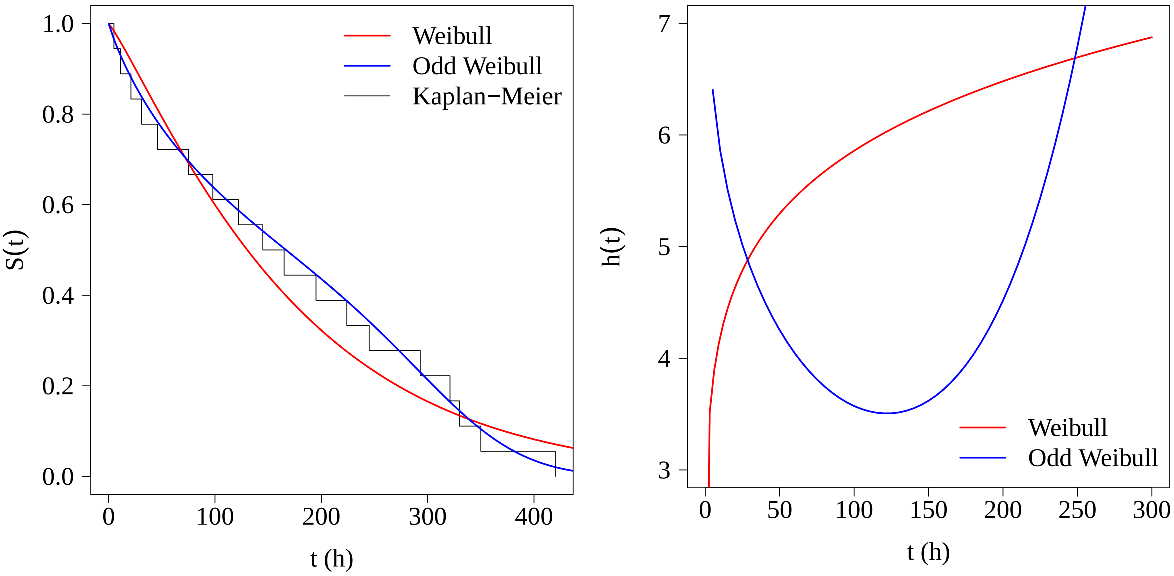
$$f(t|\mu,\sigma,\nu) = \mu\sigma\nu t^{\sigma-1}\exp\left(\mu t^{\sigma}\right)(\exp\left(\mu t^{\sigma}\right)-1)^{\nu-1}[1+(\exp\left(\mu t^{\sigma}\right)-1)^{\nu}]^{-2}, \qquad t>0, \tag{1-2}$$

with μ, σ, $\nu > 0$. The Kaplan-Meier estimator and estimated hazard functions are illustrated in figure **1-1**. As can be seen in the plots and AIC values, the odd Weibull distribution generates a better fit and can produce a bathtub-shaped hazard rate function.

1.1.2. Head and neck cancer data

The data correspond to a randomised clinical trial to compare two therapies for head and neck cancer shown in table **1-2** described by Efron (1988): 51 patients were treated with radiation only, 45 patients were treated with radiation plus chemotherapy, and the response variable was the survival times in days.

[1] Weibull I refers to a specific parameterization of the Weibull distribution. In this document, we present three versions; therefore, you will encounter Weibull II and Weibull III later on. This arrangement has been established to facilitate a comparative analysis between the standard Weibull distribution and its extensions.



(a) Plot of survival functions.

(b) Plot of hazard functions.

Figure 1-1: Estimated survival curves (left) and estimated hazard curves (right) for the time to failure on electronic equipment using Weibull and odd Weibull models. The dark curve on the left side corresponds to the Kaplan-Meier estimator for the dataset (Kaplan and Meier, 1958).

Table 1-2: Head and neck cancer data (Efron, 1988).

Radiation		Radiation + Chemotherapy	
Time	Status	Time	Status
7	1	37	1
34	1	84	1
42	1	92	1
63	1	94	1
64	1	110	1
74	0	112	1
83	1	119	1
84	1	127	1
91	1	130	1
108	1	133	1
112	1	140	1
129	1	146	1
133	1	155	1

Table 1-2: Head and neck cancer data (Efron, 1988). *(continued)*

Radiation		Radiation + Chemotherapy	
Time	Status	Time	Status
133	1	159	1
139	1	169	0
140	1	173	1
140	1	179	1
146	1	194	1
149	1	195	1
154	1	209	1
157	1	249	1
160	1	281	1
160	1	319	1
165	1	339	1
173	1	432	1
176	1	469	1
185	0	519	1
218	1	528	0
225	1	547	0
241	1	613	0
248	1	633	1
273	1	725	1
277	1	759	0
279	0	817	1
297	1	1092	0
319	0	1245	0
405	1	1331	0
417	1	1557	1
420	1	1642	0
440	1	1771	0
523	1	1776	1
523	0	1897	0
583	1	2023	0
594	1	2146	0
1101	1	2297	0
1116	0		

Table 1-2: Head and neck cancer data (Efron, 1988). *(continued)*

Radiation		Radiation + Chemotherapy	
Time	Status	Time	Status
1146	1		
1226	0		
1349	0		
1412	0		
1417	1		

Khan (2018) fitted the Weibull accelerated failure time (AFT) model (usually used) and proposed a new AFT exponentiated Weibull (EW) model using the treatment status as a covariate. Because AFT models are not the focus of this project, we fitted both Weibull and odd Weibull models with direct estimation by maximum likelihood. The Weibull II distribution (WEI) is fitted with the following parametrization[2]

$$f(t|\rho,\kappa) = \kappa\rho\left(\rho t\right)^{\kappa-1}\exp\left[-\left(\rho t\right)^{\kappa}\right], \qquad t>0, \tag{1-3}$$

where $\rho,\ \kappa > 0$. The corresponding model equation is shown below

$$T_i \sim WEI(\rho_i,\ \kappa), \tag{1-4}$$

$$.\log(\rho_i) = \beta_{01} + \beta_{11}x_i, \tag{1-5}$$

$$\log(\kappa) = \beta_{02}, \tag{1-6}$$

where T_i is the response variable, β_{jk} are the regression parameters (with $j = 0, 1$ and $k = 1, 2$) and x_i is the covariate for data point i. The EW distribution is expressed as

$$f(t|\rho,\kappa,\gamma) = \kappa\rho\gamma\left(\rho t\right)^{\kappa-1}\exp\left[-\left(\rho t\right)^{\kappa}\right]\left\{1-\exp\left[-\left(\rho t\right)^{\kappa}\right]\right\}, \qquad t>0, \tag{1-7}$$

with $\rho,\ \kappa,\ \gamma > 0$. The model equation is as follows:

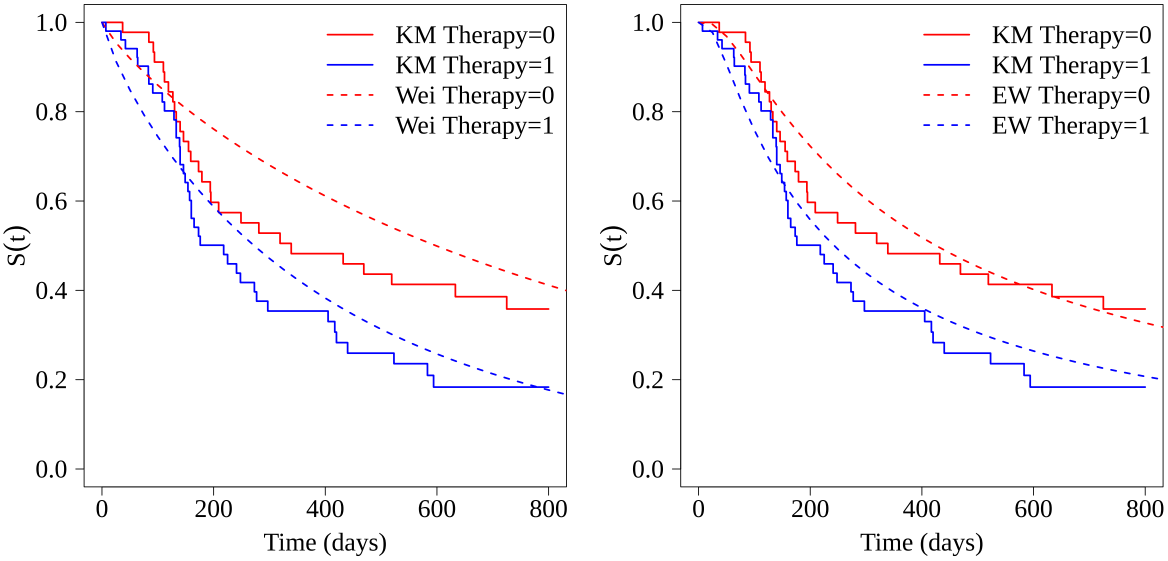
[2] Khan (2018) expressed the Weibull distribution using the rate parameter ρ, which directly represents the instantaneous failure rate of a component or system. Consequently, he chose an EW parametrization to be consistent with this characterisation.

$$
\begin{aligned}
T_i &\sim EW(\rho_i,\ \kappa,\ \gamma), && \text{(1-8)}\\
\log(\rho_i) &= \beta_{01} + \beta_{11}x_i, && \text{(1-9)}\\
\log(\kappa) &= \beta_{02}, && \text{(1-10)}\\
\log(\gamma) &= \beta_{03}. && \text{(1-11)}
\end{aligned}
$$

The indicator variable is a binary variable that takes on the value of either 1 or 0, indicating that the treatment involves radiation therapy alone or radiation plus chemotherapy, as follows:

$$
x_i = \begin{cases} 1 & \text{if subject } i \text{ receives the radiation therapy only,} \\ 0 & \text{if subject } i \text{ receives radiation plus chemotherap.} \end{cases} \qquad \text{(1-12)}
$$

The estimation under these two models is illustrated in figure **1-2**. AIC values and survival function plots show an improvement when the EW distribution is fitted instead of the Weibull distribution.



(a) Plot of Weibull survival functions.

(b) Plot of EW survival functions.

Figure 1-2: Fit for Weibull ($AIC = 1082.519$) and exponentiated Weibull ($AIC = 1064.556$) distributions.

1.2. Structure of the document

The rest of the document describes our approach to fit lifetime distributions. Chapter 2 presents a literature review that comprises a conceptual framework and the background of estimation in lifetime distributions. The following chapters introduces the **EstimationTools** R package, available in the `CRAN` (`https://cran.rstudio.com/web/packages/EstimationTools/index.html`) or in the `GitHub` development repository (`https://github.com/Jaimemosg/EstimationTools`). Consequently, chapter 3 presents our TTT plot implementation as an exploratory tool, and chapter 4 introduces the importance of high-precision computations in estimation, an essential aspect of our work; chapters 5 and 6 discuss estimation without covariates and regression models for flexible lifetime distributions, respectively. The latter contains the formulation of the regression models addressed by this project and introduces our estimation routines. Chapter 7 shows the implementation of residuals and graphic tools for diagnostics.

Despite many distributions and packages available to fit these types of models, a wide variety of these novel distributions are not yet available in some statistical software for general use, and hence many of them are not available in such a way that it is possible to fit linear regression models of their parameters in a straightforward way. Owing to this fact, the remainder of this document is about the odd Weibull distribution and its applications. Finally, in Chapter 9, some conclusions and future lines of work are offered.

Enjoy your read!

2. Theoretical framework

> *Obey the principles without being bound by them.*
>
> — **Bruce Lee**
>
> (Martial artist and actor)

Lifetime modelling is a central problem in survival analysis and reliability. Lifetime distributions are constantly proposed as an improvement with respect to classical distributions because they allow the modelling of different situations with a good fit. To use them, it is sufficient to have good estimation algorithms. In this chapter, we examine some concepts and definitions that provide the foundations of this document.

2.1. Censoring

Censoring is a condition that appears when there is a partial knowledge of the survival time observed in a sample unit (Prinja et al., 2010). There are three main censoring types (Leung et al., 1997):

- Right censoring: it occurs when the time in which the event of interest occurred exceeds a specific value, but such value is unknown.
- Left censoring: it occurs when the sample unit shows the event of interest before starting the study although the exact time is unknown. It is presented in clinical trials and other studies in which the starting point is defined.
- Interval censoring: it occurs when the time in which the event of interest occurs is between two values, but such a value is unknown.

The three censoring types are illustrated in figure **2-1**.

It is worth noting that there exists another more specific classification for right censoring (Lee and Wang, 2003; Colosimo and Ruiz Giolo, 2006a):

- Type I censoring: it occurs in studies completed within a specific time with right-censored sample units.

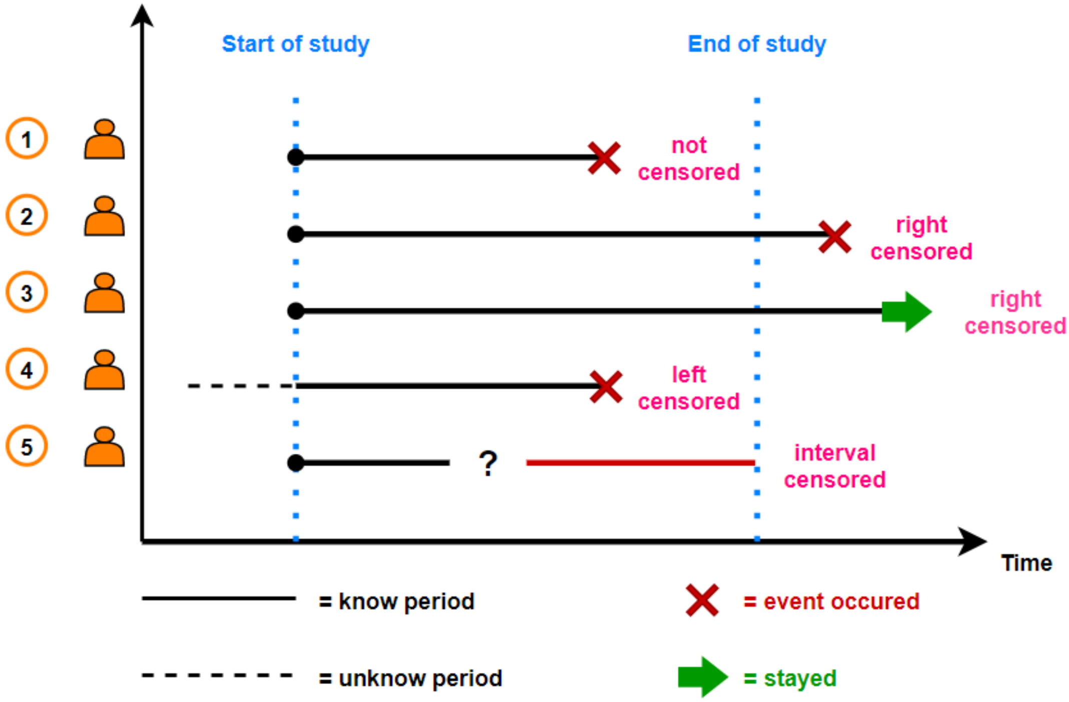


Figure 2-1: Illustration of censoring types (Gruman, 2021).

- Type II censoring: This occurs in studies whose completion is given by the observation of a predetermined number of events, so the remaining ones are right-censored events.

Estimation procedures must regard censored data points as they are present in the working dataset. Despite being incomplete, censored observations provide a piece of information about a subject's lifetime. Furthermore, omitting the censoring indicator in the computation of relevant statistics may lead to erroneous conclusions (Colosimo and Ruiz Giolo, 2006a).

2.2. Flexible lifetime distributions

The parametric statistics approach consists of using mathematical models to describe the frequency distribution of a population. Fisher (1958), in his work entitled *Statistical Methods for Research Workers*, assigned to these models the name of probability distributions and justified their application by arguing that such models provide the possibility to make predictions of the population from the sample. In particular, lifetime distributions have been useful for modelling experimental results and observational data in reliability, survival analysis, and actuarial areas (Marshall and Olkin, 2007).

One of the problems with this type of data is the remoteness of normality, mainly due to the analysis of time measurements or event counts, which are variables characterised by having a positive support (Meeker and Escobar, 1998). A key step whereby a solution to this problem was found by the Swedish scientist Waloddi Weibull. He proposed a cumulative density function for a component of multiple parts intuitively, from an exponential function, and then he added two parameters to give it flexibility. In his work, he showed that his function adjusted to seven different applications, ranging from modelling steel resistance fatigue to the height of the men who inhabited the British islands by that time. For further details, please read the Weibull's seminal work (Weibull, 1951).

From then on, new distributions have been created that arise from modifications of the Weibull distribution (Almalki and Nadarajah, 2014). These distributions may be classified within a taxonomy; whose grouping criterion is the distribution creation method. According to Murthy et al. (2004), there are five groups of modifications for the univariate Weibull distribution. Although thousands of different functions have been created, the following ones are some of them:

- **Type I models:** they emerge by transforming a Weibull variable. This is the case for the generalized extreme value distribution (Gumbel, 1941, 1958), the reflected Weibull distribution (Cohen, 1973), the inverse Weibull distribution (Drapella, 1993), the odd Weibull distribution (Cooray, 2006), the flexible Weibull (Bebbington et al., 2007).

- **Type II models:** they emerge from the generalisation of a Weibull distribution, which is achieved by adding new shape parameters. Distributions belonging to this group include the exponentiated Weibull (Mudholkar and Srivastava, 1993), the new modified Weibull distribution (Lai et al., 2003), the power generalized Weibull (Nikulin and Haghighi, 2007), the generalized modified Weibull (Carrasco et al., 2008) and the Weibull-Kumaraswamy (Cordeiro et al., 2010).

- **Type III models:** they emerge from the mix of one or more distributions. This category includes the Weibull-gamma distributions (Stacy, 1962), the additive Weibull (Xie and Lai, 1996), the beta-Weibull (Famoye et al., 2005) and the beta-Weibull Poisson (Percontini et al., 2013).

- **Type IV models:** they correspond to Weibull models of the original family that have parameters that vary over time.

- **Type V models:** they are discrete versions of the standard Weibull distribution. Different parametrizations of the discrete Weibull distribution stand out: the discrete inverse Weibull (Nakagawa and Osaki, 1975; Stein and Dattero, 1984; Padgett and Spurrier, 1985) and the discrete additive Weibull (Shafaei Nooghabi et al., 2011).

In recent years, packages for R programming language have been developed, such as **Newdistns**, with distributions applicable to the reliability area (Nadarajah and Rocha, 2016) or **FAdist**, with useful distributions to make predictions of time between floods (Aucoin, 2015), just mentioning two of them.

Weibull extensions, such as those mentioned above and other novel families with more than two parameters, correspond to what is known as flexible distributions (Ñíguez et al., 2019). These distributions can model skewness and kurtosis, and describe the mean and standard deviation much like two-parameter distributions normally do (Zhao et al., 2018). Moreover, they provide a wider range of shapes for the hazard rate function, which is particularly useful in survival analysis and reliability (Nair et al., 2018).

Flexible lifetime distributions arise when flexible distributions are used to model and analyse lifetime data. In this context, *lifetime* refers to situations in which the time to the occurrence of an event is of interest to individuals in some population (Lawless, 2003).

2.3. Maximum likelihood estimation

Let be $\boldsymbol{y}^\top = (y_1, y_2, ..., y_n)$ a random variable of n lifetime data points drawn from a population with distribution function $f(\cdot|\boldsymbol{\theta})$, where $\boldsymbol{y} \in \mathcal{S}$ and $\boldsymbol{\theta} \in \Theta$, with Θ is the parameter space of dimension p. Moreover, consider that $\mathcal{S} \in \mathbb{R}^k$ and $\Theta \in \mathbb{R}^k$.

The likelihood function of $\boldsymbol{\theta}$ for non-censored data is defined as follows:

$$\mathcal{L}(\boldsymbol{\theta}|\boldsymbol{y}) = \prod_{i=1}^{n} f(y_i|\boldsymbol{\theta}). \tag{2-1}$$

The maximum likelihood estimators (MLEs) are the values of the components in the parameter vector $\hat{\boldsymbol{\theta}}$ that maximises the likelihood function $\mathcal{L}(\cdot)$. For convenience, it is common to maximise the log-likelihood function, $l(\cdot)$. In short, the goal is

$$\hat{\boldsymbol{\theta}} = \underset{\boldsymbol{\theta} \in \Theta}{\arg \text{máx}}\; l(\boldsymbol{\theta}|\boldsymbol{y}), \tag{2-2}$$

where

$$l(\boldsymbol{\theta}|\boldsymbol{y}) = \log \mathcal{L}(\boldsymbol{\theta}|\boldsymbol{y}). \tag{2-3}$$

For censored samples, there are some additional likelihood contributions (Turkson et al., 2021)

- $\mathcal{L}_i(\boldsymbol{\theta}|y_i) = f(\boldsymbol{y}|\boldsymbol{\theta})$ for observed events,
- $\mathcal{L}_i(\boldsymbol{\theta}|y_i) = P_{\boldsymbol{\theta}}(Y_i > y_i) = S(y_i|\boldsymbol{\theta})$ for right-censored observations,
- $\mathcal{L}_i(\boldsymbol{\theta}|y_i) = P_{\boldsymbol{\theta}}(Y_i < y_i) = F(y_i|\boldsymbol{\theta})$ for left-censored observations,

where Y_i and y_i are the random variable and its corresponding value respectively, $f(\cdot)$ is the probability density function, $S(\cdot)$ is the survival function and $F(\cdot)$ is the cumulative density function. The generalised likelihood function becomes the following

$$l(\boldsymbol{\theta}|\boldsymbol{y}) = \log\left(\prod_{i=1}^{n}[f(y_i|\boldsymbol{\theta})]^{r_{i1}}\;[S(y_i|\boldsymbol{\theta})]^{r_{i2}}\;[F(y_i|\boldsymbol{\theta})]^{r_{i3}}\right), \tag{2-4}$$

where R_{il} is an element of the status matrix $\boldsymbol{R}$ defined as follows:

$$R_{il} = \begin{cases} 1, & \text{if sample unit } i \text{ has status } l, \\ 0, & \text{in other case,} \end{cases} \tag{2-5}$$

is the status matrix, with $i = 1, 2, \ldots, n$ and $l = 1, 2, 3$, with $l = 1$ for observed events, $l = 2$ for right-censored data, and $l = 3$ for left censored data.

The entire development and application of the maximum likelihood method revolves around the likelihood principle, whereby all the evidence of the sample to model parameters is contained in the likelihood function (Fisher, 1922). This principle has allowed the extension of the maximum likelihood method to estimate parameter regression models.

The maximisation of the likelihood is an optimisation problem regularly addressed by using the first derivative method, as proposed to estimate parameters in probability density functions (Fisher, 1912). In many cases, non-linear equation systems are generated from this procedure. To find a solution, numerical methods derived from the Newton-Raphson method have been used, and various non-gradient-based algorithms have been proposed.

2.3.1. Regularity Conditions

Rossi (2018) points that properties of MLEs rely on certain characteristics of the probability function $f(y_i|\boldsymbol{\theta})$ known as *weak regularity conditions*. The function $f(y_i|\boldsymbol{\theta})$ is said to satisfy these conditions if

1. The score function is always defined, or equivalently

$$\frac{\partial}{\partial\boldsymbol{\theta}}\log(f(y_i|\boldsymbol{\theta})) < \infty, \quad \forall y \text{ and } \forall\boldsymbol{\theta}\in\Theta.$$

2. When computing the expected value

   a. For any statistic $T(\boldsymbol{y})$,

$$\frac{\partial}{\partial\boldsymbol{\theta}}\int_S T(\boldsymbol{y})f(\boldsymbol{\theta}|\boldsymbol{y})d\boldsymbol{y} = \int_S T(\boldsymbol{y})\frac{\partial}{\partial\boldsymbol{\theta}}f(\boldsymbol{\theta}|\boldsymbol{y})d\boldsymbol{y}, \quad \forall\boldsymbol{\theta}\in\Theta,$$

when $f(y_i|\boldsymbol{\theta})$ is the probability function of a continuous random variable.

b. For any statistic $T(\boldsymbol{y})$,

$$\frac{\partial}{\partial\boldsymbol{\theta}}\sum_{\mathcal{S}} T(\boldsymbol{y})f(\boldsymbol{\theta}|\boldsymbol{y}) = \sum_{\mathcal{S}} T(\boldsymbol{y})\frac{\partial}{\partial\boldsymbol{\theta}}f(\boldsymbol{\theta}|\boldsymbol{y}), \quad \forall\boldsymbol{\theta}\in\Theta,$$

when $f(y_i|\boldsymbol{\theta})$ is the probability function of a discrete random variable.

3. The expected Fisher information $\mathcal{I}(\cdot)$ always exists, that is:

$$\mathcal{I}(\boldsymbol{\theta}) = -\mathbf{E}\left[\frac{\partial^2}{\partial^2\boldsymbol{\theta}}\log(f(y_i|\boldsymbol{\theta}))\right] = \mathbf{E}\left[\left(\frac{\partial}{\partial\boldsymbol{\theta}}\log(f(y_i|\boldsymbol{\theta}))\right)^2\right] < \infty, \quad \forall\boldsymbol{\theta}\in\Theta.$$

2.3.2. Standard error computation

The variance-covariance matrix of the MLEs is given by

$$\text{Var}\,(\hat{\boldsymbol{\theta}}) = \mathcal{J}^{-1}(\hat{\boldsymbol{\theta}}) = C(\hat{\boldsymbol{\theta}}), \tag{2-6}$$

where $\mathcal{J}(\cdot)$ is the observed Fisher information matrix

$$\mathcal{J}(\hat{\boldsymbol{\theta}}) = \frac{\partial^2}{\partial\boldsymbol{\theta}^2}\log(f(y_i|\boldsymbol{\theta})). \tag{2-7}$$

The standard errors can be calculated as the square root of the diagonal elements of matrix C (Pawitan, 2013b)

$$S.E(\hat{\boldsymbol{\theta}}) = \sqrt{C_{jj}(\hat{\boldsymbol{\theta}})}. \tag{2-8}$$

2.3.3. Properties of the MLEs

MLEs are functionally invariant and they are functions of sufficient statistics. When evaluated on a finite sample size, they have no optimum properties, in particular, they suffer of second order deficiency (Stigler, 2005). However, when the sample size tends to infinity, these estimators are unbiased, consistent, efficient, and asymptotically normal distributed. Let's see in more detail below.

Large sample properties

According to Rossi (2018), if the regularity conditions above-stated in sub-section 2.3.1 are satisfied by $f(\cdot|\boldsymbol{\theta})$, for $\Theta \in \mathbb{R}$, and the sample size tends to infinity, then:

i) $\hat{\boldsymbol{\theta}}$ is an asymptotically unbiased estimator of $\boldsymbol{\theta}$.

ii) $\hat{\boldsymbol{\theta}}$ is MSE-consistent[1] for $\boldsymbol{\theta}$, i.e,

$$\hat{\boldsymbol{\theta}} \xrightarrow{P} \boldsymbol{\theta}.$$

iii) $\hat{\boldsymbol{\theta}}$ is asymptotically normal, i.e.,

$$\sqrt{n}(\hat{\boldsymbol{\theta}} - \boldsymbol{\theta}) \xrightarrow{d} N\left(0, [\mathcal{I}(\boldsymbol{\theta})]^{-1}\right).$$

iv) $\hat{\boldsymbol{\theta}}$ is asymptotically efficient estimator of $\boldsymbol{\theta}$, which means that it achieves the Cramer-Rao lower bound of the variance.

Minimum-variance estimator

Because of the large sample properties i) and iv), the MLE is the UMVUE (uniformly minimum-variance unbiased estimator), which means that

$$\operatorname{Cov}(\hat{\boldsymbol{\theta}}) \geq \left[\mathcal{I}(\hat{\boldsymbol{\theta}})\right]^{-1}.$$

This can be also be verified by using both theorems mentioned below. Let's see it.

Theorem If $Y_1, ..., Y_n$ is a sample of iid random variables with PDF $f(y_i|\boldsymbol{\theta})$ for $\boldsymbol{\theta} \in \Theta$, T is a sufficient statistic for $\boldsymbol{\theta}$ and $\hat{\boldsymbol{\theta}}$ is the unique MLE of $\boldsymbol{\theta}$, then $\hat{\boldsymbol{\theta}}$ is function of the sufficient statistic T (Rossi, 2018).

According to the previous theorem, if $\hat{\boldsymbol{\theta}}$ is a MLE, then its $j-th$ element, $\hat{\theta}_j$, is a function of sufficient statistics. Using this result with the Lehmann-Sheffé theorem (Casella and Berger, 2002), one can conclude that $\hat{\boldsymbol{\theta}}$ is a unique UMVUE of $\boldsymbol{\theta}$.

Invariance property

If $\hat{\boldsymbol{\theta}}$ is the MLE of $\boldsymbol{\theta}$, and the function $g(\boldsymbol{\theta})$ is any transformation of $\boldsymbol{\theta}$, then the MLE of $g(\boldsymbol{\theta})$ is $g(\hat{\boldsymbol{\theta}})$ (Casella and Berger, 2002).

[1] When the MSE (mean square error) of an estimator $\hat{\boldsymbol{\theta}}$ of $\boldsymbol{\theta}$ goes to zero as sample size n goes to infinity, the estimator $\hat{\boldsymbol{\theta}}$ can be made as accurate as desired by taking a sufficiently large sample. In this case, the estimator $\hat{\boldsymbol{\theta}}$ is said to be an MSE-consistent estimator of $\boldsymbol{\theta}$ (Rossi, 2018).

2.4. Numerical Optimization

The following numerical methods are used to solve systems of equations arising from the likelihood maximisation process to estimate parameters.

2.4.1. Newton-Raphson

The procedure of the equation(2-2) is made by means of an optimisation with the first derivative method. If $\nabla l(\boldsymbol{\theta}|\boldsymbol{y})$ is the gradient of the log-likelihood function regarding the parameter vector, then the value that maximizes the log-likelihood function is calculated by solving the following equation system:

$$\nabla l(\boldsymbol{\theta}|\boldsymbol{y}) = \frac{\partial l(\boldsymbol{\theta}|\boldsymbol{y})}{\partial \boldsymbol{\theta}} = \mathbf{0}. \tag{2-9}$$

A first-order Taylor series expansion at $\boldsymbol{\theta}_n$ (in iteration n) for the gradient of the log-likelihood function is equivalent to

$$\nabla l(\boldsymbol{\theta}|\boldsymbol{y}) \approx \nabla l(\boldsymbol{\theta}_n|\boldsymbol{y}) + [\mathbf{H}(\boldsymbol{\theta_n})]\,(\boldsymbol{\theta} - \boldsymbol{\theta}_n)\,, \tag{2-10}$$

where

$$\mathbf{H}(\boldsymbol{\theta}) = -\frac{\partial^2 l(\boldsymbol{\theta}|\boldsymbol{y})}{\partial \boldsymbol{\theta}^2},$$

is the Hessian matrix for the log-likelihood function. To maximise the equation (2-9), the right side of the equation (2-10) must be equal to zero

$$\nabla l(\boldsymbol{\theta}_n|\boldsymbol{y}) + [\mathbf{H}(\boldsymbol{\theta_n})]\,(\boldsymbol{\theta} - \boldsymbol{\theta}_n) = \mathbf{0}. \tag{2-11}$$

By solving $\boldsymbol{\theta}$, the resulting expression is a recurrence formula for iteratively calculating the value of the parameter vector. Because the calculated value approximates the true value in each iteration, $\boldsymbol{\theta}$ is replaced in the recurrence formula by $\boldsymbol{\theta}_{n+1}$, as follows:

$$\boldsymbol{\theta}_{n+1} = \boldsymbol{\theta}_n - [\mathbf{H}(\boldsymbol{\theta_n})]^{-1}\,[\nabla l(\boldsymbol{\theta}_n|\boldsymbol{y})] = \boldsymbol{\theta}_n - [\mathcal{J}(\boldsymbol{\theta_n})]^{-1}\,[\nabla l(\boldsymbol{\theta}_n|\boldsymbol{y})] \tag{2-12}$$

where $\mathcal{J}$, in the statistics context, is called the observed Fisher information. This algorithm is based on quadratic approximations of the log-likelihood function (Lange, 2010, Cap. 11).

2.4.2. Quasi-Newton method

The quasi-Newton method is variation of Newton-Raphson algorithm which can be used in optimisation problems. In the context of log-likelihood maximisation, its recursive formula is as follows:

$$\boldsymbol{\theta}_{n+1} = \boldsymbol{\theta}_n - \left[\mathbf{B}(\boldsymbol{\theta}_n)\right]^{-1} \left[\nabla l(\boldsymbol{\theta}_n|\boldsymbol{y})\right], \tag{2-13}$$

where $\nabla l(\boldsymbol{\theta}_n)$ is the gradient of the objective function evaluated at $\boldsymbol{\theta}_n$ and $\mathbf{B}(\boldsymbol{\theta}_n)$ is the approximated Hessian matrix (Fletcher, 1987). The family of quasi-Newton methods always relies on a starting set of parameters $\boldsymbol{\theta}_0$, and an initial approximate Hessian matrix, which is set depending on the implementation. For example, the `BFGS` (Broyden-Fletcher-Goldfarb-Shanno) uses an identity matrix as the initial approximate inverse Hessian, so the first search direction is steepest descents (Nash, 2014b, Cap. 2). Update of approximate Hessian $\mathbf{B}(\boldsymbol{\theta}_n)$ is calculated by gradient evaluations

$$\mathbf{B_{n+1}}(\boldsymbol{\theta}_{n+1} - \boldsymbol{\theta}_n) = \nabla l(\boldsymbol{\theta}_{n+1}|\boldsymbol{y}) - \nabla l(\boldsymbol{\theta}_n|\boldsymbol{y}). \tag{2-14}$$

`L-BFGS-B` method is a limited-memory version of `BFGS` that undertakes problems with simple box constraints (Byrd et al., 1995).

2.4.3. Nonlinear conjugate gradient method

This is another family of gradient optimisers. This method is useful when minimising an approximately quadratic form with the Hessian as a constant positive definite matrix (Nash, 2014a, Cap. 2). The algorithms have the following steps.

Step 1 - Initiation: step 1 provides the starting parameters $\boldsymbol{\theta}_0$ and compute $l(\boldsymbol{\theta}_0)$ and $\nabla l(\boldsymbol{\theta}_0)$. The initial search direction is the negative gradient, thus:

$$\Delta\boldsymbol{\theta}_0 = -\nabla l(\boldsymbol{\theta}_0), \tag{2-15}$$

and determine the step length using a line search

$$\alpha_0 := \arg\min_{\alpha} \, l(\boldsymbol{\theta}_0 + \alpha\Delta\boldsymbol{\theta}_0), \tag{2-16}$$

$$\Delta\boldsymbol{\theta}_1 = \Delta\boldsymbol{\theta}_0 + \alpha_0\Delta\boldsymbol{\theta}_0. \tag{2-17}$$

Step 2 - Iteration: step 2 computes the gradient at the new point and uses it

$$\Delta\boldsymbol{\theta}_n = -\nabla l(\boldsymbol{\theta}_n), \tag{2-18}$$

Then, compute the new direction as follows:

$$\boldsymbol{s_n} = \Delta\boldsymbol{\theta}_0 + \beta_n \boldsymbol{s_{n-1}}, \tag{2-19}$$

where β_n is a coefficient that determines the search direction. There are several formulas to compute it (visit Fletcher (1964); Dai and Yuan (1999); Wang and Li (2011) for further information). Finally, another line search must be performed, thus,

$$\alpha_n = \arg\min_{\alpha}\, l(\boldsymbol{\theta}_n + \alpha_{n-1} s_n), \tag{2-20}$$

and the new position is updated

$$\boldsymbol{\theta}_{n+1} = \Delta\boldsymbol{\theta}_n + \alpha_n \boldsymbol{s_n}. \tag{2-21}$$

Step 3 - Termination: The algorithm terminates when no progress is made after a direction reset or when some tolerance criterion is reached.

2.4.4. Nelder-Mead algorithm

The Nelder-Mead algorithm is a direct search simplex-based iterative method devised to solve unconstrained and box-constrained minimisation problems. According to Nelder and Mead (1965), consider a set of $k+1$ vertices in the parameter space $\{\theta_j\} \in \mathbb{R}^k$, with $j = 0, 1, ..., k$ the points of a simplex S, and its corresponding function values $l(\theta_j) = f_j$. Let be,

$$f_h = \max_j (f_j), \tag{2-22}$$

$$f_l = \min_j (f_j), \tag{2-23}$$

the maximum and minimum values for the objective function and

$$\bar{\theta} = \frac{1}{n} \sum_{j \neq h} \theta_j, \tag{2-24}$$

is the centroid of the points. The procedure involves the following steps (Singer and Nelder, 2009).

Initial *Simplex*: The method begins with a *non-degenerate simplex*. This condition is satisfied when the points $\{\theta_0, \theta_1, ..., \theta_n\}$ do not lie in the same hyperplane.

Transformations: The method applies some transformations in each step (reflection, contraction and expansion) to the vertex with a greater function value in each step to decrease it.

- **Reflection:** the transformation is performed using the following expression:

  $$\theta_R = \bar{\theta} + \alpha_1(\bar{\theta} - \theta_h). \tag{2-25}$$

  It is an attempt to move away the simplex from the highest point θ_h, replacing it with the point θ_R. α_1 is a positive constant called *reflection coefficient*. If f_R lies between f_h and f_l, then θ_h is replaced by θ_R and we start again with the new simplex.

- **Expansion:** if $f_R < f_l$, then θ_R is transformed as follows

  $$\theta_E = \bar{\theta} + \alpha_2(\theta_R - \bar{\theta}). \tag{2-26}$$

  This is a movement a little bit more away from θ_h (the old one). $\alpha_2 > 1$ is another constant called *expansion coefficient*. If $f_E < f_l$, θ_h is replaced by θ_E, otherwise, θ_h is replaced by θ_R.

- **Contraction:** if $f_R > f_l \, \forall i \neq h$, then we define a new θ_h to be either the old θ_h or θ_R, whichever has the lower value of f, and we apply the following transformation

  $$\theta_C = \bar{\theta} + \alpha_3(\theta_h - \bar{\theta}). \tag{2-27}$$

  This is a movement in opposite direction of the previous exploration made with reflection transformation. $0 < \alpha_3 < 1$ is the *contraction coefficient*. We replace θ_h with θ_C unless $f_C > \text{mín}(f_h, f_R)$; in this case, all θ_i are replaced by $(\theta_i + \theta_l)/2$ and restart the process.

Termination: The algorithm ends when the standard error of the function values at the vertices of simplex is less than a tolerance value, or when iterations number is reached, and the solution is given by the centroid of the last simplex.

2.4.5. Implementations in R

R (R Core Team, 2023) is a free language developed for statistical computing and equipped with unconstrained and box-constrained general-purpose optimization tools. Maximum likelihood estimation can be performed with these tools only if users implements the (log-)likelihood as the objective function themselves.

Implementations of the aforementioned optimization tools are in **stats** package: Nash (1979) implemented `optim`, a function that performs optimization based on three algorithms: (1) Nelder and Mead (1965), (2) quasi-Newton (`BFGS`) and (3) conjugate-gradient (`CG`); Fox et al. (1978) developed the function `nlminb` for optimization using PORT (portable Fortran programs for numerical computation) routines.

Improvements in `CG` implementation in `optim` where made by Dai and Yuan (2001), which was implemented in **Rcgmin** (Nash, 2014c) for bounded and unbounded minimization problems. Moreover, Nielsen and Mortensen (2016) developed the package **ucminf**, with an alternative implementation of unbounded `BFGS`, updating of the inverse Hessian and soft line search with a trust region type monitoring of the input to the line search algorithm. Nash (2018) reimplemented this method in **Rvmmin** package in order to enable constrained minimization.

2.5. Survival analysis and regression models

Lifetime distributions themselves are an improvement respect to classical distributions because they allows to model different situations with a good fitting. To use them, it is enough to have good estimation algorithms. However, estimation with no covariates is not always suitable to analyse the results from clinical trials in survival analysis or accelerated life test for reliability. In such situations, the inclusion of explanatory variables is necessary. From the classic statistics approach, modelling proposals have been made based on regression analysis.

In this section, we present relevant proposals referring to computational routine processes for parameter estimation, as well as modelling proposals developed in the framework of the regression models.

2.5.1. Accelerated failure time models

These models have a particular interest to adjust data of accelerated failure time tests. Its mathematical form arises from the assumption that the experimental units associated with the treatment i are different from those associated with the treatment j in terms of survival by a factor or acceleration rate ϕ (Kalbfleisch and Prentice, 2002)

$$S_i(t) = S_j(\phi t), \tag{2-28}$$

where for the hazard function and for the probability density function, the following is fulfilled:

$$h_i(t) = \phi h_j(\phi t), \quad f_i(t) = \phi f_j(\phi t), \tag{2-29}$$

where $h(\cdot)$ is the hazard function and $f(\cdot)$ is the probability density function. The final result is an ordinary regression model for the logarithm of the survival times, as follows:

$$\log T_i = \boldsymbol{X}^\top \boldsymbol{\beta} + \sigma \epsilon_i. \tag{2-30}$$

The distribution of the residuals ϵ_i depends on the distribution of T. In reliability engineering, it is common to work with the distributions of Weibull, generalized gamma, log-normal, and log-logistic, which correspond to residuals that include Gumbel, normal, and logistic distributions, respectively. However, this approach has also been used to create regression models of families that are generalisations of the traditional ones, such as the case of the log-generalized modified Weibull distribution (Ortega et al., 2011).

2.5.2. Distributional regression models

A more recent approach to model data coming from experiments in survival analysis consists of adjusting the parameters of a distribution selected in terms of explanatory variables. This strategy was adopted as an alternative to model data from an accelerated failure time tests of a high-strength alloy. These data were used to model the scale parameter and the Weibull distribution shape parameter. Although the effects of the covariates were not significant for the shape parameter, their use improved the predictions of the model (Escobar and Meeker, 2006).

Another development in this direction is related to GAMLSS (*Generalized Additive Models for Location, Shape and Scale*) by Rigby and Stasinopoulos (2005). Within this framework, it is possible to implement any distribution that has four parameters or less and estimate its parameters in terms of covariates from a penalised log-likelihood function. This proposal includes the creation of **gamlss** package (Stasinopoulos and Rigby, 2007). It is possible to perform parameter estimation of any distribution implemented as a `gamlss.family` structure. Visit Stasinopoulos et al. (2017b) for more details.

Finally, the regression models for the exponentiated Weibull family with the non-constant shape parameter stand out, whose estimations are obtained via maximum likelihood (Mazucheli et al., 2013) was developed under the GAMLSS framework.

3. Total Time on Test (TTT) Transform

> ” *It is a capital mistake to theorize before one has data. Insensibly one begins to twist facts to suit theories, instead of theories to suit facts.*
>
> — **Sherlock Holmes**
>
> (The best detective)

Estimation of parameters via maximum likelihood estimation has been detailed in the previous chapter. This technique and other recurrent techniques in statistics and machine learning are based on numerical methods for optimisation, i.e, these solutions are not closed and are obtained using computational algorithms.

Sometimes it is useful to perform a deep EDA (exploratory data analysis) in order to get insights for guiding numerical search of parameters. The empirical total time on test (TTT) transform is frequently used for investigating the hazard shape corresponding to a given dataset. This is particularly useful for flexible distributions (Cooray [2006], Panja and Ray [2007], Granzotto et al. [2018], Abdi et al. [2019], among others).

In this chapter we show the implementation of TTT plot in R in order to enhance MLE.

3.1. TTT Transform

Let $T_0 \leq T_1 \leq \cdots \leq T_n$ a complete ordered sample of times to failure from n identical nonrepairable[1] units with distribution $F(t)$ and survival function $S(t) = 1 - F(t)$ (Bergman and Klefsjo, 1984). The TTT transform $H_{F^{-1}}(u)$ can be defined as

$$H_{F^{-1}}(u) = \int_0^{F^{-1}(u)} S(t)dt, \quad 0 \leq u \leq 1. \tag{3-1}$$

Example 3.1.1. *Exponential distribution (Nachlas, 2017)*

[1] In other words, we measure the time to a specific event that occurs, which is a single, irreversible transition. The event in question may be death, disease occurrence, disease recurrence or recovery in a medical context, churning in a business context, or machine failure in an engineering context.

Let's recall the exponential lifetime distribution with rate parameter ρ

$$f(t|\rho) = \rho \exp(-\rho t), \qquad t > 0, \tag{3-2}$$

with $\rho > 0$*. If a population of devices has this distribution with* $\rho = 0.84$*, then* $F(0.75) = 0.467$*. To illustrate how to calculate the TTT transform, we compute it at* $u = 0.467$

$$H_{F^{-1}}(u = 0.467) = \int_0^{F^{-1}(0.467)} S(t)dt = \int_0^{0.75} \exp(-\rho t)dt \quad = \frac{1 - \exp(-0.75\rho)}{\rho} = \frac{0.467}{0.84} = 0.556$$

□

3.1.1. TTT transform properties

The TTT transform has two important properties that can be used to derive the criteria for characterising the hazard function shape.

Property 1 - Relation with the mean When $u = 1$, TTT transform yields to the mean of the random variable, i.e

$$H_{F^{-1}}(u = 1) = \int_0^{F^{-1}(u=1)} S(t)dt = \int_0^{\infty} S(t)dt = \mathbf{E}(T), \tag{3-3}$$

where T is a positive random variable that follows a lifetime distribution. The relationship between the integral of the survival function and the expected value is proved in the Appendix A.

Property 2 - Derivative of the TTT transform The derivative of the TTT transform evaluated at any point in the cumulative failure probability is equivalent to the reciprocal of the corresponding value of the hazard function,

$$\frac{d}{du} H_{F^{-1}}(u) = \frac{1}{h(t)}. \tag{3-4}$$

Further details about the proof can be found in the Appendix B.

In survival analysis, it is common to address the scaled TTT transform $\phi(u)$, whose expression is given by

$$\phi(u) = \frac{H_{F^{-1}}(u)}{H_F^{-1}(1)}, \tag{3-5}$$

where

$$H_{F^{-1}}(1) = \int_0^\infty S(t)dt. \tag{3-6}$$

3.2. TTT plot

The sample counterpart of the scaled TTT transform is the TTT-statistic, also known as empirical TTT, given by the following expression:

$$\phi_n\left(\frac{i}{n}\right) = \frac{H_{F^{-1}}(i/n)}{H_{F^{-1}}(1)}. \tag{3-7}$$

For non-censored data the scaled empirical TTT can be computed from the empirical probability function. The resulting expression is the Barlow estimator (Aarset, 1987), given by

$$\phi_n\left(\frac{i}{n}\right) = \frac{\left(\sum\limits_{j=1}^{i} T_{(j)}\right) + (n-i)T_{(i)}}{\sum\limits_{i=j}^{n} T_j}, \tag{3-8}$$

where $T_{(i)}$ is the r^{th} order statistic, with $T_{(0)} = 0$, $r = 1, 2, \ldots, n$, and n being the sample size. For censored data, scaled empirical TTT can be computed using the expression of Kaplan-Meier survival estimator based on expressions Westberg and Klefsjö (1994).

$$\phi_n\left(\frac{i}{n}\right) = \frac{\sum\limits_{j=1}^{i}\left[\prod\limits_{k=1}^{j}\left(1-\frac{b_k}{c_k}\right)\right]\left(T_{(j)}-T_{(j-1)}\right)}{\sum\limits_{j=1}^{n}\left[\prod\limits_{k=1}^{j}\left(1-\frac{b_k}{c_k}\right)(T_j - T_{j-1})\right]}, \tag{3-9}$$

where b_k is the number of events and c_k is the number of units at risk until time k. In appendix Appendix C we show how to obtain equation 3-9.

It is interesting to recall that the TTT plot (plot of empirical TTT transform vs. empirical probability) can be used as an exploratory tool. Using expression 3-8 or 3-9 the values of $\frac{i}{n}$ and $\phi_n\left(\frac{i}{n}\right)$ are calculated and a scatter diagram is plotted with them, which gives rise to the TTT plot. It is possible to identify hazard function shapes described by data as decreasing, increasing, bathtub, and unimodal if the shape of the points (or curve) obtained in TTT transform is convex, concave, convex then concave, and concave then convex, respectively.

3.2.1. Illustration

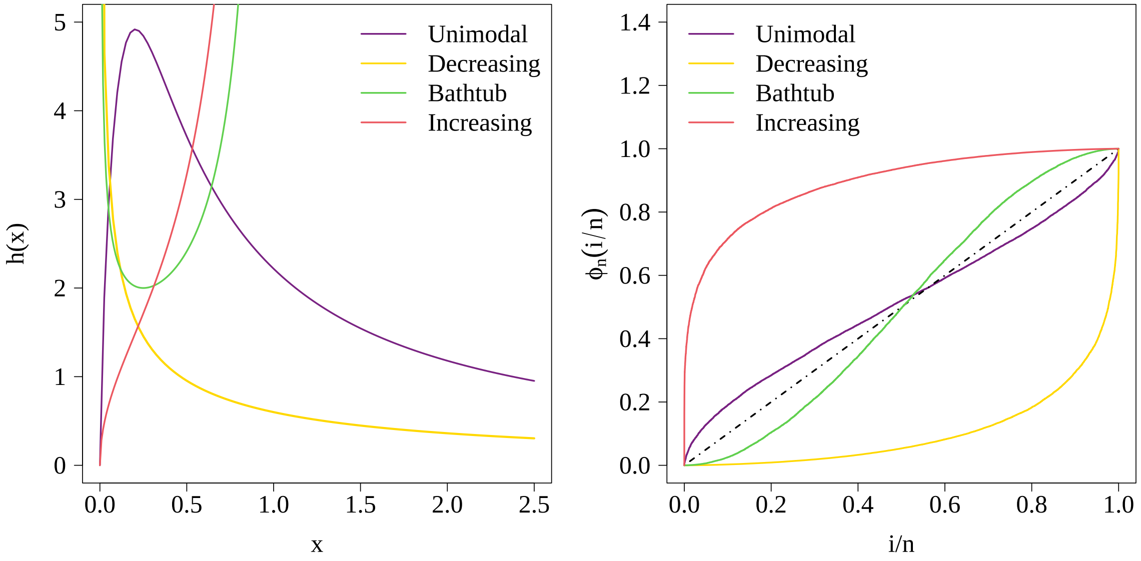
Consider the Generalized Weibull family (GWF) by Mudholkar and Kollia (1994). The PDF is

$$f(x|\mu,\sigma,\nu) = \mu\sigma x^{\sigma-1}\left(1-\mu\nu x^{\sigma}\right)^{\frac{1}{\nu}-1}, \qquad t>0, \tag{3-10}$$

with μ, $\sigma > 0$ and $-\infty < \nu < \infty$. The CDF is

$$F(x|\mu,\sigma,\nu) = 1-\left(1-\mu\nu q^{\sigma}\right)^{\frac{1}{\nu}}, \tag{3-11}$$

and the hazard rate function is



(a) Hazard functions.

(b) TTT plots.

Figure 3-1: Different hazard function shapes (left) and their corresponding TTT plots shapes (right) with simulated data using the Generaized Weibull Family (sample size of 20000 in each case). Umimodal shape correposponds to $GWF(\mu = 15,\ \sigma = 1.7,\ \nu = -0.7)$; decreasing shape to $GWF(\mu = 2,\ \sigma = 0.5,\ \nu = -2)$; bathtub shape to $GWF(\mu = 1,\ \sigma = 0.5,\ \nu = 1)$ and incresing shape to $GWF(\mu = 2.1,\ \sigma = 5,\ \nu = 0.5)$.

$$h(x|\mu,\sigma,\nu) = \mu\sigma x^{\sigma-1}\left(1-\mu\nu x^{\sigma}\right)^{-1}. \tag{3-12}$$

Some examples of TTT plots and their corresponding hazard functions are illustrated in figure **3-1**. We simulated the data using the GWF and the Barlow estimator from equation 3-8.

3.3. Implementation and Features

We implemented the TTT transform in the `TTTE_Analytical` function within the **EstimationTools** package. Our implementation extends the approach established by (Marinho et al., 2019) in the **AdequacyModel** package, going beyond the mere application of the Barlow expression detailed in equation (3-8). We also integrated the application of equation (3-9), thus providing a more comprehensive analytical tool. Furthermore, `TTTE_Analytical` generates a S3 object of class `EmpiricalTTT`. The function can be used to define their main arguments.

```
TTTE_Analytical(formula, scaled, method)
```

where `formula` is a formula object of R defining the response variable to be explored and `scaled` is used for obtain the scaled version. The `method = "Barlow"` is used for computing equation (3-8) (Barlow method), and `method = "censored"` to compute equation (3-9) (Kaplan-Meier estimator).

The corresponding `plot` method to obtain the TTT plot is also implemented in **EstimationTools**

```
plot(x, add)
```

where `x` is an `EmpiricalTTT` object, and `add` is used to add overlapping plots in the same device. The source codes can be found in the empirical TTT transform script and the TTT plot script from **EstimationTools** package repository.

Finally, we also implemented `TTT_hazard_shape`. This is a wrapper for `TTTE_Analytical` which computes the empirical TTT and performs a LOESS (Local Estimated Scatterplot Smoothing) non-parametric estimation to determine the convexity of the transform throughout the `loess()` R function. The function can be executed by defining at least an R `formula` object

```
TTT_hazard_shape(formula)
```

Alternatively, it can be passed an `EmpircalTTT` object.

```
TTT_hazard_shape(object)
```

The output is an object of `HazardShape` class with its corresponding `plot` and `print` methods. In **EstimationTools** package can also be found the following scripts:

- `TTT_hazard_shape.R` script for the wrapper.
- `plot.HazardShape.R` script for the graphical method.
- `print.HazardShape.R` script for printing the description of the hazard function shape.

3.4. Application examples

In this section, we explore the usage of `TTTE_Analytical` and `TTT_hazard_shape` with two real datasets.

3.4.1. Reduction cells data

The dataset in table **3-1** contains the failure ages (in units of 1000 days) of 20 aluminium-reduction cells of a standard design.

0.468	0.725	0.838	0.853	0.965	1.554	1.658	1.764	1.776	1.139	1.990
1.142	2.010	1.304	1.317	2.224	$2.279^{\dagger}$	1.427	$2.244^{\dagger}$	$2.286^{\dagger}$		

Table 3-1: Time to failure of 20 reduction cells (Whitmore, 1983).[2]

Whitmore (1983) fitted an inverse Gaussian distribution (IG) with the following PDF

$$f(t|\mu,\varphi) = \sqrt{\frac{1}{(2\pi\varphi t^3)}}\exp\left(\frac{-(t-\mu)^2}{2\mu^2\varphi t}\right), \quad t>0, \qquad (3\text{-}13)$$

where $\mu, \varphi > 0$ are the scale and shape parameters, respectively. The dataset can be recovered from **EstimationTools** package.

```
library(EstimationTools)
data("reduction_cells")
head(reduction_cells, n = 3)

##    days status
## 1 0.468      1
```

[2] Censored datum indicated with †.

```
## 2 0.725      1
## 3 0.838      1
```

The TTT transform can be computed as follows:

```
TTT_IG <- TTTE_Analytical(Surv(days, status) ~ 1, data = reduction_cells,
                          method = ’censored’)
i_n <- TTT_IG$‘i/n‘
phi_n <- TTT_IG$phi_n
head(data.frame(i_n, phi_n), n = 3)
```

```
##    i_n     phi_n
## 1 0.00 0.0000000
## 2 0.05 0.3199942
## 3 0.10 0.4864690
```

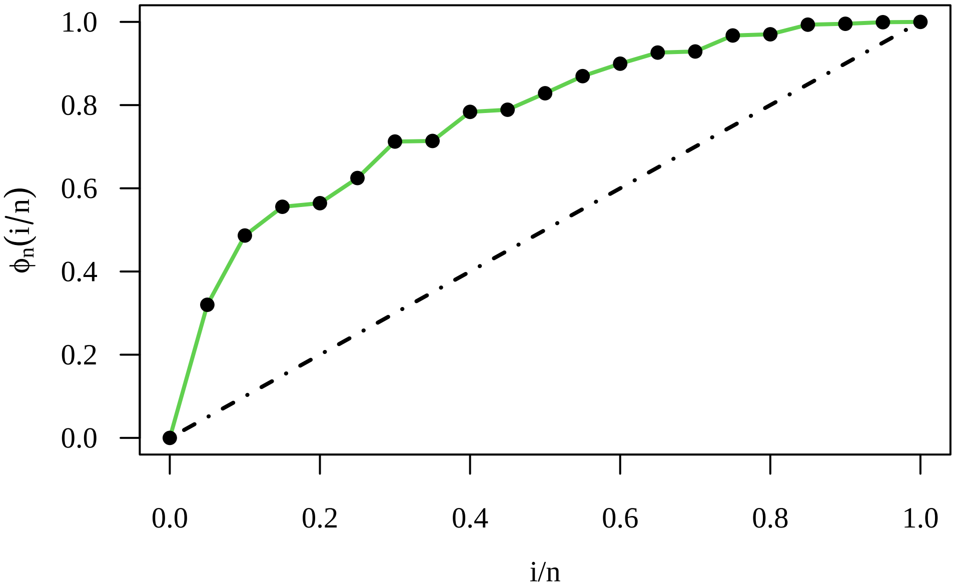


Figure 3-2: TTT plot for the reduction cells data.

The plot in figure **3-2** is obtained using the following command lines:

```
plot(TTT_IG, type = "l", las = 1, lwd = 3, col = 3)
plot(TTT_IG, type = "p", pch = 16, add = TRUE)
```

The comparison between figure **3-2** with patterns observed in figure **3-1**b suggests an increasing hazard shape. `TTT_hazard_shape` helps to confirm this behaviour with the `print()` method.

```
HS_IG <- TTT_hazard_shape(TTT_IG, data = reduction_cells)
print(HS_IG)

## ------------------------------------------------------------
## Hazard shape: Increasing
## ------------------------------------------------------------

## Warning in print.HazardShape(HS_IG): Non-parametric estimate for Empirical
TTT is irregular.
## Please, use the 'plot()' method to see the TTT shape
```

The empirical TTTs depicted in **3-1**b appear as smooth functions, which can be attributed to the large sample size used in their plotting ($n = 20000$). Conversely, the example at hand uses a much smaller sample size ($n = 20$), which does not guarantee that the empirical TTT transform will possess a similarly smooth shape. Consequently, the LOESS estimate derived from this smaller sample size exhibits irregularities.

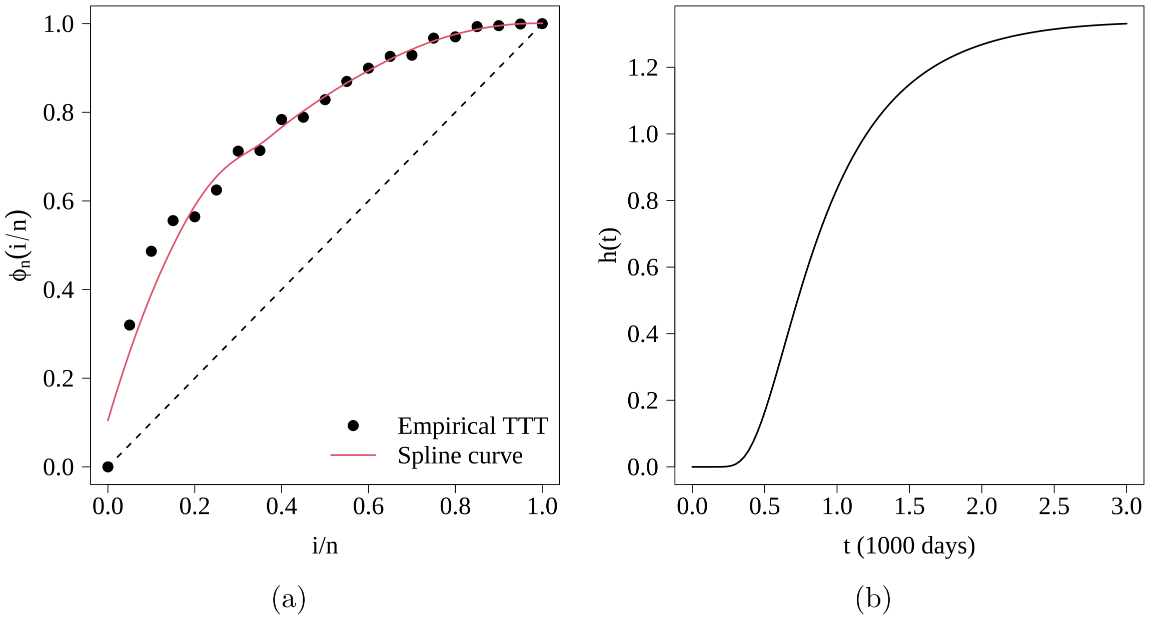


Figure 3-3: Comparison between TTT plot and estimated hazard function for reduction cells data. (a) TTT plot with its LOESS estimate (with default `loess()` R parameters) obtained with `plot()` method applied on `HS_IG` object; (b) Inverse Gaussian hazard function with $\hat{\mu} = 1.61$ and $\hat{\varphi} = 5.96$.

The LOESS estimation is performed by `TTT_hazard_shape` under the hood using `stats::loess` with the default parameters (`span = 0.75` and `degree = 2`). It can be plotted just using the `plot` method for `HazardShape` objects. The TTT plot can be observed in figure **3-3**a alongside the actual hazard function. It is a matter of typing `plot(HS_IG)` in the R console.

Whitmore (1983) also found that the maximum likelihood estimators are $\hat{\mu} = 1.61$ and $\hat{\varphi} = 5.96$. In fact, these estimated parameters correspond to an increasing IG hazard function (see figure **3-3**b).

3.4.2. Tensile strength data

The data presented in Table **3-2** correspond to the tensile strength T (in GPa) of 69 specimens of carbon fiber tested under tension at gauge lengths of 20 mm.

1.312	1.314	1.479	1.552	1.700	1.803	1.861	1.865	1.944	1.958	1.966
1.997	2.006	2.027	2.055	2.063	2.098	2.14	2.179	2.224	2.240	2.253
2.270	2.272	2.274	2.301	2.301	2.359	2.382	2.382	2.426	2.434	2.435
2.478	2.490	2.511	2.514	2.535	2.554	2.566	2.57	2.586	2.629	2.633
2.642	2.648	2.684	2.697	2.726	2.770	2.773	2.800	2.809	2.818	2.821
2.848	2.88	2.954	3.012	3.067	3.084	3.090	3.096	3.128	3.233	3.433
3.585	3.585									

Table 3-2: Tensile strength of 69 fibers (Devendra and Rangaswamy, 2013).

Ghitany et al. (2013) fitted their power Lindley (PL) distribution with the following PDF:

$$f(u|\mu,\sigma) = \frac{\mu\sigma^2}{\sigma+1}(1+u^\mu)u^{\mu-1}e^{-\sigma u^\mu}, \quad u > 0, \tag{3-14}$$

where $\mu, \sigma > 0$. This dataset can also be recovered from **EstimationTools** package.

```
data("Fibers")
head(Fibers, n = 3)
```

```
##   Strenght
## 1    1.312
## 2    1.314
## 3    1.479
```

As before, we can compute the TTT transform using `TTTE_Analytical`

```
TTT_PL <- TTTE_Analytical(Strenght ~ 1, data = Fibers)
i_n <- TTT_PL$`i/n`
phi_n <- TTT_PL$phi_n
head(data.frame(i_n, phi_n), n = 3)
```

```
##          i_n     phi_n
## 1 0.00000000 0.0000000
## 2 0.01449275 0.5352189
## 3 0.02898551 0.5360230
```

and obtain the TTT plot (see figure **3-4**)

```
plot(TTT_PL, type = 'l', las = 1, lwd = 3, col = 3)
plot(TTT_PL, type = 'p', pch = 16, add = TRUE)
```

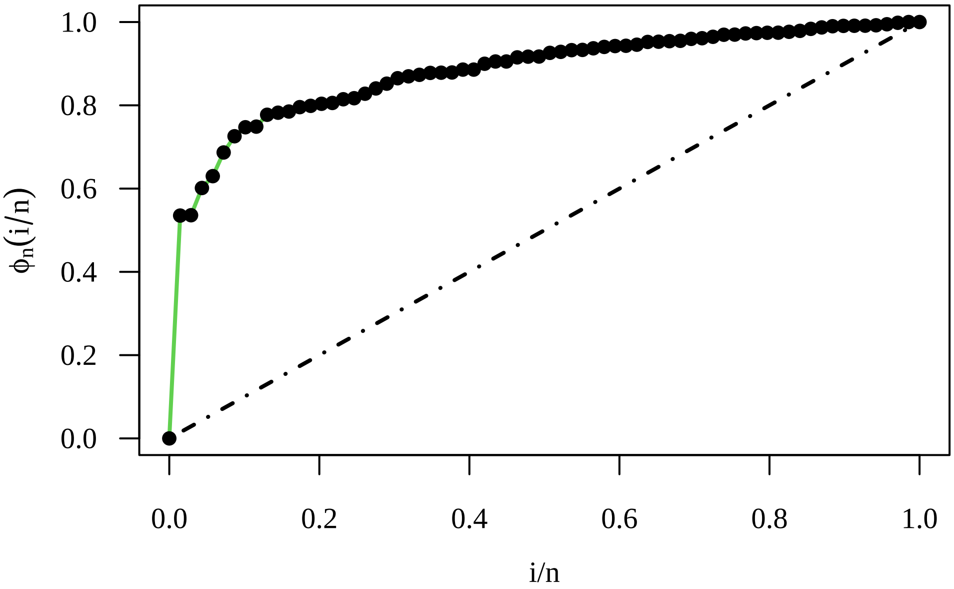
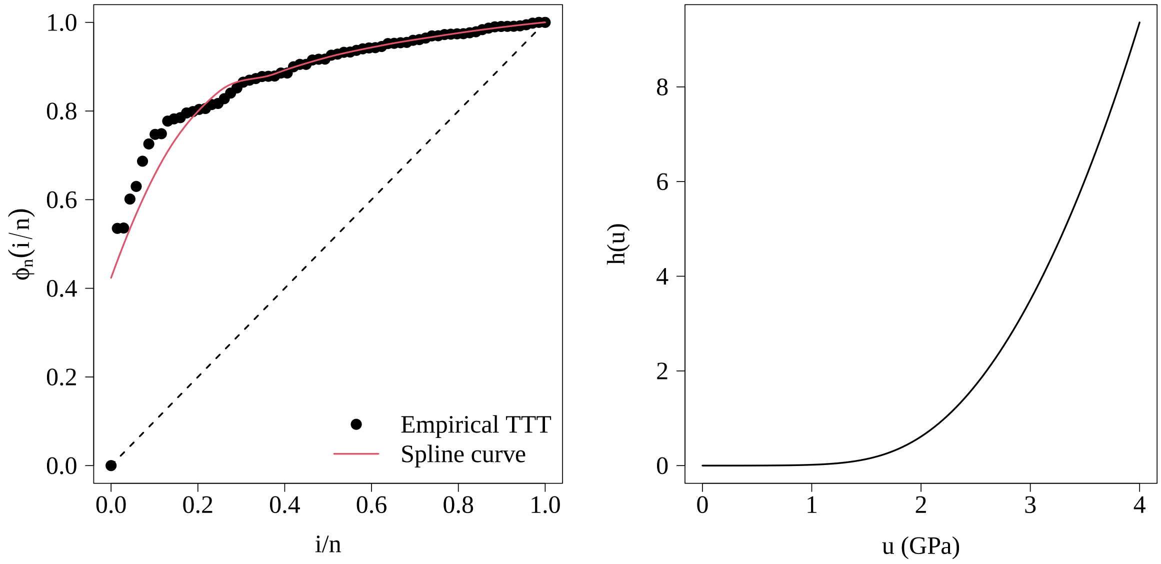


Figure 3-4: TTT plot for the tensile strenght data.



(a)

(b)

Figure 3-5: Comparison between TTT and estimated hazard function plots for tensile strength data. (a) TTT plot with its LOESS estimate; (b) Power Lindley hazard function with $\hat{\mu} = 3.8678$ and $\hat{\sigma} = 0.0497$.

Estimates by Ghitany et al. (2013) ($\hat{\mu} = 3.8678$ and $\hat{\sigma} = 0.0497$) corresponds to an increasing PL hazard function.

```
HS_PL <- TTT_hazard_shape(TTT_PL, data = Fibers)
print(HS_PL)

## --------------------------------------------------------------------
## Hazard shape: Increasing
## --------------------------------------------------------------------
```

As expected, `TTT_hazard_shape` points an increasing hazard shape, as can be seen in figure **3-5**.

3.5. Conclusion

We have implemented the empirical TTT transform and the TTT plot in separate computational routines in order to exploit its capabilities as an exploratory tool and provide potential users with flexibility because they can use our `plot` method or choose their own graphical representations. We also implement the Kaplan-Meier estimator in order to compute the transform for censored data. Finally, we have reviewed two application examples in which we have observed the usage and demonstrated the performance of our implementation.

4. Floating-point issues and High-precision arithmetic

> ” *People think of these eureka moments and my feeling is that they tend to be little things, a little realisation and then a little realisation built on that.*
>
> — **Roger Penrose**
>
> (Mathematician, physicist and Nobel Prize)

All real numbers in the R environment are stored as numeric objects with double precision format (R Core Team, 2022). This means R works with 64-bits floating-point arithmetic, which has been producing results of sufficient accuracy in a great variety of science applications. Despite using 64-bit arithmetic, many scientists and engineers who conduct extensive computations have realised, to their disappointment, that numerical challenges have arisen because of the rapidly growing scale of their computations, which raises doubts about the precision of their results (Bailey and Borwein, 2015).

High-precision computation plays a key role in modern statistics because of the increased complexity of statistical analyses. Numerical errors can significantly impact on the accuracy of the results in many computational statistics implementations. High-precision arithmetic is a crucial component for ensuring the accuracy of numerical optimisation algorithms, which are widely used in statistical inference. One noteworthy example is the exact computation of p-values for the Friedman rank sums test. This advancement allows a more accurate calculation of p-values, especially in the tails of the distribution, where large-sample approximations can be inaccurate. As highlighted by Eisinga et al. (2017), it has the potential to improve the reliability of the Friedman rank sums test.

The problem has been addressed by some authors from the R community. They have implemented packages that provide APIs (Application Programming Interfaces) to set the precision of numerical objects. The **bignum** package (Hall, 2021) provides arbitrary precision arithmetic for integers, rational numbers, and floating-point numbers. In the same direction, the **gmp** package (Lucas et al., 2023) is also a well-known tool that performs high-precision arithmetic operations using the GNU Multiple Precision (GMP) library as back-end and

Rmpfr package (Mächler, 2022) is another alternative that uses the GNU MPFR C library.

4.1. The Rmpfr package

The **Rmpfr** package is a valuable tool for performing high-precision computations in the R environment. It allows access to multiprecision arithmetic, also known as arbitrary precision arithmetic. The package offers various functions for basic arithmetic, logical operations, transcendental functions, and various mathematical functions. Unlike other alternatives, **Rmpfr** is preferred because it is both precise and easy to use, and it is compatible with other R packages. It is also effective and stable, meaning that it can be used in combination with other tools for data analysis and visualisation In addition, the package is efficient and fast, thanks to optimised algorithms for performing arithmetic operations on high-precision numbers and parallel computations (Mächler, 2011). With a large community behind it and regular updates, **Rmpfr** is a mature package. Moreover, there are useful packages like **numbers** that use **Rmpfr** and provide number-theoretic functions for factorisation prime numbers, twin primes, primitive roots, modular logarithm and inverses, extended GCD, Farey series, and continued fractions (Borchers, 2022).

The real potential of **Rmpfr** is completely unleashed if higher-precision numerical objects and high-precision arithmetic algorithms are used to compute functions. A particularly interesting case regarding the latter in applied mathematics and especially in statistics is the concern with $\log(1+x)$ and $\exp(x)-1$ functions. They are fundamental constituents of beta, gamma (Didonato and Morris, 1992), exponential, Weibull, t, logistic, geometric, and hypergeometric distributions, and even for the logit link function in logistic regression. It is known that the direct calculation of the first produces severe cancellation for $|x| \ll 1$. Similarly, the latter is partially cancelled by -1 for $|x| \ll 1$. Therefore, the alternative algorithms $\text{log1p}(x)$ and $\text{expm1}(x)$ were proposed and included in the C libraries and the C standards (IEEE and Open Group, 2004b,a). They use only a few terms of the Taylor series expansion (Tang, 1990, 1992)

$$\text{expm1}(x) = \exp(x) - 1 = x + \frac{x^2}{2!} + \frac{x^3}{3!} + \dots, \text{ for } |x| < 1, \tag{4-1}$$

$$\text{log1p}(x) = \log(1+x) = x - \frac{x^2}{2} + \frac{x^3}{3} - \dots, \text{ for } |x| < 1. \tag{4-2}$$

Consider the following mathematical function, which is commonly used in the terms of various novel distributions:

$$g_p = \log(1 - \exp(a_p)), \quad p \in \{dp, hp\}. \tag{4-3}$$

Here, *dp* and *hp* are subscripts that indicate the usage of double-precision (64 bits) and high-precision (1024 bits) arithmetic, respectively. Additionally, consider the relative error ε, which defined as follows:

$$\varepsilon = 1 - \frac{g_{dp}}{g_{hp}}. \tag{4-4}$$

Mächler (2012) computed the metric above using `mpfr()` function for the 1024-bits numbers to demonstrate that the algorithms implemented from equations from (4-1) are not sufficient to compute $\log(1 - \exp(x))$ and $\log(1 + \exp(x))$ (see figure **4-1**).

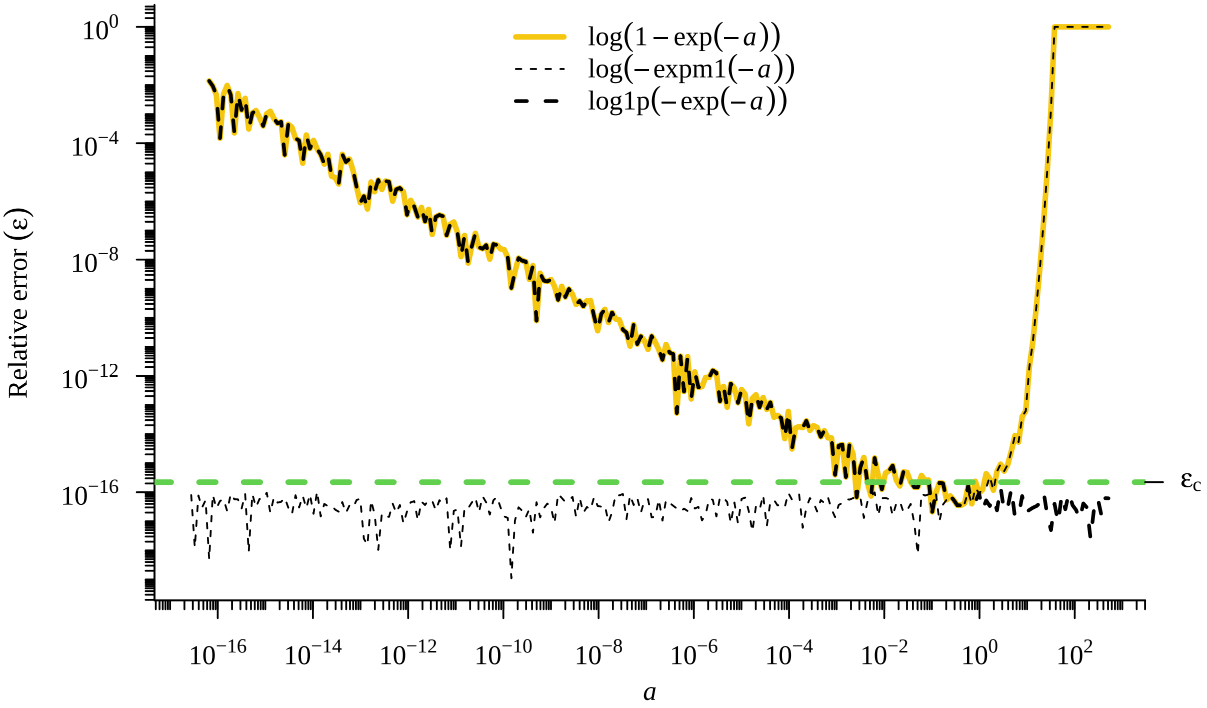


Figure 4-1: Relative error (ε) of the default, $\log(1-\exp(a))$, and the two methods `expm1()` and `log1p()` because of the usage of double-precision compared with the usage of high-precision arithmetic (Mächler, 2012). ε_c (green dashed line) is the smallest positive floating-point number in R (normally `2.220446e-16`).

Khan (2018) and Muhammad et al. (2021) required to implement a combination of `log1p`–`expm1` functions with high-precision numbers using `mpfr()` function to obtain successful maximum likelihood estimation routines. The use of standard floating-point arithmetic in these scenarios can result in significant numerical errors, thereby undermining the accuracy or even the convergence of statistical models.

In the next section, we show the positive impact of using high-precision arithmetic algorithms such `expm1` or `log1p` alongside with high-precision numerical objects with the `mpfr()` function in writing distributions in R.

4.2. Illustrative examples

In this section, we select two relevant reliability distributions whose estimation procedures benefit from high-precision arithmetic practises

4.2.1. Odd Weibull distribution

The odd Weibull (OW) distribution, proposed by Cooray (2006), which can model a bathtub-shaped failure rate, is a more versatile version of the standard Weibull distribution. Further information can be found in section 8.1. The PDF is given by

$$\begin{aligned} f_{OW}(x|\mu,\sigma,\nu) = \left(\frac{\sigma\nu}{x}\right)(\mu x)^{\sigma}\exp(\mu x)^{\sigma}\left(\exp(\mu x)^{\sigma}-1\right)^{\nu-1}\left[1+\left(\exp(\mu x)^{\sigma}-1\right)^{\nu}\right]^{-2}, \\ x>0,\ \mu,\ \sigma,\ \nu>0. \end{aligned} \tag{4-5}$$

Let's implement the PDF using $\log(f_{OW}(x))$

$$\begin{aligned} \log(f_{OW}(x|\mu,\sigma,\nu)) = \log\left(\frac{\sigma\nu}{x}\right) + \sigma\log(\mu x) + (\mu x)^{\sigma} + (\nu-1)\log\left(\exp(\mu x)^{\sigma}-1\right) \\ -2\log\left[1+\left(\exp(\mu x)^{\sigma}-1\right)^{\nu}\right]. \end{aligned}$$

Estimating the odd Weibull distribution parameters can be challenging because of the combination of exponential and logarithmic functions in the likelihood function $\log(f_{OW}(x))$ when the estimated parameters have low values. In table **4-1** are presented a couple of examples with lifetime data that exhibit this condition.

Parameter	Electronic equipment	Mice mortallity
μ	0.00535	0.00761
σ	3.22388	6.22780
ν	0.28424	0.74950

Table 4-1: μ, σ and ν values for electronic equipment and mice mortality OW fitted models (Cooray, 2006).

Regarding the relative error metric again

$$\varepsilon_{OW} = 1 - \frac{f_{OW,dp}}{f_{OW,hp}}, \tag{4-6}$$

where $f_{OW,dp}$ and $f_{OW,hp}$ are the double-precision and high-precision (1024 bits) OW implementations, respectively. Let's implement two versions of the OW distribution and let's see what happens with ε_{OW} at low values of μ when `mpfr()`, `log1p()` and `expm1()` are discarded from the implementation given $x = 1$ and σ, ν fixed with values from table **4-1**.

```r
dOW_64_bits <- function(x, mu, sigma, nu, log=FALSE){
  prod1 <- (mu*x)^sigma
  loglik <- log(sigma*nu) - log(x) + sigma*( log(mu) + log(x) ) +
    (mu*x)^sigma + ( nu - 1 )*log(exp( prod1 ) - 1) -
    2*log( 1 + (exp(prod1) - 1)^nu )

  if (log == FALSE)
    dens <- exp(loglik)
  else
    dens <- loglik
  return(dens)
}
```

```r
dOW_1024_bits <- function(x, mu, sigma, nu, log=FALSE){
  prod1 <- Rmpfr::mpfr((mu*x)^sigma, 1024)
  loglik <- log(sigma*nu) - log(x) + sigma*( log(mu) + log(x) ) +
    (mu*x)^sigma + ( nu - 1 )*log(expm1( prod1 )) -
    2*log1p( (expm1(prod1))^nu )

  if (log == FALSE)
    dens <- exp(loglik)
  else
    dens <- loglik
  return(dens)
}
```

Figure **4-2** illustrates the behaviour of the relative errors. For electronic equipment and mice mortality data, the relative errors associated with the parameter $\hat{\mu}$, when computed using standard double-precision floating-point numbers, are on the order of $\mathcal{O}(10^{-9})$ and $\mathcal{O}(10^{-3})$[1], respectively, across most of the parameter space. These errors exceed the smallest representable floating-point number, 2.22×10^{-16}. These discrepancies underscore the necessity of employing high-precision arithmetic when working with this distribution. It should also be noted that even larger errors may occur for $\mu > 1$.

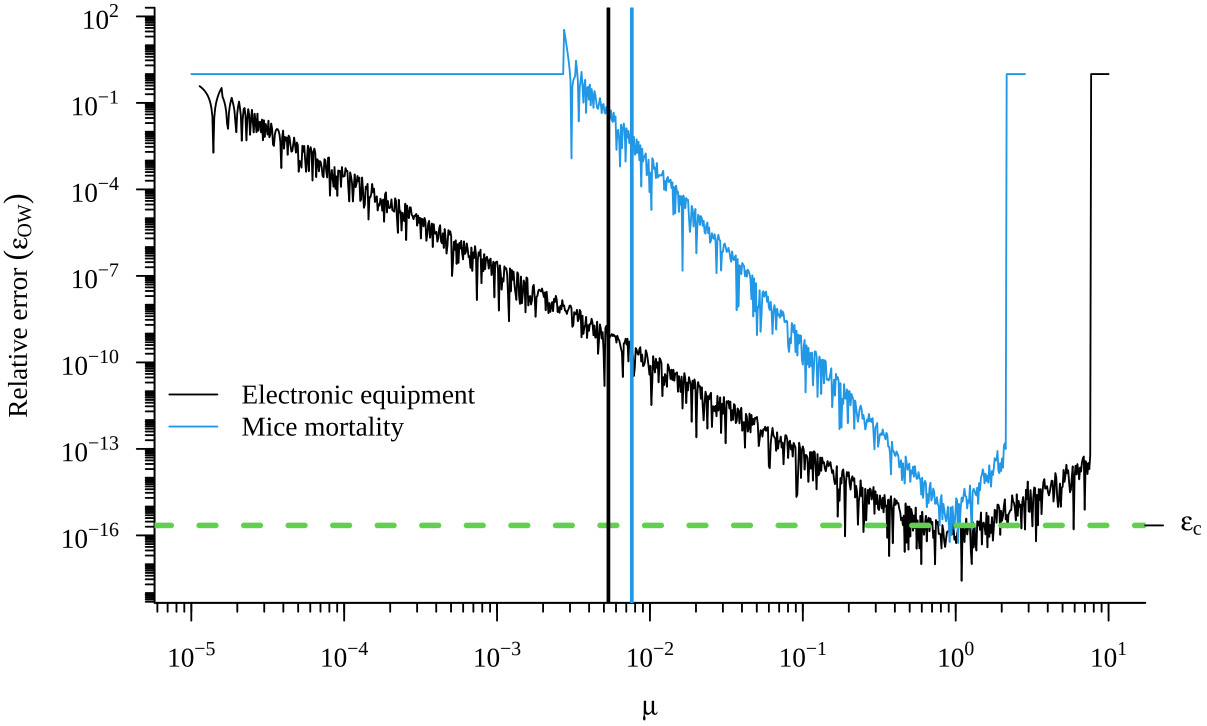


Figure 4-2: Relative error (ε_{OW}) of the 64-bits mean density function implementation of the OW distribution. Both vertical lines correspond to the estimated value for μ in each case (i.e. black line for the electronic equipment data and blue line for the mice mortality data). ε_c (green dashed line) is the smallest positive floating-point number in R (normally `2.220446e-16`).

4.2.2. Exponentiated Weibull distribution

The exponentiated Weibull (EW) distribution is a flexible probability distribution that is widely used in survival analysis and reliability engineering. This is a generalisation of the Weibull distribution proposed by Mudholkar and Hutson (1996). The PDF is given by the following expression:

$$f_{EW}(x|\rho,\kappa,\gamma) = \kappa\rho\gamma\,(\rho x)^{\kappa-1}\exp\left[-\left(\rho x\right)^{\kappa}\right]\left\{1-\exp\left[-\left(\rho x\right)^{\kappa}\right]\right\}^{\gamma-1}, \qquad x>0,\ \rho,\ \kappa,\ \gamma>0. \tag{4-7}$$

[1] $\mathcal{O}(\cdot)$ denotes the big-O notation, also known as Bachmann–Landau notation. It is used to describe the error magnitude in the context of analytic number theory (De Bruijn, 1981).

Logarithm of the density function

$$\begin{aligned}\log(f_{EW}(x|\rho,\kappa,\gamma)) = &\log(\kappa) + \log(\rho) + \log(\gamma) + (\kappa - 1)\log(\rho x) - (\rho x)^{\kappa} + \\ &(\gamma - 1)\left[1 - \exp\left(-(\rho x)^{\kappa}\right)\right]\end{aligned} \tag{4-8}$$

Regarding the head and neck cancer data (Efron, 1988) from section 1.1.2, its corresponding model equation is shown below:

$$T_i \overset{\text{iid.}}{\sim} EW(\rho,\ \kappa), \tag{4-9}$$
$$\log(\rho) = \beta_0 + \beta_1 x_i, \tag{4-10}$$
$$\log(\kappa) = \alpha_0, \tag{4-11}$$
$$\log(\gamma) = \lambda_0, \tag{4-12}$$

with $\rho,\ \kappa > 0$ and x_i is the treatment indicator covariate. The parameter values estimated for the EW model by Khan (2018) are reported in table **4-2**

Parameter	Estimate
β_0	10.183
β_1	0.561
$\log(\kappa)$	-2.116
$\log(\gamma)$	6.708

Table 4-2: β_0 and β_1 (which explain ρ), κ and γ values for head and neck EW fitted models (Efron, 1988).

Similarly as was done in section 4.2.1, we implemented two versions of the EW distribution. We examined the outcomes of using the double precision implementation far from β_1 true value given $x = 1$ and κ, γ, β_0 fixed (with values from table **4-2**). For further details regarding the code implementation, please refer to section *EW implementations*.

In the depicted Figure **4-3**, the behaviour of the relative error is presented. When performing parameter estimation for head and neck cancer data, the relative error using traditional double precision computations near the actual parameter value is roughly $\mathcal{O}\left(10^{-14}\right)$. This value is larger than the smallest floating-point value of 2.22×10^{-16}, indicating that the traditional double-precision computations may not be sufficient for accurate estimation of the ρ parameter in this distribution.

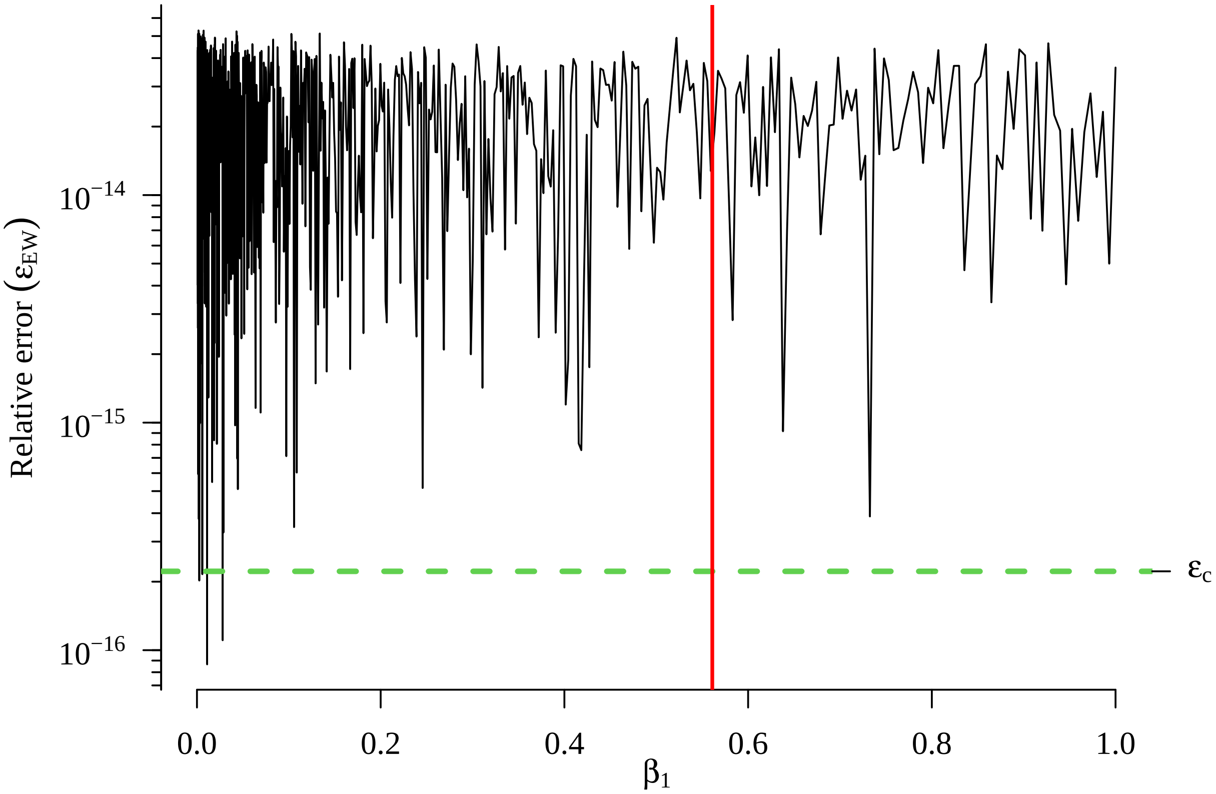


Figure 4-3: Relative error (ε_{EW}) of the 64-bits mean density function implementation of the EW distribution. The vertical lines correspond to the estimated value for β_1. ε_c (green dashed line) is the smallest positive floating-point number in R (normally `2.220446e-16`).

EW implementations

This sections shows the standard double-precision (64 bits) implementation and the hig-precision version (1024 bits) of the EW distribution.

```r
dEW_64_bits <- function(x, rho, kappa, gamma, log=FALSE){
  prod1 <- rho*x
  prod2 <- prod1^kappa
  loglik <- log(rho) + log(kappa) + log(gamma) + (kappa - 1)*log(prod1) -
    prod2 + (gamma - 1)*log(1 - exp(-prod2))
  if (log == FALSE)
    dens <- exp(loglik)
  else
    dens <- loglik
  return(dens)
}

dEW_1024_bits <- function(x, rho, kappa, gamma, log=FALSE){
  prod1 <- mpfr(rho*x, 1024)
  prod2 <- mpfr(-(rho*x)^kappa, 1024)
  loglik <- log(rho) + log(kappa) + log(gamma) + (kappa - 1)*log(prod1) +
    prod2 + (gamma - 1)*log( -expm1(prod2) ) # log1mexp(prod2)
  if (log == FALSE)
    dens <- exp(loglik)
  else
    dens <- loglik
  return(dens)
}
```

4.3. Conclusion

High-precision arithmetic played a crucial role in the specific examples presented in this chapter and in the development of this thesis, effectively controlling computation errors that stem from small true parameter values for a couple of flexible lifetime distributions. However, it is common to find other flexible lifetime distributions that also depend on combinations of exponential and logarithmic functions. Therefore, analogous analyses to those presented in this chapter could elucidate the behaviour of the distributions and enable the identification of regions in the parameter space that require greater attention to ensure reliable results.

Part II

Estimation

5. Parameter estimation

> *The journey of a thousand miles begins with one step.*
>
> — **Lao Tzu**
>
> (Philosopher and poet)

Parameter estimation is a central problem in statistics. This problem has traditionally been tackled as a likelihood maximisation problem, as introduced by Fisher (1912). As explained previously (chapters 2 and 3), the maximum likelihood (ML) estimation method involves performing an optimisation through the first derivative of the log-likelihood function and solving the outgoing system of equations. This method is preferred because it produces consistent and efficient estimators (Daniels, 1961). The method of moments is an alternative introduced by Diaconis (1987); Sheynin (1994) and exploited by Pearson (1936), which consists of equating the population moments to the sample moments, obtaining a system of equations whose variables are the parameters of interest. While the previously mentioned methods focus solely on the observed data, the Bayesian framework incorporates prior information into the estimation process, leading to a more comprehensive and nuanced understanding of the parameters (Gelman et al., 2017).

R possess an extensive number of packages (add-ons) to enhance its capabilities, e.g **MASS** package with its function `fitdistr`, which is a general purpose maximum-likelihood estimation routine that implements `optim` in order to fit any of the univariate distributions of table **5-1** with or without constraints (Venables and Ripley, 2013). **fitdistrplus** is a package that extends `fitdistr` because it provides functions for fitting distributions with quantile matching estimation, moment matching estimation, maximum goodness-of-fit ,and maximum likelihood estimation with censored and non-censored data; In addition, **fitdistrplus** allows the implementation of a user-supplied optimisation algorithm (Delignette-Muller and Dutang, 2015). There are other interesting efforts: the **actuar** package estimates parameters with maximum goodness-of-fit, using seven different types of distances (Goulet, 2008); **distrMod** consist on implementation of probability models in `S4` object system, with methods for fitting distributions based on user-supplied optimization functions (Kohl and Ruckdeschel, 2010); **modest** (Poncet, 2019), **lmomco** (Asquith, 2018) and **Lmoments** (Karvanen, 2006, 2019) supply tools for estimating the mode, moments and L-moments, respectively. Furthermore, Henningsen and Toomet (2011) developed **maxLik** package in order to provide an environment for exclusively performing maximum likelihood estimation. They implemented some other optimisation methods, such as BHHH (Berndt-Hall-Hall-Hausman). The package has a wrapper `maxOptim`, which calls `optim` in order to implement SANN (simulated annealing), BFGS and Nelder-Mead algorithms.

Distribution	R name
beta	`beta`
binomial	`binom`
Cauchy	`cauchy`
Chi-squared	`chisq`
exponential	`exp`
F	`f`
gamma	`gamma`
geometric	`geom`
hypergeometric	`hyper`
log-normal	`lnorm`
logistic	`logis`
negative-binomial	`nbinom`
normal	`norm`
Poisson	`pois`
T	`t`
uniform	`unif`
Weibull	`weibull`
Wilcoxon	`wilcox`

Table 5-1: Probability distributions and its function name in `fitdistr`.

In the Bayesian framework, several packages are available that provide tools for fitting models. The package **BayesTools** offers an extensive array of tools for Bayesian statistical modelling and inference, particularly for JAGS and Stan models, as detailed by Bartoš (2021). **bayesm** is specifically tailored for efficient Bayesian estimation and model selection, as Rossi (2023) discusses. **LaplacesDemon** features a user-friendly interface for Bayesian inference and model comparison (Statisticat and LLC., 2021). For Markov chain Monte Carlo (MCMC) sampling, a crucial technique in Bayesian inference for complex models, **MCMpack** provides robust tools (Martin et al., 2011). Lastly, **nimble** is recognised for its flexibility and efficiency in Bayesian modelling and inference, especially in hierarchical models, as presented by de Valpine et al. (2017).

In this chapter, we introduce the `maxlogL` function available in the **EstimationTools** package (Mosquera Gutiérez and Hernández, 2023) from the CRAN (`https://cran.rstudio.com/web/packages/EstimationTools/index.html`). `maxlogL` is a computational procedure that performs log-likelihood maximisation stated on equation (2-2) based on different box-constrained algorithms included previously in `optim` or `nlminb` only with the density/mass function implemented as usual in R. We will show that the user can define its own distribution or use any existing distribution in any package. The remainder of this chapter defines the maximization problem computationally. We then present a simulation study to evaluate the performance of `maxlogL` with data generated from normal, ZIP, and user-defined distributions. Finally, we present an application with a real dataset and present some conclusions.

5.1. Estimation

Let be $\boldsymbol{y}^\top = (y_1, y_2, ..., y_n)$ a random sample with n observations drawn from a population with distribution $f(\cdot|\boldsymbol{\theta})$, with $\boldsymbol{\theta}$ a vector of parameters. The likelihood function of $\boldsymbol{\theta}$ is

$$\mathcal{L}(\boldsymbol{\theta}|\boldsymbol{y}) = \prod_{i=1}^{n} f(y_i|\boldsymbol{\theta}). \tag{5-1}$$

The ML estimation method finds the parameter values that make the data more probable. This is achieved by computing a vector $\hat{\boldsymbol{\theta}}$ such that

$$\hat{\boldsymbol{\theta}} = \underset{\boldsymbol{\theta}\in\Theta}{\arg\,\text{máx}}\ \mathcal{L}(\boldsymbol{\theta}|\boldsymbol{y}). \tag{5-2}$$

It is usual to perform maximisation of the log-likelihood function, i.e. $l(\boldsymbol{\theta}|\boldsymbol{y}) = \log \mathcal{L}(\boldsymbol{\theta}|\boldsymbol{y})$. The variance-covariance matrix of the ML estimators is given by

$$Var(\hat{\boldsymbol{\theta}}) = \mathcal{J}^{-1}(\hat{\boldsymbol{\theta}}) = C(\hat{\boldsymbol{\theta}}), \tag{5-3}$$

where $\mathcal{J}(\hat{\boldsymbol{\theta}})$ is the observed Fisher information matrix

$$\mathcal{J}(\hat{\boldsymbol{\theta}}) = \frac{\partial^2}{\partial \boldsymbol{\theta}^2} \log(f(y_i|\boldsymbol{\theta})). \tag{5-4}$$

The standard errors can be calculated as the square root of the diagonal elements of matrix C (Pawitan, 2013b)

$$S.E(\hat{\boldsymbol{\theta}}) = \sqrt{C_{jj}(\hat{\boldsymbol{\theta}})}. \tag{5-5}$$

The R function presented here calculates $l(\boldsymbol{\theta}|\boldsymbol{y})$ computationally, and computes standard errors from the Hessian matrix.

5.2. Basic usage and features

EstimationTools package can be downloaded and used typing the following instructions in the console:

```r
install.packages("EstimationTools")
library(EstimationTools)
```

or these others command lines

```r
if (!require('devtools')) install.packages('devtools')
devtools::install_github('Jaimemosg/EstimationTools', force = TRUE)
library(EstimationTools)
```

EstimationTools loads several functions in the global environment, `maxlogL` function is one of them. It is a wrapper function developed for flexible maximum likelihood estimation and produces an S3 object of class `maxlogL` with the estimates and some details about the estimation process as its attributes. It can be executed by stating its most important arguments

```r
maxlogL(x, dist, optimizer, lower = NULL, upper = NULL)
```

where the argument `x` is a vector with data to be fitted, `dist` corresponds to the probability density/mass function of the working distribution, whereas `upper` and `lower` are limits used when the user selects box-constrained algorithms. Through the argument `optimizer`, `maxlogL` allows to use any of `optim` algorithms for optimisation, `nlminb` routine for unconstrained/box-constrained optimisation or `DEoptim` for computation with differential evolutionary algorithm. For further information, please visit the stable repository `https://cran.r-project.org/web/packages/DEoptim/index.html` or the last development version `https://github.com/ArdiaD/DEoptim`.

Hence, the user must pass a vector with the data and specify a probability distribution function available in R. For example, let's fit a sample generated from a normal distribution,

$$Z \sim \mathrm{NO}(\mu = 10,\ \sigma^2 = 1)$$

.

This could be done using the next command lines

```r
set.seed(1000)
z1 <- rnorm(n = 1000, mean = 10, sd = 1)

fit1 <- maxlogL(
  x = z1,
  dist = "dnorm",
  start = c(2, 3),
  lower = c(-15, 0),
  upper = c(15, 10)
)
```

EstimationTools package also provides a `summary` method for class `maxlogL`, which displays AIC (Akaike Information Criterion), BIC (Bayesian Information Criterion), ML estimates, their standard errors, the optimisation routine selected by the user, and the method used for the computation of the standard error.

```
summary(fit1)
```

```
## ---------------------------------------------------------------
## Optimization routine: nlminb
## Standard Error calculation: Hessian from optim
## ---------------------------------------------------------------
##        AIC      BIC
##   2804.033 2813.849
## ---------------------------------------------------------------
##      Estimate  Std. Error Z value Pr(>|z|)
## mean   9.98752    0.03103  321.87   <2e-16 ***
## sd     0.98126    0.02194   44.72   <2e-16 ***
## ---
## Signif. codes:  0 ’***’ 0.001 ’**’ 0.01 ’*’ 0.05 ’.’ 0.1 ’ ’ 1
## ---------------------------------------------------------------
## Note: p-values valid under asymptotic normality of estimators
## ---
```

The estimated parameters $\hat{\mu} = 9.988$ and $\hat{\sigma} = 0.981$ are close to the true values, as expected. The standard errors were computed with the inverse Hessian (recall equations **??** and 5-5). The `maxlogL` function has two options

- If `StdE_Method = optim`, the Hessian matrix is computed with `optim` (with option ‘hessian = TRUE‘ under the hood). This is the default option.
- If the previous implementation fails or if the user chooses `StdE_Method = numDeriv`, it is calculated using `hessian` function from **numDeriv** package. Let's fit using using this option below.

```
fit2 <- maxlogL(
  x = z1,
  dist = "dnorm",
  start = c(2, 3),
  lower = c(-15, 0),
  upper = c(15, 10),
  StdE_method = "numDeriv"
)
summary(fit2)

## ---------------------------------------------------------------
## Optimization routine: nlminb
## Standard Error calculation: numDeriv::hessian
## ---------------------------------------------------------------
##        AIC      BIC
##   2804.033 2813.849
## ---------------------------------------------------------------
##      Estimate  Std. Error Z value Pr(>|z|)
## mean   9.98752    0.03103  321.87   <2e-16 ***
## sd     0.98126    0.02194   44.72   <2e-16 ***
## ---
## Signif. codes:  0 ’***’ 0.001 ’**’ 0.01 ’*’ 0.05 ’.’ 0.1 ’ ’ 1
## ---------------------------------------------------------------
## Note: p-values valid under asymptotic normality of estimators
## ---
```

Optionally, the **EstimationTools** provides the `bootstrap_maxlogL` method for `maxlogL` objects, which uses the `boot` function from the **boot** package (Davison and Hinkley, 1997; Canty and Ripley, 2017). This method replaces the fitted regression parameters and standard errors, initially calculated using the Hessian matrix, with those computed using bootstrap.

```
fit1_bootstrap <- fit1
bootstrap_maxlogL(fit1_bootstrap, R = 200)

##
## ...Bootstrap computation of Standard Error. Please, wait a few minutes...
##
##  -->Done <---

summary(fit1_bootstrap)

## ---------------------------------------------------------------
## Optimization routine: nlminb
## Standard Error calculation: Bootstrap
## ---------------------------------------------------------------
##        AIC      BIC
##   2804.033 2813.849
## _______________________________________________________________
##      Estimate  Std. Error Z value Pr(>|z|)
## mean   9.98752    0.02832  352.63   <2e-16 ***
## sd     0.98126    0.02034   48.23   <2e-16 ***
## ---
## Signif. codes:  0 ’***’ 0.001 ’**’ 0.01 ’*’ 0.05 ’.’ 0.1 ’ ’ 1
## _______________________________________________________________
## Note: p-values valid under asymptotic normality of estimators
## ---
```

p-values for the estimated parameters were calculated using the Wald test (Wald, 1943). The `link` argument of `maxlogL` is a list with entries `fun` and `over`, which specify the link functions applied and the name of the linked parameters in the probability function implemented in R respectively. The estimation performed above can be performed by applying a logarithmic link function to σ, avoiding problems of estimation in the boundary of parametric space. The usage is illustrated in the following code snippet:

```
fit3 <- maxlogL(
  x = z1,  dist = "dnorm", link = list(over = "sd", fun = "log_link")
)
summary(fit3)
```

```
## _______________________________________________________________
## Optimization routine: nlminb
## Standard Error calculation: Hessian from optim
## _______________________________________________________________
##        AIC      BIC
##   2804.033 2813.849
## _______________________________________________________________
##      Estimate  Std. Error Z value Pr(>|z|)
## mean   9.98752    0.03103  321.87   <2e-16 ***
## sd     0.98126    0.02194   44.72   <2e-16 ***
## ---
## Signif. codes:  0 ’***’ 0.001 ’**’ 0.01 ’*’ 0.05 ’.’ 0.1 ’ ’ 1
## _______________________________________________________________
## Note: p-values valid under asymptotic normality of estimators
## ---
```

The user can apply link functions to more than one parameter of the distribution

```
fit4 <- maxlogL(
  x = z1,
  dist = "dnorm",
  link = list(
    over = c("mean", "sd"),
    fun = c("log_link", "log_link")
  )
)
summary(fit4)
```

```
## _______________________________________________________________
## Optimization routine: nlminb
## Standard Error calculation: Hessian from optim
## _______________________________________________________________
##        AIC      BIC
##   2804.033 2813.849
## _______________________________________________________________
##      Estimate  Std. Error Z value Pr(>|z|)
## mean   9.98752    0.03103   321.87   <2e-16 ***
## sd     0.98126    0.02194    44.72   <2e-16 ***
## ---
## Signif. codes:  0 ’***’ 0.001 ’**’ 0.01 ’*’ 0.05 ’.’ 0.1 ’ ’ 1
## _______________________________________________________________
## Note: p-values valid under asymptotic normality of estimators
## ---
```

Other link functions available are logit and negative inverse, which must be specified as `"logit_link"` and `"NegInv_link"`. On the other hand, `maxlogL` allows to define fixed known parameters, e.g., the sample size n in the estimation of success proportion in a binomial distribution such the following:

$$N \sim \mathrm{BI}(p = 0.3,\ n = 10).$$

These parameters can be specified with `fixed` argument, which is a list that stores the fixed parameter values specified by their names

```
set.seed(100)
N <- rbinom(n = 100, size = 10, prob = 0.3)
phat <- maxlogL(
  x = N,
  dist = "dbinom",
  fixed = list(size = 10),
  link = list(over = "prob", fun = "logit_link")
)
summary(phat)
```

```
## _______________________________________________________________
## Optimization routine: nlminb
## Standard Error calculation: Hessian from optim
## _______________________________________________________________
##        AIC      BIC
##   334.9805 334.9805
## _______________________________________________________________
##      Estimate  Std. Error Z value Pr(>|z|)
## prob   0.31200    0.01465    21.3   <2e-16 ***
## ---
## Signif. codes:  0 ’***’ 0.001 ’**’ 0.01 ’*’ 0.05 ’.’ 0.1 ’ ’ 1
## _______________________________________________________________
## Note: p-values valid under asymptotic normality of estimators
## ---
```

As can be seen, the procedure applies the inverse of the link function to return the parameter to the original scale.

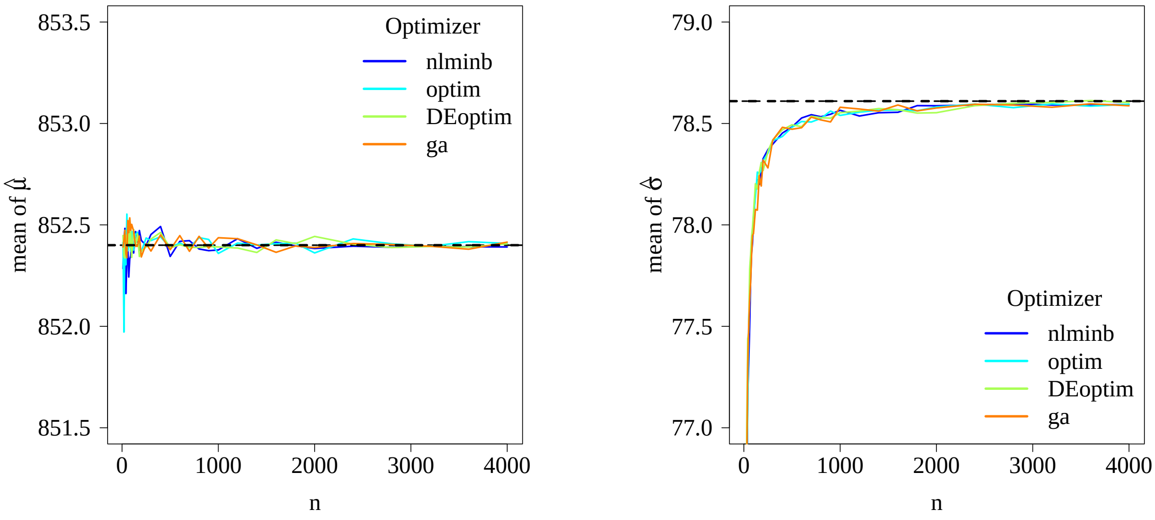
5.3. Simulation Study

We simulate the following two types of random variables:

- $Z_1 \sim \mathrm{NO}(\mu = 852.4,\ \sigma^2 = 78.61)$ based on MLE estimates by Pawitan (2013a) of 100 measurements of the speed of light, extracted from Stigler (1977).
- $Z_2 \sim \mathrm{ZIP}(\lambda = 0.36,\ \pi = 0.44)$ based on MLE estimates of traffic accident data by Wagh and Kamalja (2018).

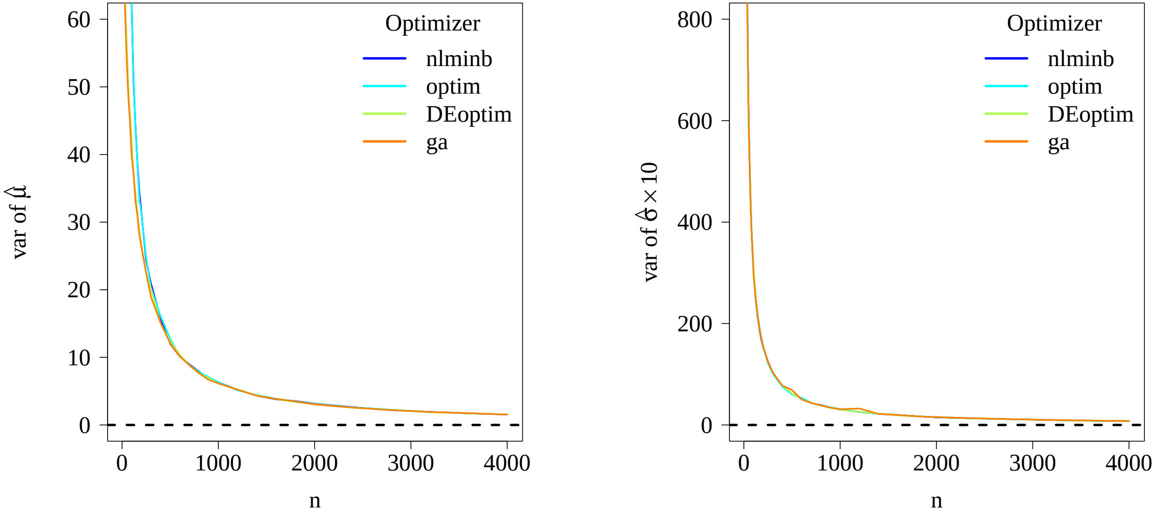
The estimation was performed with `maxlogL`with 10000 replications for each sample size using the four available optimisers: `nlminb`, `optim` with BFGS algorithm, `DEoptim` with its Differential Evolution algorithm and `ga`, an implementation of genetic algorithms. The last two have been applied using the default values for the arguments. The results of the Monte Carlo study for the two above-mentioned distributions are presented in Figure **5-1** and **5-3**.

Figure **5-1** illustrates the convergence of `nlminb`, `DEoptim`, and `ga` toward specific parameter values as the sample size increases in the context of the normal distribution. The `ga` algorithm exhibits higher variability in its parameter estimates for smaller sample sizes, requiring more



(a) (b)

Figure 5-1: Mean value for (a) location parameter $\hat{\mu}$ and (b) scale parameter $\hat{\sigma}$ versus sample size n in normal distribution based on 4000 replications. Horizontal red lines represent the true values of the parameters.

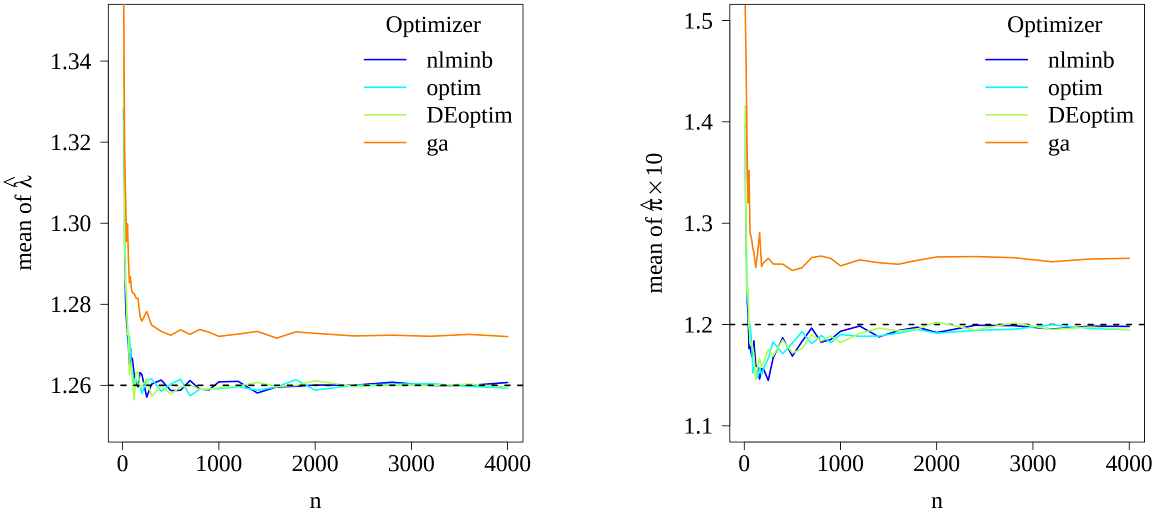


(a) (b)

Figure 5-2: Variance for (a) location parameter $\hat{\mu}$ and (b) scale parameter $\hat{\sigma}$ versus sample size n in normal distribution based on 10000 replications.

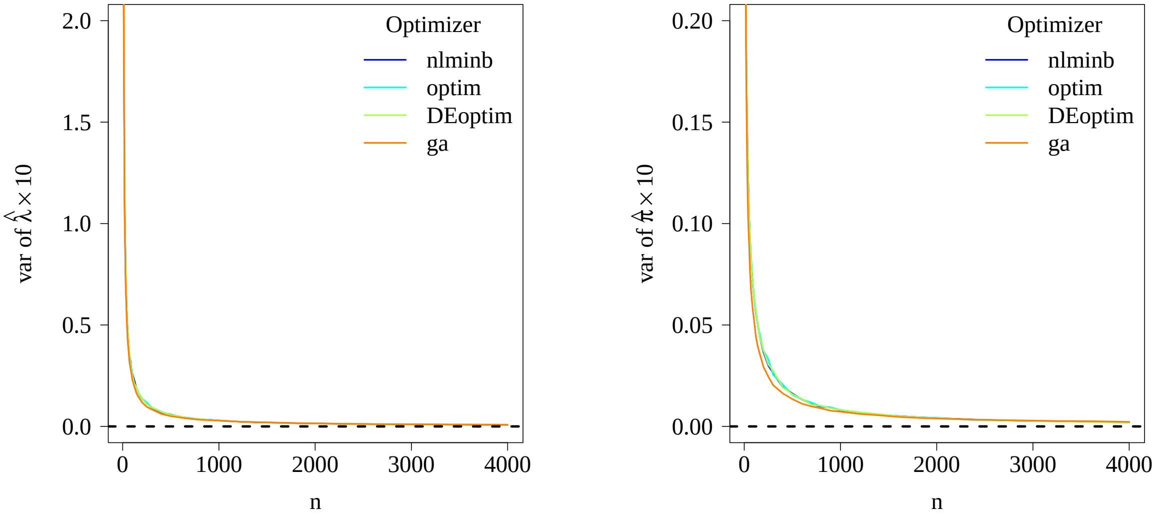
time to stabilize. Conversely, `DEoptim` consistently provides stable and reliable estimates across various sample sizes. However, it is important to note that the `optim` algorithm yields biased estimates in this particular case.

Figure **5-3** demonstrates that as the sample size increases, the mean estimates converge toward the true ZIP parameter values as expected under regularity conditions, with the exception of the `ga` algorithm, which consistently produces biased estimates. Furthermore,



(a) (b)

Figure 5-3: Mean value for (a) rate parameter $\hat{\lambda}$ and (b) extra zeros proportion $\hat{\pi}$ versus sample size n in the ZIP distribution based on 10000 replications. Horizontal red lines represent the true values of the parameters.



(a) (b)

Figure 5-4: Variance for (a) rate parameter $\hat{\lambda}$ and (b) extra zero proportion $\hat{\pi}$ versus sample size n in the ZIP distribution based on 10000 replications.

the variance of the estimated parameters decreases as the sample size increases, in accordance with the efficiency property of maximum likelihood estimators (Gurland, 1954; Daniels, 1961). These findings are visually represented in Figures **5-2** and **5-4** for the normal and ZIP distributions, respectively.

5.4. Illustrative examples

In the following examples, we replicate the maximum likelihood method with `maxlogL` in two applications: fitting the power Lindley distribution to model the tensile strength of carbon fibers and parameter estimation in a two-stage hierarchical model of retention proportions in memory tests.

5.4.1. Tensile strength data: power Lindley distribution

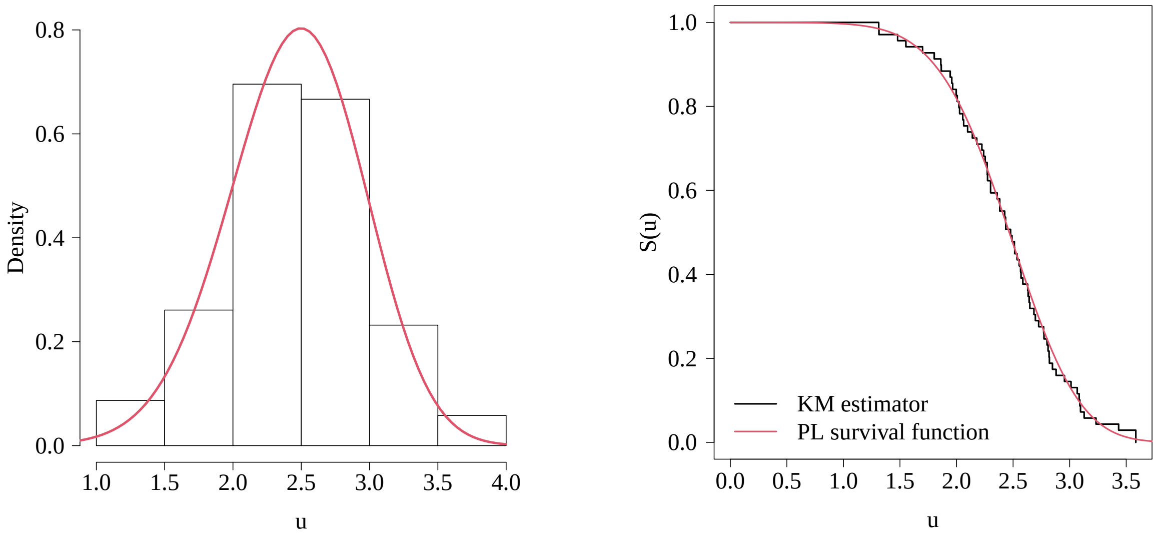
In section 3.4.2 we presented this dataset, which comprises the tensile strength of 69 specimens of carbon fiber modelled with a power Lindley (PL) distribution. We implemented the density and cumulative density functions in the R `dPL` and `pPL` functions displayed below

```r
dPL <- function(x, mu, sigma, log=FALSE){
  if (any(x < 0))
    stop(paste("x must be positive", "\n", ""))
  if (any(mu <= 0))
    stop(paste("mu must be positive", "\n", ""))
  if (any(sigma <= 0))
    stop(paste("sigma must be positive", "\n", ""))

  loglik <- log(mu) + 2*log(sigma) - log(sigma+1) +
    log(1+(x^mu)) + (mu-1)*log(x) - sigma*(x^mu)

  if (log == FALSE)
    density <- exp(loglik)
  else density <- loglik
  return(density)
}

pPL <- function(q, mu, sigma,
                lower.tail=TRUE, log.p=FALSE){
  if (any(q < 0))
    stop(paste("q must be positive", "\n", ""))
  if (any(mu <= 0))
    stop(paste("mu must be positive", "\n", ""))
  if (any(sigma <= 0))
    stop(paste("sigma must be positive", "\n", ""))

  cdf <- 1 - (1+((sigma/(sigma+1))*q^mu))*exp(-sigma*(q^mu))
  if (lower.tail == TRUE)
    cdf <- cdf
  else cdf <- 1 - cdf
  if (log.p == FALSE)
    cdf <- cdf
  else cdf <- log(cdf)
  cdf
}
```

Then, we estimate the parameters with `maxlogL` taking the vector of the strengths from data set `fibers`, as follows:

```
# Fitting of tensile strenght data
st <- Fibers$Strenght
theta <- maxlogL(x = st, dist = "dPL",
                 link = list(over = c("mu", "sigma"),
                             fun = c("log_link", "log_link")))
summary(theta)

## ---------------------------------------------------------------
## Optimization routine: nlminb
## Standard Error calculation: Hessian from optim
## ---------------------------------------------------------------
##       AIC      BIC
##   102.119 106.5872
## ---------------------------------------------------------------
##       Estimate  Std. Error Z value Pr(>|z|)
## mu      3.86778    0.31371  12.329  < 2e-16 ***
## sigma   0.04967    0.01599   3.107  0.00189 **
## ---
## Signif. codes:  0 ’***’ 0.001 ’**’ 0.01 ’*’ 0.05 ’.’ 0.1 ’ ’ 1
## ---------------------------------------------------------------
## Note: p-values valid under asymptotic normality of estimators
## ---
```

Estimations are $\hat{\mu} = 3.8678$ and $\hat{\sigma} = 0.0497$. Essentially, we obtain the same values computed by Ghitany et al. (2013). In Figure **5-5** we showed the performance of parameter estimation plotting the corresponding density along with the histogram in the left panel and the estimated survival function along with Kaplan-Meier estimator in the right panel.



(a) (b)

Figure 5-5: Fitting of tensile strength data: (a) estimated density and histogram; (b) estimated power Lindley survival function and Kaplan-Meier (KM) estimator.

5.4.2. Forgetting data: hierarchical binomial distribution

The `maxlogL` function is capable of fitting hierarchical models with the proper function definition of the input variable. To illustrate the estimation, we replicated the forgetting data example presented in Myung (2003), which used data from Murdock, Bennet B. (1961).

The retention function is a probability function that models the proportion of correct recall at time t_i in each trial in memory tests. Myung (2003) studied the following two models in their application example:

$$\begin{aligned} \text{power model: } p(a,b,t) &= at^{-b}, \\ \text{exponential model: } p(a,b,t) &= a\exp\left(-bt\right), \quad a,b>0,\ t>0. \end{aligned} \tag{5-6}$$

$m = 100$ (number of trials)						
Retention Interval (sec.)	1	3	6	9	12	18
Observed proportion	0.94	0.77	0.40	0.26	0.24	0.16

Table 5-2: Observed proportion of recalls at each time.

Each observation in the dataset corresponds to a proportion obtained as the quotient of correct responses (w_i) and the total number of independent trials (replications of each memory test, represented by m). This type of experiment can be modelled with a binomial distribution

$$f(w_i|a,b,m) = \frac{m!}{(m-w_i)!\,w_i!} p(a,b,t_i)\,[1-p(a,b,t_i)]^{m-w_i} \tag{5-7}$$

The usefulness of each retention equation is given by its goodness-of-fit.

Power model implementation

The hierarchical model with the power retention function is implemented in an R function as follows:

```
# Power model implementation
power_logL <- function(x, a, b, log = FALSE){
  p <- a * x[,1]^(-b)
  f <- dbinom(x = x[,2], size = m, prob = p)
  if (log == TRUE)
    density <- log(f) else density <- f
  return(density)
}
```

The conditional density in equation (5-7) depends on w_i, but the proportion of successes depends on t, as equation (5-6) shows. Therefore, the input argument `x` must be a $n \times 2$ matrix, where n is the sample size. In forgetting data, $n = 6$. Then, the estimation is performed as usual with `maxlogL`. Note lines six in the following chunk of code, which corresponds to the matrix definition of the input data aforementioned

```
# Power model estimation
m <- 100 # Independent trials
t <- c(1,3,6,9,12,18) # time intervals
p.ob <- c(0.94,0.77,0.40,0.26,0.24,0.16) # Observed proportion
w <- p.ob*m # Correct responses
w <- as.integer(w)
Wi <- matrix(c(t, w), ncol=2, nrow=6)

retention.pwr <- maxlogL(x = Wi, dist = "power_logL", lower = c(0.01,0.01),
                         upper = c(1,1), start = c(0.1,0.1))
summary(retention.pwr)

## ---------------------------------------------------------------
## Optimization routine: nlminb
## Standard Error calculation: Hessian from optim
## ---------------------------------------------------------------
##       AIC    BIC
##   57.4522 58.422
## ---------------------------------------------------------------
##   Estimate  Std. Error Z value Pr(>|z|)
## a   0.95312    0.01860   51.25   <2e-16 ***
## b   0.49793    0.03236   15.38   <2e-16 ***
## ---
## Signif. codes:  0 ’***’ 0.001 ’**’ 0.01 ’*’ 0.05 ’.’ 0.1 ’ ’ 1
## ---------------------------------------------------------------
## Note: p-values valid under asymptotic normality of estimators
## ---
```

In this application, convergence was achieved by solving a box-constrained likelihood optimisation whose boundaries were specified in arguments `lower` and `upper`. Furthermore, tuning the initial values was necessary with argument `start`. The computed values are $\hat{\boldsymbol{\theta}} = (\hat{a}_{\text{pwr}}, \hat{b}_{\text{pwr}}) = (0.953, 0.498)$, which are the same estimates of Myung (2003). Power fitting is illustrated in Figure **5-6**.

Exponential model implementation

Similarly as before, the exponential retention function is implemented and the input data is defined as a matrix.

```
# Exponential model implementation
exp_logL <- function(x, a, b, log = FALSE){
  p <- a * exp(-x[,1]*b)
  f <- dbinom(x = x[,2], size = m, prob = p)
  if (log == TRUE)
    density <- log(f) else density <- f
  return(density)
}

# Exponential model estimation
m <- 100 # Independent trials
t <- c(1,3,6,9,12,18) # time intervals
p.ob <- c(0.94,0.77,0.40,0.26,0.24,0.16) # Observed proportion
w <- p.ob*m # Correct responses
w <- as.integer(w)
Wi <- matrix(c(t, w), ncol=2, nrow=6)

retention.exp <- maxlogL(x = Wi, dist = ’exp_logL’, lower = c(0.1,0.1),
                         upper = c(2,2), start = c(0.1,0.2))
```

```
summary(retention.exp)

## ---------------------------------------------------------------
## Optimization routine: nlminb
## Standard Error calculation: Hessian from optim
## ---------------------------------------------------------------
##       AIC     BIC
##   41.3329 42.3027
## ---------------------------------------------------------------
##   Estimate  Std. Error Z value Pr(>|z|)
## a  1.070112   0.031342   34.14   <2e-16 ***
## b  0.130826   0.009252   14.14   <2e-16 ***
## ---
## Signif. codes:  0 ’***’ 0.001 ’**’ 0.01 ’*’ 0.05 ’.’ 0.1 ’ ’ 1
## ---------------------------------------------------------------
## Note: p-values valid under asymptotic normality of estimators
## ---
```

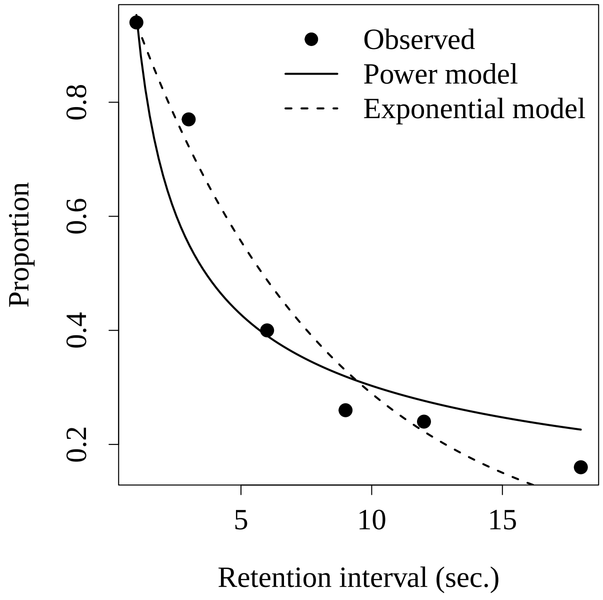


Figure 5-6: Observed proportion of recalls and models fitted.

Again, we obtain the same estimates of Myung (2003): $\hat{\boldsymbol{\theta}} = (\hat{a}_{\text{exp}}, \hat{b}_{\text{exp}}) = (1.070, 0.131)$. Exponential fitting is shown in Figure **5-6**. According to the Akaike information criterion, the exponential model has better fitness ($AIC_{\text{exp}} = 41.33$ against $AIC_{\text{pwr}} = 57.45$).

5.5. Conclusions

We implemented classic estimation via maximisation of the log-likelihood function through basic optimisation routines in R such as `optim` and `nlminb`. With `maxlogL`, we enable researchers, developers, and users to compute MLEs of any distribution in a fast and reliable way. The algorithm computes the standard error of estimates through a Hessian matrix, and with our `summary` method, it is possible to print it. Alternatively, a bootstrap algorithm can be used. In some cases, it is possible to implement estimation in hierarchical models with appropriate tuning of initial values.

Furthermore, we explored the implementation of evolutionary algorithms and and genetic algorithms, which are useful for performing estimation in distributions with regularity issues (Haupt and Haupt, 2003; Dorsey and Mayer, 1995). Further investigation investigation is necessary to address biases observed in simulation studies when using the `ga` algorithm.

In the next chapter, we will leverage our routine to delve deeper into log-likelihood estimation, with a focus on developing new parametric regression models.

6. Regression model

> ”
>
> *All models are approximations. Essentially, all models are wrong, but some are useful. However, the approximate nature of the model must always be borne in mind.*
>
> — **George E. P. Box**
>
> (Industrial Chemist and Statistician (by accident))

Lifetime distributions themselves are an improvement with respect to classical distributions because they allow us to model data described by a bathtub-shaped hazard function with a good fitting. However, estimation without covariates is not always suitable for analysing the results from clinical trials in survival analysis or accelerated life test for reliability. In such situations, the inclusion of explanatory variables is necessary.

From the classic statistics approach, modelling proposals have been made based on regression analysis. For example, Cox (1972) proposed the proportional hazards semiparametric model, with the fundamental assumption that the covariates have a multiplicative effect on the hazard function, Therneau and Grambsch (2000a) and implemented this model in the `coxph` function, from **survival** package. Despite the underlying assumption, this is a flexible methodology because it does not depend on distribution specification. However, for predicting outcomes for specific sample units or computing the mean survival, the parametric approach is more convenient (Jackson, 2016) (actually, Cox preferred the parametric approach [Reid, 1994]). An alternative for fulfilling this idea lies in the Event History Analysis by (Broström, 2018). This approach is pertinent for regression problems in survival analysis because it proposes using parametric proportional hazard models for common distribution families (Weibull, Gompertz, lognormal and Piecewise constant hazard functions) through `aftreg` function from the **eha** package (Broström, 2020).

Another important class of models, the Accelerated Failure Time (AFT) models, arise from the assumption that the experimental units associated with the baseline treatment are different from those associated with the other treatments in terms of survival by a factor or acceleration rate (Kalbfleisch and Prentice, 2002). They can be fitted with the `survreg` function, also from the **survival** package. This implementation provides distributions such as Weibull, exponential, Gaussian, logistic, lognormal, and loglogistic, but also enables the users to include distributions with location and scale parameters using guidelines from `survreg.distributions` framework, i.e, it supports only two-parameter distributions.

Other interesting approaches have been developed to overcome the restriction in distribution complexity, e.g, `flexsurreg` from **flexsurv** package. The last allows the fitting of time-to-event with MLE not only with common distributions but also with the Royston-Parmar spline model, generalised gamma, and generalised F distributions (Jackson, 2016). A more recent approach to model data from experiments in survival analysis consists of adjusting the parameters of a distribution selected in terms of explanatory variables. A development in this direction is related to GAMLSS (*Generalized Additive Models for Location, Shape and Scale*) by Rigby and Stasinopoulos (2005). Within this framework, it is possible to implement any distribution that has four parameters or less and estimate its parameters in terms of covariates from a log-likelihood function. This proposal includes the creation of **gamlss** package (Stasinopoulos and Rigby, 2007). It is possible to perform parameter estimation of any distribution implemented as a `gamlss.family` structure. Visit Stasinopoulos et al. (2017b) for more details.

All of these frameworks are relevant and enrich the possibilities of survival analysis. Those packages that offer flexible modelling require the programming of new distributions from the creation of complex functions, including data structures and combinations of various types of objects, which makes the usability feasible for medium or advanced R users. In this chapter, we specify the mathematical form of the regression models covered in this project and present a wrapper computational procedure, `maxlogLreg`, from package **EstimationTools**. It performs the log-likelihood maximum estimation of linear models only with a straightforward implementation of the density function, as is customary in R based on the capabilities of the aforementioned `maxlogL`.

6.1. Model Statement

Let be $\boldsymbol{y}^\top = (y_1, y_2, ..., y_n)$ a random sample with n observations distributed as $\mathcal{D}$. This distribution has k parameters, i.e, it has a probability (density) function $f(y_i|\theta_{i1}, \theta_{i2}, \ldots, \theta_{ik})$, where θ_{ij} is the value of the j^{th} parameter for the i^{th} observation. A regression model with relaxation of the distributional assumption can be expressed as follows:

$$y_i \sim \mathcal{D}(\theta_{i1}, \theta_{i2}, \ldots, \theta_{ij}, \ldots, \theta_{ik}) \tag{6-1}$$

$$g(\boldsymbol{\theta}_j) = \boldsymbol{\eta}_j = \mathbf{X}_j^\top \boldsymbol{\beta}_j. \tag{6-2}$$

$g_k(\cdot)$ is the k-th link function, $\boldsymbol{\eta}_j$ is the value of the linear predictor for the j^{th} for all observations, $\boldsymbol{\beta}_j = (\beta_{0j}, \beta_{1j}, \ldots, \beta_{(p_j-1)j})^\top$ are the fixed effects vector, where p_j is the number of parameters in linear predictor j, $\mathbf{X}_j$ is a known design matrix of order $n \times p_j$ and y_i is the target variable.

6.2. Estimation

Let be $\boldsymbol{\beta} = (\beta_{01}, \dots, \beta_{(p_1-1)1}, \beta_{02}, \dots, \beta_{(p_2-1)2}, \dots, \beta_{0k}, \dots, \beta_{(p_k-1)k})^\top$ a vector that contains all the regression parameters and $\boldsymbol{y}$ a vector with all the observations y_i. To find the maximum likelihood (ML) estimators of $\boldsymbol{\beta}$, we apply the MLE method

$$\hat{\boldsymbol{\beta}} = \underset{\boldsymbol{\beta} \in B}{\arg \text{máx}}\ l(\boldsymbol{\beta}|\boldsymbol{y}), \tag{6-3}$$

where $l(\boldsymbol{\beta}|\boldsymbol{y})$ is the log-likelihood function of the regression parameters. The values η_{ij} and θ_{ij} can be computed if $\boldsymbol{\beta}$ is known. The function is computed as follows (Turkson et al., 2021)

$$l(\boldsymbol{\beta}|\boldsymbol{y}) = \log\left(\prod_{i=1}^{n} [f(y_i|\boldsymbol{\beta})]^{R_{i1}}\ [S(y_i|\boldsymbol{\beta})]^{R_{i2}}\ [F(y_i|\boldsymbol{\beta})]^{R_{i3}}\right), \tag{6-4}$$

where $f(y_i|\theta_{i1}, \theta_{i2}, \dots, \theta_{ik}) = f(\boldsymbol{y}|\boldsymbol{\beta})$ is the probability density function of $\mathcal{D}$ in terms of the regression parameters. Similarly, $S(\boldsymbol{y}|\boldsymbol{\beta})$ and $F(\boldsymbol{y}|\boldsymbol{\beta})$ are the survival and the cumulative density functions, and R_{il} is an element of the status matrix $\boldsymbol{R}$ defined as follows:

$$R_{il} = \begin{cases} 1, & \text{if sample unit } i \text{ has status } l, \\ 0, & \text{in other case,} \end{cases} \tag{6-5}$$

with $i = 1, 2, \dots, n$ and $l = 1, 2, 3$ ($l = 1$: observation status, $l = 2$: right censorship status, $l = 3$: left censorship status).

6.3. Standard Error

The variance-covariance matrix of the ML estimators is given by the inverse observed Fisher's Information Matrix Pawitan (2013b), as follows:

$$\text{Var}\,(\hat{\boldsymbol{\beta}}) = \left[\mathcal{J}(\hat{\boldsymbol{\beta}})\right]^{-1} = C(\hat{\boldsymbol{\beta}}), \tag{6-6}$$

where $\mathcal{J}(\hat{\boldsymbol{\beta}})$ is the observed Fisher information matrix. The standard errors can be calculated as the square root of the diagonal elements of matrix C (Pawitan, 2013b)

$$S.E(\hat{\boldsymbol{\beta}}) = \sqrt{C_{jj}(\hat{\boldsymbol{\beta}})}. \tag{6-7}$$

Recall the information matrix definition,

$$\mathcal{J}(\boldsymbol{\beta}) = [C(\boldsymbol{\beta})]^{-1} = -\frac{\partial^2 l(\boldsymbol{\beta}|\boldsymbol{y})}{\partial \boldsymbol{\beta}^2} = \mathbf{H}(\boldsymbol{\beta}), \tag{6-8}$$

where $\mathbf{H}(\cdot)$ is the Hessian matrix. Accordingly, the standard errors can be estimated by computing equation (6-7) with the Hessian matrix, thereby

$$S.E(\hat{\boldsymbol{\beta}}) = \sqrt{\mathbf{H}_{jj}^{-1}(\hat{\boldsymbol{\beta}})}. \tag{6-9}$$

Alternatively, standard errors can be calculated using Bootstrap.

6.4. Basic usage and features

`maxlogLreg` is a function implemented in **EstimationTools** for fitting the flexible regression models stated in section 6.1. The execution can be performed by defining at least two arguments

```
maxlogLreg(formulas, y_dist)
```

where `y_dist` is an R formula defining the distribution of the response variable ($\mathcal{D}(\boldsymbol{\theta})$) and `formulas` is a list of formulas defining the linear predictors for each parameter of the defined distribution, i.e, equations (6-2). We provided also a `summary` method since the output is an S3 object of class `maxlogL` (recall section 5.2).

Let's explore the capabilities of `maxlogLreg` taking the `lung` dataset. This study contains survival times in patients with advanced lung cancer from the North Central Cancer Treatment Group Loprinzi et al. (1994).

The fitted model is as follows:

$$\begin{align} T_i &\sim WEI(\alpha_i,\ k), \tag{6-10} \\ \log(\alpha_i) &= \beta_{01} + \beta_{11} \times ph.ecog_i + \beta_{12} \times sex_i + \beta_{13} \times age_i, \tag{6-11} \\ \log(k) &= \beta_{02}, \tag{6-12} \end{align}$$

The model uses the `dweibull` function parametrization, i.e

$$f(t|\alpha,k) = \frac{\alpha}{k}\left(\frac{t}{k}\right)^{\alpha-1}\exp\left[-\left(\frac{t}{k}\right)^{\alpha}\right], \qquad t > 0. \tag{6-13}$$

$\alpha, k > 0$ are the shape and scale parameters,, respectively. The R code to fit the model is displayed below.

```r
library(EstimationTools)

# Formulas with linear predictors
formulas <- list(scale.fo = ~ ph.ecog + sex + age, shape.fo = ~ 1)

lung_data <- na.omit(
  lung[, c("time", "status", "ph.ecog", "sex", "age")]
)
lung_data$sex <- as.factor(lung_data$sex)

# The model
fit_wei1 <- maxlogLreg(
  formulas,
  data = lung_data,
  y_dist = Surv(time, status) ~ dweibull,
  link = list(
    over = c("shape", "scale"), fun = rep("log_link", 2)
  )
)
```

EstimationTools provides a `summary` method for objects of `maxlogL` class. It displays AIC (Akaike Information Criterion), BIC (Bayesian Information Criterion), estimates, their standard errors, the optimisation routine selected by the user, and the method used for the computation of the standard error.

```r
summary(fit_wei1)

## ---------------------------------------------------------------
## Optimization routine: nlminb
## Standard Error calculation: Hessian from optim
## ---------------------------------------------------------------
##        AIC      BIC
##   2274.878 2292.002
## ---------------------------------------------------------------
## Fixed effects for log(shape)
## ---------------------------------------------------------------
##             Estimate Std. Error Z value Pr(>|z|)
## (Intercept) 0.313193   0.061459  5.0959 3.47e-07 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## ---------------------------------------------------------------
## Fixed effects for log(scale)
## ---------------------------------------------------------------
##               Estimate Std. Error Z value  Pr(>|z|)
## (Intercept)  6.6745266  0.4272819 15.6209 < 2.2e-16 ***
## ph.ecog     -0.3396383  0.0834348 -4.0707 4.687e-05 ***
## sex2         0.4010900  0.1237365  3.2415  0.001189 **
## age         -0.0074754  0.0067519 -1.1072  0.268223
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## ---------------------------------------------------------------
## Note: p-values valid under asymptotic normality of estimators
## ---
```

`fit_wei1` estimates are consistent with those got with `survreg()`. Further details can be found in *Parametric regression model for survival data: Weibull regression model as an example* (Zhang, 2016).

`maxlogLreg` allows the user can also to customize the bounds and initial guesses. Let's try again to fit a model with the failure data used above.

```r
# Bounds for optimization. Upper bound set with default values (Inf)
start <- list(
  scale = list(Intercept = 20, ph.ecog = 0, sex = 5, age = 0),
  shape = list(Intercept = 1)
)
lower <- list(
  scale = list(Intercept = 0, ph.ecog = -5, sex = 0, age = -5),
  shape = list(Intercept = 0)
)

fit_wei2 <- maxlogLreg(
  formulas,
  y_dist = Surv(time, status) ~ dweibull,
  start = start,
  lower = lower,
  data = lung_data,
  link = list(
    over = c("shape", "scale"), fun = rep("log_link", 2)
  )
)
summary(fit_wei2)
```

```
## _______________________________________________________________
## Optimization routine: nlminb
## Standard Error calculation: Hessian from optim
## _______________________________________________________________
##        AIC      BIC
##   2274.878 2292.002
## _______________________________________________________________
## Fixed effects for log(shape)
## ---------------------------------------------------------------
##             Estimate Std. Error Z value  Pr(>|z|)
## (Intercept) 0.313192   0.061459  5.0959 3.471e-07 ***
## ---
## Signif. codes:  0 ’***’ 0.001 ’**’ 0.01 ’*’ 0.05 ’.’ 0.1 ’ ’ 1
## _______________________________________________________________
## Fixed effects for log(scale)
## ---------------------------------------------------------------
##               Estimate Std. Error Z value  Pr(>|z|)
## (Intercept)  6.6745213  0.4272819 15.6209 < 2.2e-16 ***
## ph.ecog     -0.3396384  0.0834349 -4.0707 4.687e-05 ***
## sex2         0.4010902  0.1237366  3.2415  0.001189 **
## age         -0.0074754  0.0067519 -1.1072  0.268227
## ---
## Signif. codes:  0 ’***’ 0.001 ’**’ 0.01 ’*’ 0.05 ’.’ 0.1 ’ ’ 1
## _______________________________________________________________
## Note: p-values valid under asymptotic normality of estimators
## ---
```

As usual in R objects, we have provided some useful methods, such `coef`

```
coef(fit_wei2, parameter = ’scale’)
```

```
##  (Intercept)      ph.ecog         sex2          age
##  6.674521289 -0.339638368  0.401090223 -0.007475383
```

```
coef(fit_wei2, parameter = ’shape’)
```

```
## (Intercept)
##   0.3131917
```

and a `coefMany` to obtain the coefficients of all available linear predictors in the fitted model

```
coefMany(fit_wei2)

## $shape
## (Intercept)
##   0.3131917
##
## $scale
##  (Intercept)      ph.ecog         sex2          age
##  6.674521289 -0.339638368  0.401090223 -0.007475383
```

6.5. Standard Error of predictions

We also provided a `predict` method, which computes predictions for values of any distribution parameter in link or original scale. For example, to predict values for `shape` parameter in the link scale, we must run the following lines:

```
head(predict(fit_wei2, parameter = ’shape’), n = 6)

##         1         2         3         4         5         6
## 0.3131917 0.3131917 0.3131917 0.3131917 0.3131917 0.3131917
```

but predictions in original (`'response'`) scale

```
head(predict(fit_wei2, parameter = ’shape’, type = ’response’), n = 6)

##        1        2        3        4        5        6
## 1.367784 1.367784 1.367784 1.367784 1.367784 1.367784
```

In both cases, the standard error of prediction can be computed.

```
link_shape <- predict(fit_wei2, parameter = ’shape’, se.fit = TRUE)
head(link_shape$se.fit, n = 6)

##          1          2          3          4          5          6
## 0.06145935 0.06145935 0.06145935 0.06145935 0.06145935 0.06145935

response_shape <- predict(fit_wei2, parameter = ’shape’,
                          type = ’response’, se.fit = TRUE)
head(response_shape$se.fit, n = 6)

##         1         2         3         4         5         6
## 0.2413286 0.2413286 0.2413286 0.2413286 0.2413286 0.2413286
```

Actually, `predict` method computes the standard error using asymptotic approximation for the `'link'` scale

$$\mathrm{Var}\,(\hat{\boldsymbol{\eta}}_j) = \mathbf{X}_j \mathrm{Var}\,(\hat{\boldsymbol{\beta}}_j)\mathbf{X}_j^{\top}, \tag{6-14}$$

whereas the values for the `'response'` scale uses the Delta method. The approximation yields the following expression:

$$\mathrm{Var}\,(\hat{\boldsymbol{\theta}}_j) = \mathbf{D}_j \mathrm{Var}\,(\hat{\boldsymbol{\eta}}_j)\mathbf{D}_j, \tag{6-15}$$

where $\mathbf{D}_j$ is a diagonal matrix with elements $\partial\theta_{ij}/\partial\eta_{ij}$.

6.6. Ilustrative examples

Two datasets will be considered to illustrate how to fit time-to-event data using the `maxlogLreg` framework. Regression analysis was conducted using Weibull and/or exponentiated Weibull distributions.

6.6.1. Head and neck cancer

Let's revisit the head and neck cancer data from section 1.1.2. The dataset is stored in the **EstimationTools** package

```r
library(EstimationTools)
data("head_neck_cancer")
head(head_neck_cancer, n = 3)
```

```
##   Time Therapy Status
## 1    7       1      1
## 2   34       1      1
## 3   42       1      1
```

Weibull model

Let's fit the traditional two-parameter Weibull (WEI) distribution. Recall the pdf

$$f(t|\rho,\kappa) = \kappa\rho\,(\rho t)^{\kappa-1}\exp\left[-(\rho t)^{\kappa}\right], \qquad t > 0, \tag{6-16}$$

The pdf above must be included in the R Global Environment, so let's run the following code cell:

```r
dW <- function(x, rho, k, log = FALSE) {
  if (any(x < 0))
    stop(paste("x must be positive", "\n", ""))
  if (any(rho <= 0))
    stop(paste("rho must be positive", "\n", ""))
  if (any(k <= 0))
    stop(paste("k must be positive", "\n", ""))
  prod1 <- mpfr(-(rho*x)^k, 500)
  prod2 <- mpfr((rho*x)^(k-1), 500)
  f <- k*rho*prod2*exp(prod1)
  if (log == FALSE)
    density <- f
  else density <- log(f)
  return(density)
}
```

and run the following to add the CDF

```r
pW <- function(q, rho, k, lower.tail = TRUE, log.p = FALSE) {
  if (any(q < 0))
    stop(paste("q must be positive", "\n", ""))
  if (any(rho <= 0))
    stop(paste("rho must be positive", "\n", ""))
  if (any(k <= 0))
    stop(paste("k must be positive", "\n", ""))
  prod <- -(rho*q)^k
  cdf <- -expm1(prod)

  if (lower.tail == TRUE)
    cdf <- cdf
  else cdf <- 1 - cdf
  if (log.p == FALSE)
    cdf <- cdf
  else cdf <- log(cdf)
  cdf
}
```

The corresponding model equation is showed below:

$$T_i \sim WEI(\rho_i,\, \kappa), \tag{6-17}$$

$$\log(\rho) = \beta_{01} + \beta_{11} x_i, \tag{6-18}$$

$$\log(\kappa) = \beta_{02}. \tag{6-19}$$

with $\rho_i,\, \kappa > 0$ and

$$x_i = \begin{cases} 1 & \text{if subject } i \text{ receives the radiation therapy only,} \\ 0 & \text{if subject } i \text{ receives radiation plus chemotherap.} \end{cases} \tag{6-20}$$

This could be specified in R using our framework

```
distribution <- Surv(Time, Status) ~ dW
support <- list(interval = c(0, Inf), type = ’continuous’)
formulas <- list(rho.fo = ~ Therapy, k.fo = ~ 1)
links <- list(over = c("rho", "k"), fun = rep("log_link", 2))
```

Initial values and boundaries for numerical optimisation must be established

```
start_vaues <- list(
  rho = list(Intercept = 0, Therapy = 0),
  k = list(Intercept = -1)
)

lower <- list(
  rho = list(Intercept = -20, Therapy = -5),
  k = list(Intercept = -5)
)
```

Finally, let’s run the estimation

```
mod_wei_hncd <- maxlogLreg(
  formulas = formulas,
  y_dist = distribution,
  link = links,
  support = support,
  start = start_vaues,
  lower = lower,
  data = head_neck_cancer
)
```

```
summary(mod_wei_hncd)

## ---------------------------------------------------------------
## Optimization routine: nlminb
## Standard Error calculation: Hessian from optim
## ---------------------------------------------------------------
##        AIC      BIC
##   1082.519 1090.212
## ---------------------------------------------------------------
## Fixed effects for log(rho)
## ---------------------------------------------------------------
##              Estimate Std. Error  Z value  Pr(>|z|)
## (Intercept) -6.82476    0.21116 -32.3204 < 2.2e-16 ***
## Therapy      0.78603    0.27888   2.8185  0.004824 **
## ---
## Signif. codes:  0 ’***’ 0.001 ’**’ 0.01 ’*’ 0.05 ’.’ 0.1 ’ ’ 1
## ---------------------------------------------------------------
## Fixed effects for log(k)
## ---------------------------------------------------------------
##              Estimate Std. Error Z value Pr(>|z|)
## (Intercept) -0.16185    0.09212  -1.757  0.07892 .
## ---
## Signif. codes:  0 ’***’ 0.001 ’**’ 0.01 ’*’ 0.05 ’.’ 0.1 ’ ’ 1
## ---------------------------------------------------------------
## Note: p-values valid under asymptotic normality of estimators
## ---
```

Exponentiated Weibull model

Let’s estimate the exponentiated Weibull (EW) regression model. Recall the PDF

$$f(t|\rho,\kappa,\gamma) = \kappa\rho\gamma\left(\rho t\right)^{\kappa-1}\exp\left[-\left(\rho t\right)^{\kappa}\right]\left\{1-\exp\left[-\left(\rho t\right)^{\kappa}\right]\right\}^{\gamma-1}, \qquad t>0. \tag{6-21}$$

As before, we need the pdf and cdf in the R Global Environment

```
library(Rmpfr)

dEW <- function(x, rho, k, gam, log = FALSE) {
  if (any(x < 0))
    stop(paste("x must be positive", "\n", ""))
  if (any(rho <= 0))
    stop(paste("rho must be positive", "\n", ""))
  if (any(k <= 0))
    stop(paste("k must be positive", "\n", ""))
  if (any(gam <= 0))
    stop(paste("gam must be positive", "\n", ""))
  prod1 <- mpfr(-(rho*x)^k, 500)
  prod2 <- mpfr((rho*x)^(k-1), 500)
  f <- k*rho*gam*prod2*exp(prod1)*( -expm1(prod1) )^(gam-1)
  if (log == FALSE)
    density <- f
  else density <- log(f)
  return(density)
}

pEW <- function(q, rho, k, gam, lower.tail = TRUE, log.p = FALSE) {
  if (any(q < 0))
    stop(paste("q must be positive", "\n", ""))
  if (any(rho <= 0))
    stop(paste("rho must be positive", "\n", ""))
  if (any(k <= 0))
    stop(paste("k must be positive", "\n", ""))
  if (any(gam <= 0))
    stop(paste("gam must be positive", "\n", ""))

  prod <- mpfr(-(rho*q)^k, 500)
  cdf <- (-expm1(prod))^gam

  if (lower.tail == TRUE)
    cdf <- cdf
  else cdf <- 1 - cdf
  if (log.p == FALSE)
    cdf <- cdf
  else cdf <- log(cdf)
  cdf
}
```

Retrieve the model equation

$$
\begin{aligned}
T_i &\sim EW(\rho_i,\ \kappa,\ \gamma), && \text{(6-22)}\\
\log(\rho_i) &= \beta_{01} + \beta_{11} x_i, && \text{(6-23)}\\
\log(\kappa) &= \beta_{02}, && \text{(6-24)}\\
\log(\gamma) &= \beta_{03}, && \text{(6-25)}
\end{aligned}
$$

with ρ_i, κ, $\gamma > 0$ and x_i is the indicator variable for the treatment. Let’s define the model

```
distribution <- Surv(Time, Status) ~ dEW
support <- list(interval = c(0, Inf), type = ’continuous’)
formulas <- list(rho.fo = ~ Therapy, k.fo = ~ 1, gam.fo = ~ 1)
links <- list(over = c("rho", "k", "gam"), fun = rep("log_link", 3))
```

Set configuration options

```
start_vaues <- list(
  rho = list(Intercept = 5, Therapy = 0),
  k = list(Intercept = -1),
  gam = list(Intercept = 0)
)

lower <- list(
  rho = list(Intercept = 0, Therapy = 0),
  k = list(Intercept = -10),
  gam = list(Intercept = 0)
)

upper <- list(
  rho = list(Intercept = 20, Therapy = 5),
  k = list(Intercept = 10),
  gam = list(Intercept = 10)
)
```

Fit the model

```
mod_ew_hncd <- maxlogLreg(
  formulas = formulas,
  y_dist = distribution,
  link = links,
  support = support,
  start = start_vaues,
  lower = lower,
  data = head_neck_cancer
)

summary(mod_ew_hncd)
```

```
## ---------------------------------------------------------------
## Optimization routine: nlminb
## Standard Error calculation: Hessian from optim
## ---------------------------------------------------------------
##        AIC      BIC
##   1064.556 1074.813
## ---------------------------------------------------------------
## Fixed effects for log(rho)
## ---------------------------------------------------------------
##             Estimate Std. Error Z value Pr(>|z|)
## (Intercept) 10.18305   15.48487  0.6576  0.51079
## Therapy      0.56090    0.26728  2.0986  0.03585 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## ---------------------------------------------------------------
## Fixed effects for log(k)
## ---------------------------------------------------------------
##             Estimate Std. Error Z value Pr(>|z|)
## (Intercept) -2.11601    0.64903 -3.2603 0.001113 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## ---------------------------------------------------------------
## Fixed effects for log(gam)
## ---------------------------------------------------------------
##             Estimate Std. Error Z value Pr(>|z|)
## (Intercept)   6.7084     4.2295  1.5861   0.1127
## ---------------------------------------------------------------
## Note: p-values valid under asymptotic normality of estimators
## ---
```

```
AIC(mod_ew_hncd, mod_wei_hncd)
```

```
##              df      AIC
## mod_ew_hncd   4 1064.556
## mod_wei_hncd  3 1082.519
```

6.6.2. Survival of patients with leukemia

Consider a partial version of the dataset from the Acute Lymphoblastic Leukemia (ALL) clinic trial employed by Colosimo and Ruiz Giolo (2006b), but firstly analysed by Lawless (2002). Let’s call it the ALL dataset.

The dataset contains survival times, in weeks, of 17 patients with acute leukemia. For these patients, their white blood cell counts (WBC) were recorded at the time of diagnosis.

```
library(EstimationTools)
data(ALL_colosimo_table_4_1)
head(ALL_colosimo_table_4_1, n = 3)
```

```
##   times status  wbc lwbc
## 1    65      1 2300 3.36
## 2   156      1  750 2.88
## 3   100      1 4300 3.63
```

Weibull model

Consider a Weibull model to fit these data, regarding the following parametrization:

$$f(t|\alpha,\,\sigma) = \frac{\alpha}{\sigma}\left(\frac{t}{\sigma}\right)^{\alpha-1}\exp\left(-\frac{t}{\sigma}^{\alpha}\right), \tag{6-26}$$

with and without the log transformation over the predictor.

```
distribution <- Surv(times, status) ~ dweibull
support <- list(interval = c(0, Inf), type = ’continuous’)

formulas <- list(
  shape.fo = ~ lwbc,
  scale.fo = ~ 1
)

links <- list(
  over = c("shape", "scale"),
  fun = rep("log_link", 2)
)


model_wei_all_log <- maxlogLreg(
  formulas = formulas,
  y_dist = distribution,
  support = support,
  link = links,
  data = ALL_colosimo_table_4_1
)

summary(model_wei_all_log)
```

```
## ---------------------------------------------------------------
## Optimization routine: nlminb
## Standard Error calculation: Hessian from optim
## ---------------------------------------------------------------
##        AIC      BIC
##   174.1553 176.6549
## ---------------------------------------------------------------
## Fixed effects for log(shape)
## ---------------------------------------------------------------
##             Estimate Std. Error Z value Pr(>|z|)
## (Intercept)  4.40584    1.82606  2.4128  0.01583 *
## lwbc        -1.06469    0.44001 -2.4197  0.01553 *
## ---
## Signif. codes:  0 ’***’ 0.001 ’**’ 0.01 ’*’ 0.05 ’.’ 0.1 ’ ’ 1
## ---------------------------------------------------------------
## Fixed effects for log(scale)
## ---------------------------------------------------------------
##             Estimate Std. Error Z value  Pr(>|z|)
## (Intercept)  4.76840    0.22127   21.55 < 2.2e-16 ***
## ---
## Signif. codes:  0 ’***’ 0.001 ’**’ 0.01 ’*’ 0.05 ’.’ 0.1 ’ ’ 1
## ---------------------------------------------------------------
## Note: p-values valid under asymptotic normality of estimators
## ---
```

Exponential model

Next, consider exponential models. The corresponding pdf can be derived from the Weibull distribution by making the shape parameter $\alpha = 1$; hence,

$$f(t|\sigma) = \frac{1}{\sigma} \exp\left(-\frac{t}{\sigma}\right) \tag{6-27}$$

```
formulas <- list(
  scale.fo = ~ lwbc
)
fixed <- list(shape = 1)

links <- list(
  over = "scale",
  fun = "log_link"
)

lower <- list(
  scale = list(Intercept = -30, wbc = -30)
)

model_exp_all_log <- maxlogLreg(
  formulas = formulas,
  y_dist = distribution,
  fixed = fixed,
  support = support,
  link = links,
  data = ALL_colosimo_table_4_1
)

summary(model_exp_all_log)

## ---------------------------------------------------------------
## Optimization routine: nlminb
## Standard Error calculation: Hessian from optim
## ---------------------------------------------------------------
##        AIC      BIC
##   171.7541 173.4205
## ---------------------------------------------------------------
## Fixed effects for log(scale)
## ---------------------------------------------------------------
##              Estimate Std. Error Z value  Pr(>|z|)
## (Intercept)  8.47750    1.71122  4.9541 7.268e-07 ***
## lwbc        -1.10930    0.41357 -2.6822  0.007313 **
## ---
## Signif. codes:  0 ’***’ 0.001 ’**’ 0.01 ’*’ 0.05 ’.’ 0.1 ’ ’ 1
## ---------------------------------------------------------------
## Note: p-values valid under asymptotic normality of estimators
## ---
```

Finally, we compare the models

```
AIC(
  model_exp_all_log,
  model_wei_all_log
)
```

```
##                   df      AIC
## model_exp_all_log  2 171.7541
## model_wei_all_log  3 174.1553
```

According to Colosimo and Ruiz Giolo (2006b), the exponential model is the best. Plausibly `model_exp_all_log` is the best model for AIC.

6.7. Conclusions

With the `maxlogLreg` function we have implemented the usual estimation workflow for maximization for maximum likelihood estimation in R. Consequently, we enable users to fit regression models using any distribution with a highly readable syntax and the usual R methods known by the community such as `summary`, `predict` and `coef`.

7. Diagnostic tools for parametric survival models

> *The best is the enemy of the good.*
>
> — **Voltaire**
>
> (Writer, historian, and philosopher)

Model validation and evaluation is a key step in data analysis to assess the reliability of the estimates. Typically, residuals have been used for this purpose because they are feasible for checking the goodness-of-fit of a model and identifying potential outliers and influential observations to a fitted model Halabi et al. (2020). Several flavours of residuals for survival analysis have been proposed: the Cox-Snell to cheque the adequacy of models fitted with right censored data (Cox and Snell, 1968) and the martingale residuals for investigating the functional form of a covariate (Therneau et al., 1990; Barlow and Prentice, 1988).

Another useful approach is to search for a definition of residuals that is not dependent on the type or distribution of the response variable. Randomized quantile residuals are well suited for this purpose because they generate values that follow a normal distribution, apart from sampling variability in the estimated parameter (Dunn and Smyth, 1996). They have been successfully used in R for GLMs (Dunn and Smyth, 2018) and within the GAMLSS framework (Stasinopoulos et al., 2017a).

7.1. A working example

Consider the application example presented in section 6.6.2, which uses the ALL dataset. We will simulate a dataset comprising 1000 samples and then proceed to fit the true model (well-defined) along with two additional ill-defined models.

7.1.1. Dataset

The exponential model with a covariate is the best according to Colosimo and Ruiz Giolo (2006b), hence let's simulate a dataset based on it.

$$
\begin{aligned}
T_i &\sim WEI(\alpha,\, \sigma_i), && \text{(7-1)}\\
lwbc_i &\sim U(lwbc_{min},\, lwbc_{max}) && \text{(7-2)}\\
\alpha &= 1, && \text{(7-3)}\\
\log(\sigma_i) &= 8.47750 - 1.10930 \times lwbc_i, && \text{(7-4)}
\end{aligned}
$$

where $U(\cdot)$ is the continuous uniform distribution, $lwbc_{min}$, $lwbc_{max}$ are the minimum and maximum logarithm of white blood cells count from the original dataset and $WEI(\alpha,\, \sigma)$ is the Weibull distribution with the following parametrization

$$
f(t|\alpha,\, \sigma) = \frac{\alpha}{\sigma}\left(\frac{t}{\sigma}\right)^{\alpha-1} \exp\left(-\frac{t}{\sigma}^{\alpha}\right). \qquad \text{(7-5)}
$$

7.1.2. Model 0

Model 0 is based on the true model structure specified in equation 7-1, therefore it is defined as follows

$$
\begin{aligned}
T_i &\sim WEI(\alpha,\, \sigma), && \text{(7-6)}\\
lwbc_i &\sim U(lwbc_{min},\, lwbc_{max}) && \text{(7-7)}\\
\alpha &= 1, && \text{(7-8)}\\
\log(\sigma) &= \beta_0 + \beta_1 \times lwbc_i. && \text{(7-9)}
\end{aligned}
$$

7.1.3. Model 1

Model 1 is a poorly defined model. It is an exponential with no covariates.

$$
\begin{aligned}
T_i &\overset{\text{iid.}}{\sim} WEI(\alpha,\, \sigma), && \text{(7-10)}\\
\alpha &= 1, && \text{(7-11)}\\
\log(\sigma) &= \beta_0, && \text{(7-12)}
\end{aligned}
$$

7.1.4. Model 2

This corresponds to an exponential model in terms of `wbc^0.5`, i.e the squared root of white blood cell counts.

$$T_i \sim WEI(\alpha,\ \sigma), \tag{7-13}$$

$$wbc_i \sim U(wbc_{min},\ wbc_{max}) \tag{7-14}$$

$$\alpha = 1, \tag{7-15}$$

$$\log(\sigma) = \beta_0 + \beta_1 wbc_i^{0.5}, \tag{7-16}$$

where wbc_{min}, wbc_{max} are the minimum white blood cells counts from the original dataset. Similarly, a `sim_exp_model_2` can be obtained from a computational implementation using our `maxlogLreg` framework.

The three models are compared in table **7-1**. Model 0 is the best, as expected.

Model	Degrees of Freedom	AIC
Model 0	2	10255.16
Model 1	1	10780.34
Model 2	2	10780.16

Table 7-1: AIC for the three models implemented for the working example.

7.1.5. Code implementation

The R code for the three models is displayed in this subsection for users who prefer to follow the reading with code practise.

```r
# Dataset -----
set.seed(1112)

library(EstimationTools)
data(ALL_colosimo_table_4_1)

n <- 1000
lwbc <- ALL_colosimo_table_4_1$lwbc
lwbc <- runif(n = n, min = min(lwbc), max = max(lwbc))

status <- rep(1, n)

beta_0 <- 8.47750
beta_1 <- -1.10930
scale <- exp(beta_0 + beta_1 * lwbc)
```

```r
y <- rweibull(
  n = n,
  shape = 1,
  scale = scale
)

formulas <- list(
  scale.fo = ~ lwbc
)

# Model 0 -----
distribution <- Surv(y, status) ~ dweibull
support <- list(interval = c(0, Inf), type = ’continuous’)

formulas <- list(
  scale.fo = ~ lwbc
)
fixed <- list(shape = 1)

links <- list(
  over = "scale",
  fun = "log_link"
)

sim_exp_model_0 <- maxlogLreg(
  formulas = formulas,
  y_dist = distribution,
  fixed = fixed,
  support = support,
  link = links
)

# Model 1 -----
formulas <- list(
  scale.fo = ~ 1
)

sim_exp_model_1 <- maxlogLreg(
  formulas = formulas,
  y_dist = distribution,
  fixed = fixed,
  support = support,
  link = links
)

# Model 2 -----
wbc <- ALL_colosimo_table_4_1$wbc
wbc <- runif(n = n, min = min(wbc), max = max(wbc))

formulas <- list(
  scale.fo = ~ I(wbc^0.5)
)

sim_exp_model_2 <- maxlogLreg(
  formulas = formulas,
  y_dist = distribution,
  fixed = fixed,
  support = support,
  link = links
)
```

In the subsequent sections of this chapter, we explore the aforementioned residuals, and their implementation in **EstimationTools** package using the simulated data we presented above.

7.2. Residuals and diagnostic plots

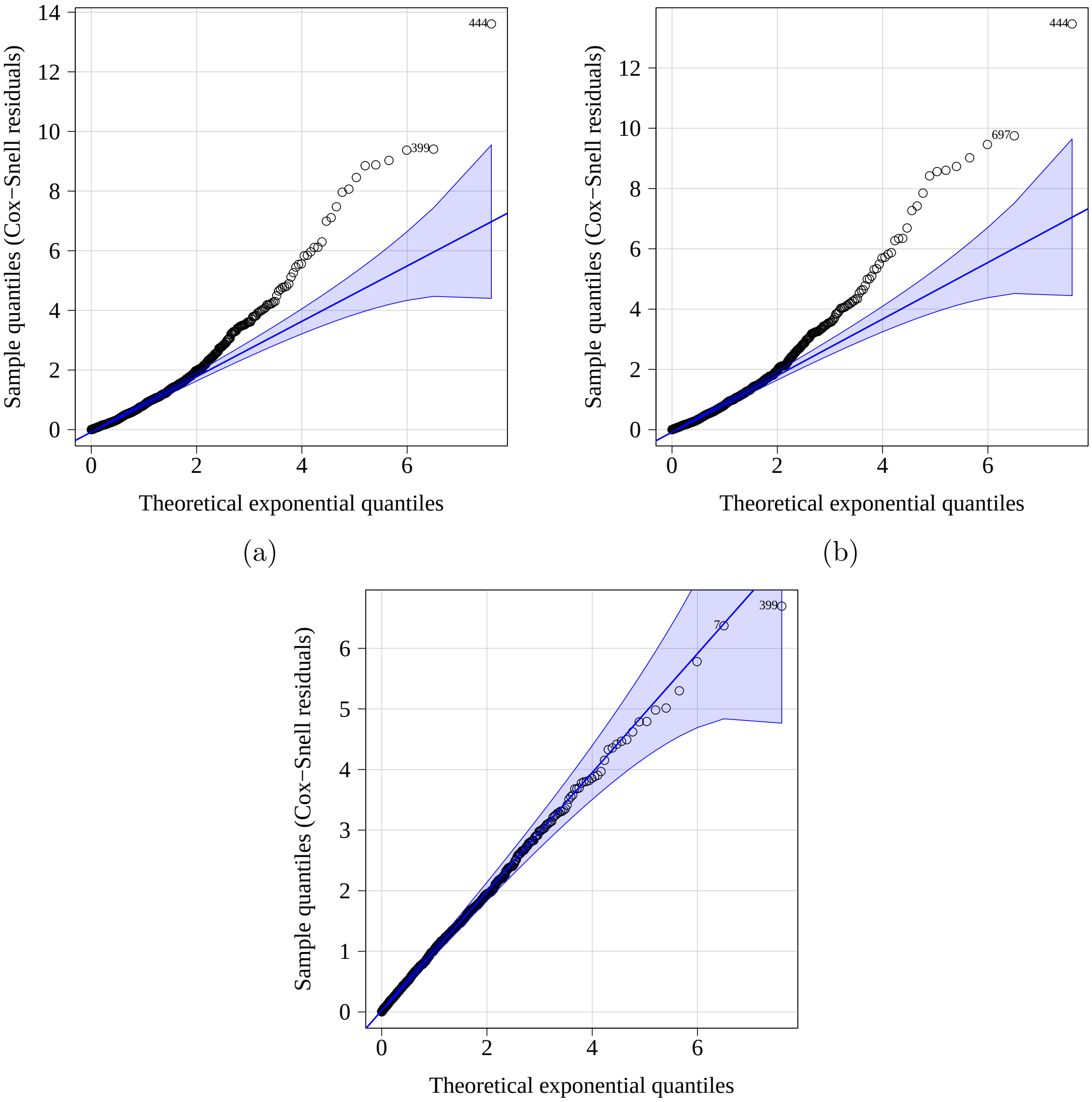
Residual-based diagnostic plots are particularly regarded for regression models. Let's state their definition and revisit their application.

7.2.1. Cox-Snell residuals

Cox and Snell (1968) introduce the concept of Cox-Snell residuals to check the adequacy of models fitted with right-censored data. They are given by the following expression:

$$r_i^{CS} = \hat{H}(t|X_i), \tag{7-17}$$

where $\hat{H}(t|X_i)$ is the estimated cumulative hazard function (Barlow and Prentice, 1988; Therneau and Grambsch, 2000b; Colosimo and Ruiz Giolo, 2006b).



(a)

(b)

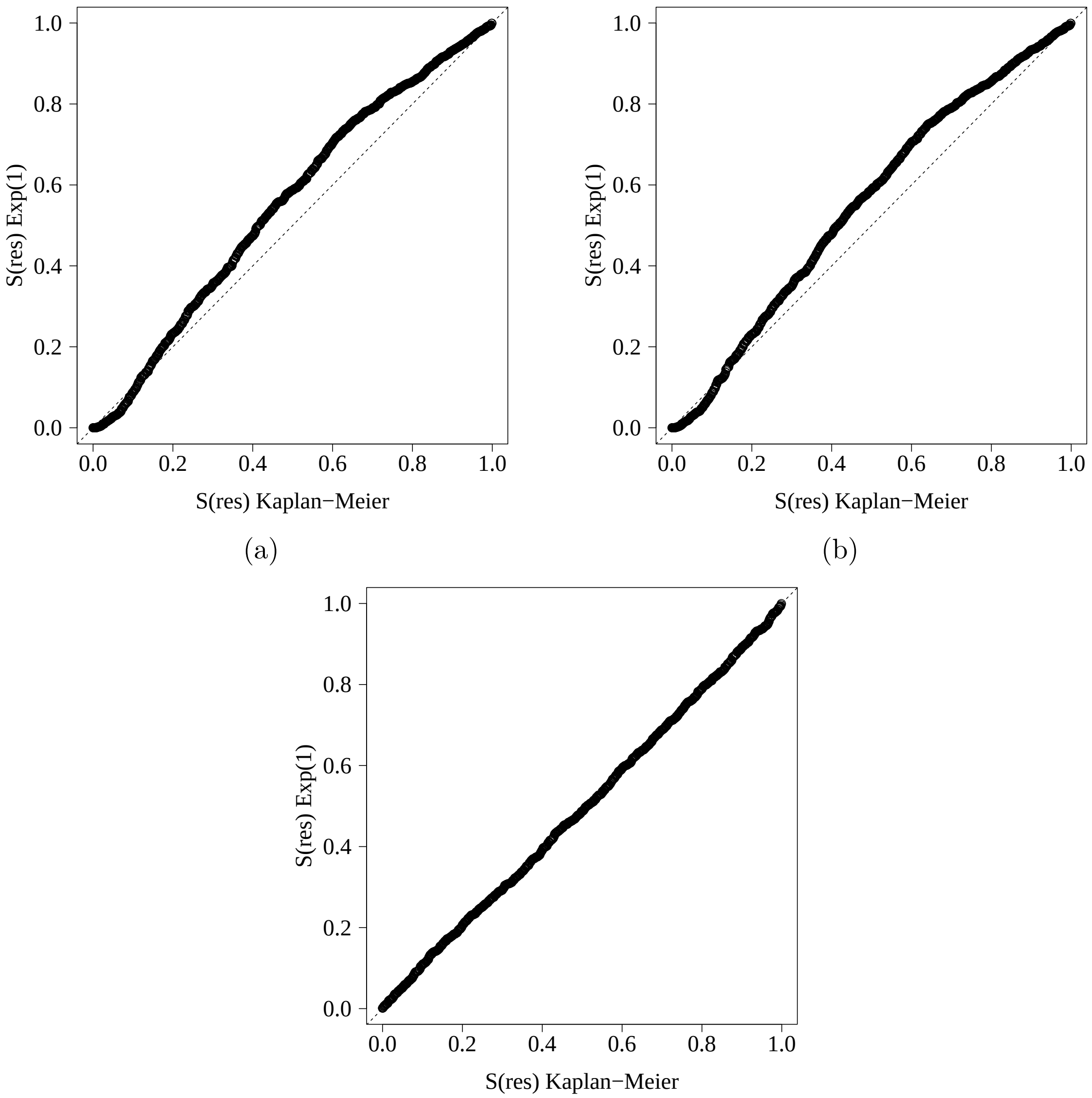
(c)

Figure 7-1: Standard exponential Q-Q plot of Cox-Snell residuals: (a) model 2; (b) model 1; (c) model 0.

Diagnostic plots have been suggested using these residuals. The most remarkable is the Q-Q plot based on the unit exponential distribution, which is motivated by the fact that if a random failure time T follows a distribution with a cumulative hazard function H, then the function $H(T)$ will be unit exponential distributed (Therneau and Grambsch, 2000a).

Plots of the corresponding Cox-Snell residuals for the models of head and neck cancer dataset fitted in section 6.6.1 can be obtained using the following commands:

```
plot(sim_exp_model_1, type = "cox-snell")
plot(sim_exp_model_2, type = "cox-snell")
plot(sim_exp_model_3, type = "cox-snell")
```



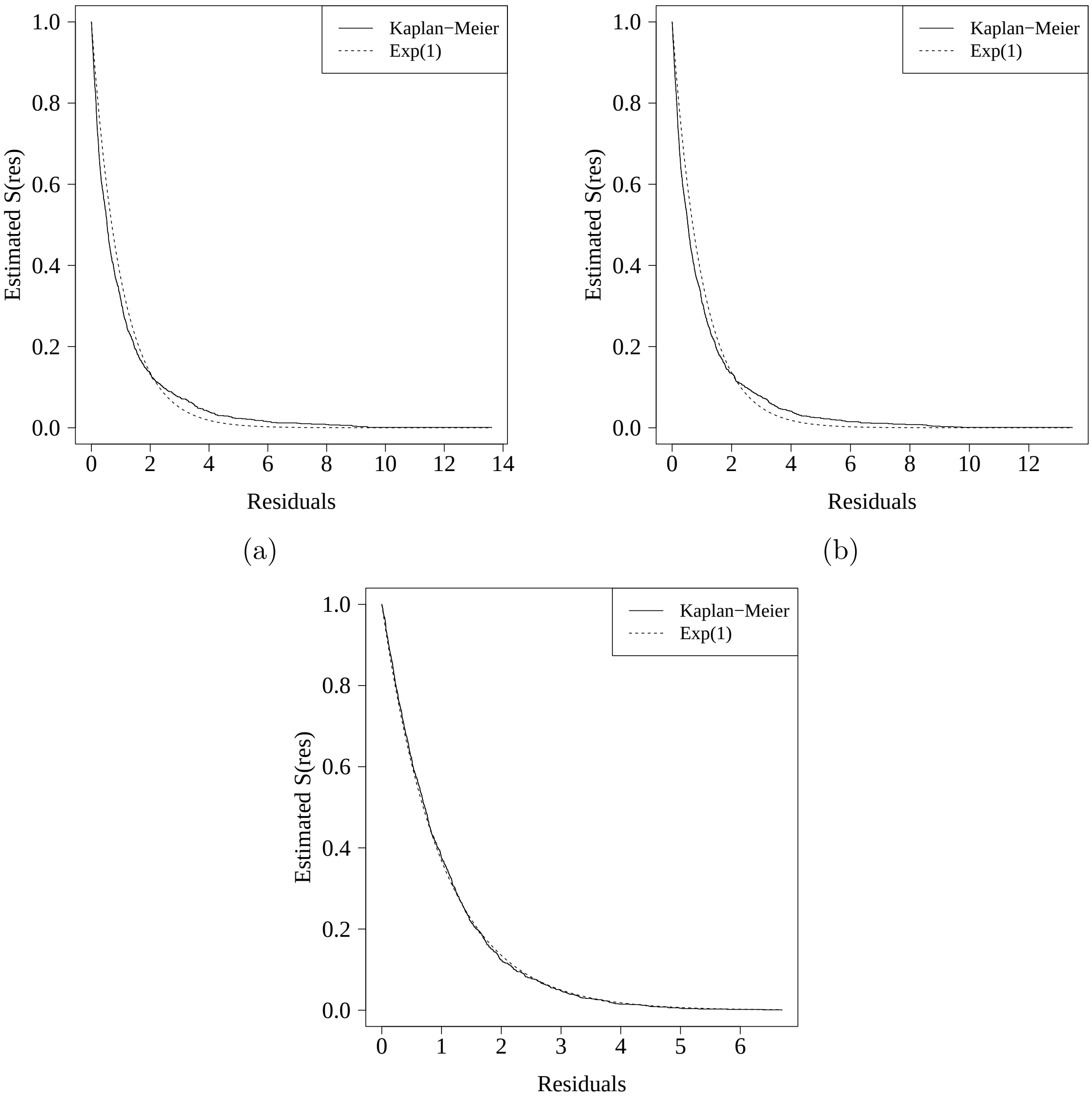
(a)

(b)

(c)

Figure 7-2: Standard exponential survival function against Kaplan-Meier estimator of residuals.: (a) model 2; (b) model 1; (c) model 0.

The behaviour of residuals for Model 0 is more appropriate, as anticipated. In contrast, residuals for both Model 1 and Model 2 deviate from the $Exp(1)$ distribution, indicating model misspecification, as depicted in Figure **7-1**. Moreover, as seen in Figure **7-2**, residuals for both Model 1 and Model 2 exhibit a more pronounced departure from the straight line when compared to the residuals for Model 0.



(c)

Figure 7-3: Kaplan-Meier estimator of residuals and theoretical exponential survival function: (a) Model 2; (b) Model 1; (c) Model 0.

Likewise, the survival function of Cox-Snell residuals is more suitable for Model 0 (figure **7-3**c) because the Kaplan-Meier estimator of residuals is much closer to the theoretical standard exponential survival function than those for Model 1 and Model 2 (see figures **7-3**b and **7-3**a).

7.2.2. Martingale residuals

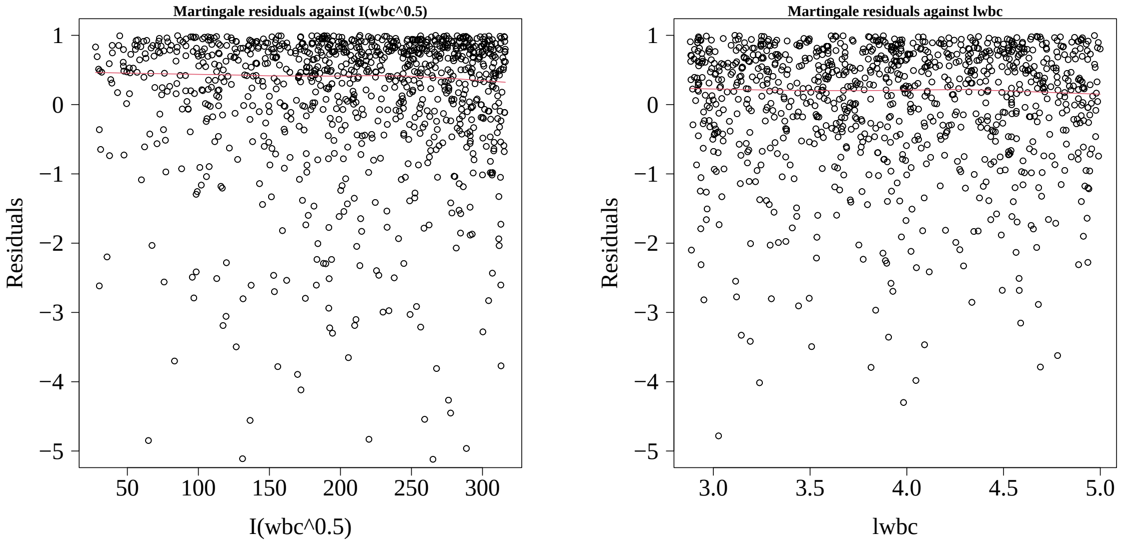
Martingale residuals were developed by Therneau et al. (1990) for survival models based on the residuals for relative risk models proposed by Barlow and Prentice (1988). They are defined by the following expression:

$$r_i^M = \delta_i - r_i^{CS}, \tag{7-18}$$

where δ_i is the failure indicator variable and r_i^{CS} are the Cox-Snell residuals. They are interpreted as the difference between the number of observed and expected events. Its interpretation is similar to that of the classic linear model: a plot of martingale residuals against a continuous covariate showing some pattern suggests a non-linear relationship (Colosimo and Ruiz Giolo, 2006b).

Let's revisit our working example and the parametric models previously fitted (section 7.1) to check the martingale residuals. They can be obtained also using the `plot()` method specifying the `xvar` argument.

```r
plot(sim_exp_model_0, type = "martingale", xvar = c("lwbc"))
plot(sim_exp_model_1, type = "martingale", xvar = c("I(wbc^0.5)"))
```



(a) Residuals for Model 2.

(b) Residuals for Model 0.

Figure 7-4: Martingale residuals. The red line represents the smoothed trend.

In a well-specified survival model, the martingale residuals against covariate plot should not show any clear pattern. as can be seen in **7-4**. Effectively, the misspecified model (Model 2 in the left side) exhibits residuals concentrated in the upper right region of the graph.

7.2.3. Deviance residuals

The deviance is a goodness-of-fit metric for statistical models. It measures the extent to which the log-likelihood of the saturated model[1] $l(\boldsymbol{\beta}_S|\boldsymbol{y})$ is different from the log-likelihood of the chosen model[2] $l(\boldsymbol{\beta}_{\hat{\mu}}|\boldsymbol{y})$. It can be computed as

$$D(\boldsymbol{y}, \hat{\boldsymbol{\mu}}) = -2\left(l(\boldsymbol{\beta}_{\hat{\mu}}|\boldsymbol{y}) - l(\boldsymbol{\beta}_S|\boldsymbol{y})\right), \tag{7-19}$$

where $\hat{\mu}_i$ is the estimated mean. The statistic $D(\boldsymbol{y}, \hat{\boldsymbol{\mu}})$ is called the deviance. An equivalent expression is as follows:

$$D(\boldsymbol{y}, \hat{\boldsymbol{\mu}}) = -2\sum_{i=1}^{n}\left[l_f(\boldsymbol{\beta}_{\hat{\mu}}|y_i) - l_f(\boldsymbol{\beta}_S|y_i)\right] = \sum_{i=1}^{n} d_i, \tag{7-20}$$

where $l_f(\cdot|y_i)$ is the logarithm of the probability density/mass function and d_i is the contribution of each value of the response variable in the dataset to the deviance. Since $l(\boldsymbol{\beta}_{\hat{\mu}}|\boldsymbol{y}) \leq l(\boldsymbol{\beta}_S|\boldsymbol{y})$, then $D(\boldsymbol{y}, \hat{\boldsymbol{\mu}}) \geq 0$ and consequently $\sum_{i=1}^{n} d_i \geq 0$ (Agresti, 2015b).

The deviance residuals are computed using d_i

$$r_i^D = \text{sign}(y_i - \hat{E}(Y_i))d_i^{1/2}, \tag{7-21}$$

Another valid expression in the context of survival analysis with right-censored data can also be obtained as a transformation to normalise the martingale residuals (Therneau et al., 1990), which produces a more symmetric configuration around the zero y-axis value in a plot of residuals against a covariate, in such a way as to facilitate outliers detection (Colosimo and Ruiz Giolo, 2006b). These right-censored deviance residuals are defined by the following expression:

$$r_i^{CD} = \text{sign}(r_i^M)\left[-2\left(r_i^M + \delta_i \log(\delta_i - r_i^M)\right)\right]^{1/2}, \tag{7-22}$$

where,

$$\text{sign}(r_i^M) = \begin{cases} -1 & \text{if } r_i^M < 0, \\ 0 & \text{if } r_i^M = 0, \\ 1 & \text{if } r_i^M > 0, \end{cases}$$

[1] The *saturated model* has a separate parameter for each observation (Agresti, 2015a).

[2] The *chosen model* has a separate parameter for each covariate selected by the analyst.

and r_i^M are the Martingale residuals. Deviance residuals used in survival analysis are similar in form to those of the Poisson regression, and analogously, they are used to measure the discrepancy of a linear model (McCullagh and Nelder, 1989). Indeed, deviance residuals are random variables approximately standard normally distributed (Pregibon, 1981; Cordeiro and NETO, 2004).

For the working example (section 7.1), the deviance residuals are shown in figure **7-5**. As usual, they can be obtained using the `plot()` method as follows:

```
plot(sim_exp_model_1, type = "right-censored-deviance")
plot(sim_exp_model_2, type = "right-censored-deviance")
plot(sim_exp_model_3, type = "right-censored-deviance")
```

Model 0 (shown in figure **7-5**c) displays a closer random pattern among the three models.

7.2.4. Randomized Quantile Residuals

Let $F(y|\theta)$ be the cumulative distribution function of the response variable y. If $F(\cdot)$ is continuous, the $F(y|\theta)$ values are uniformly distributed on $[0,\ 1]$. In this case, the randomised quantile residuals (RQR) are defined by Dunn and Smyth (1996) with the following expression:

$$r_i^q = \Phi^{-1}\left[F(y|\theta)\right], \tag{7-23}$$

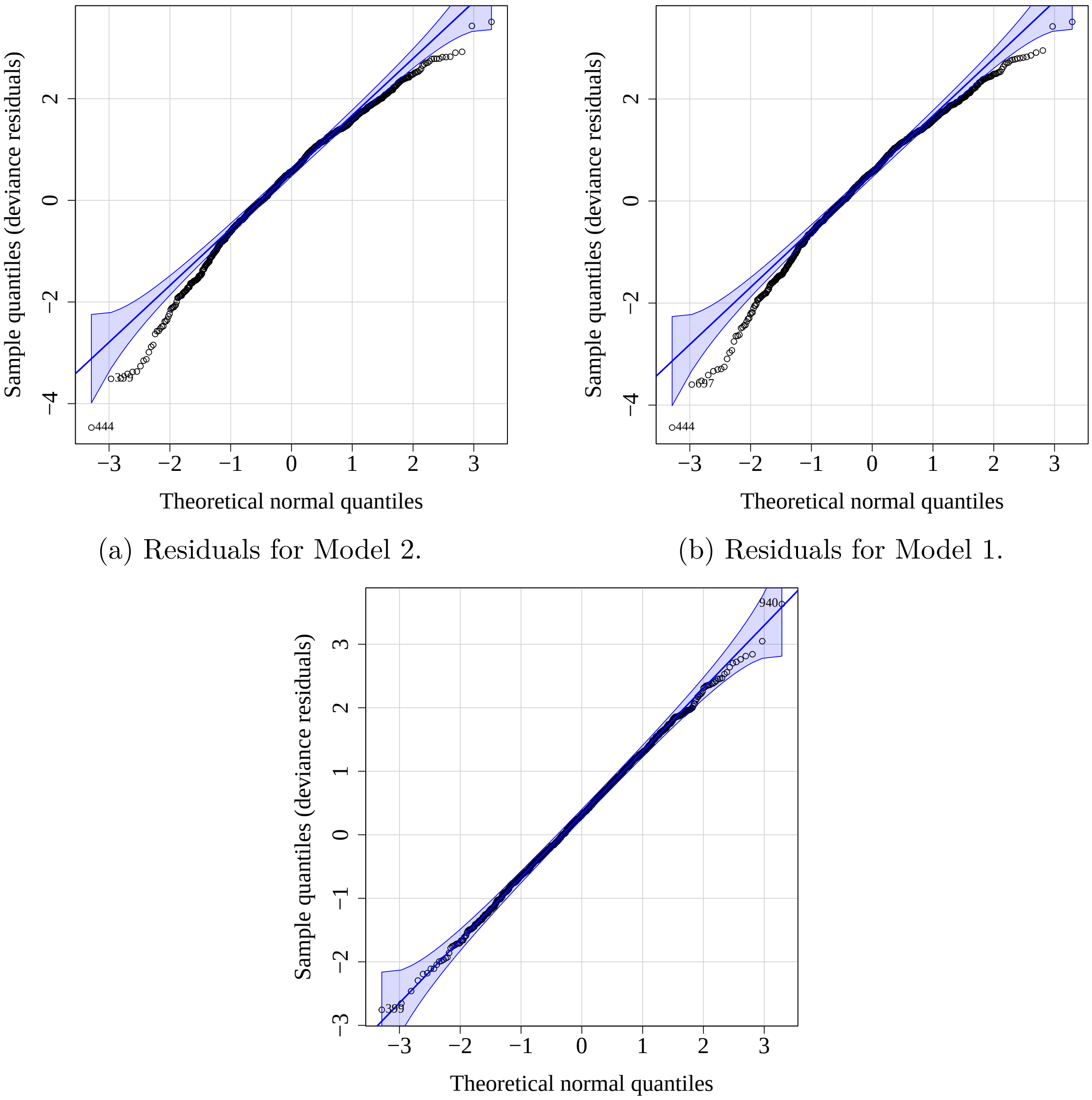
where $\Phi(\cdot)$ is the cumulative distribution function of the standard normal. Stasinopoulos et al. (2017a) plot RQR against fitted values and RQR against index and applied the usual interpretation: the plot should show a random scatter of residuals around zero, indicating that the residuals are uncorrelated with the fitted values.

Let's verify this behaviour with the working example (section 7.1) using, again, the `plot()` method. The output plots for the three models are displayed and compared in figures **??**, **7-7**a and **7-8**a.

```
plot(sim_exp_model_1, type = "rqres", parameter= "scale")
plot(sim_exp_model_2, type = "rqres", parameter= "scale")
plot(sim_exp_model_3, type = "rqres", parameter= "scale")
```

In addition to sampling variability in the parameter vector $\hat{\theta}$, r_i^q have a standard normal distribution. Accordingly, a Q-Q plot of the standard normal distribution against r_i^q is a viable diagnostic plot, as illustrated by Dunn and Smyth (1996).

RQRs against fitted values and RQRs against index plots show a significantly better adequacy for Model 0 compared to model 2 and model 1 (see figures **7-6** and **7-7**). However,



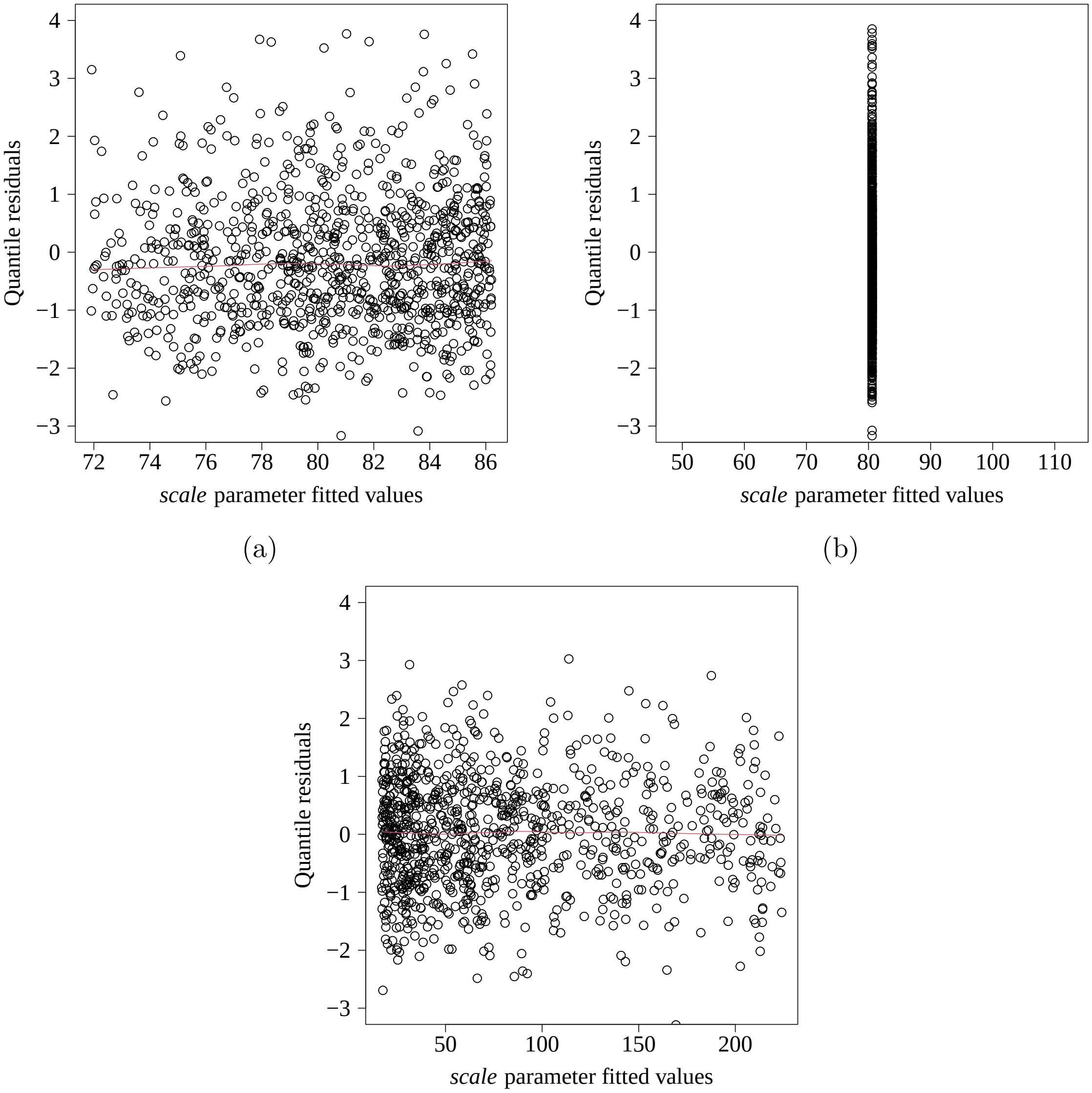
(a) Residuals for Model 2.

(b) Residuals for Model 1.

(c) Residuals for Model 0.

Figure 7-5: Right censored deviance residuals.

the normal Q-Q plot of RQRs is more conclusive: the residuals for Model 0 in figure **7-8** are nearly normal.

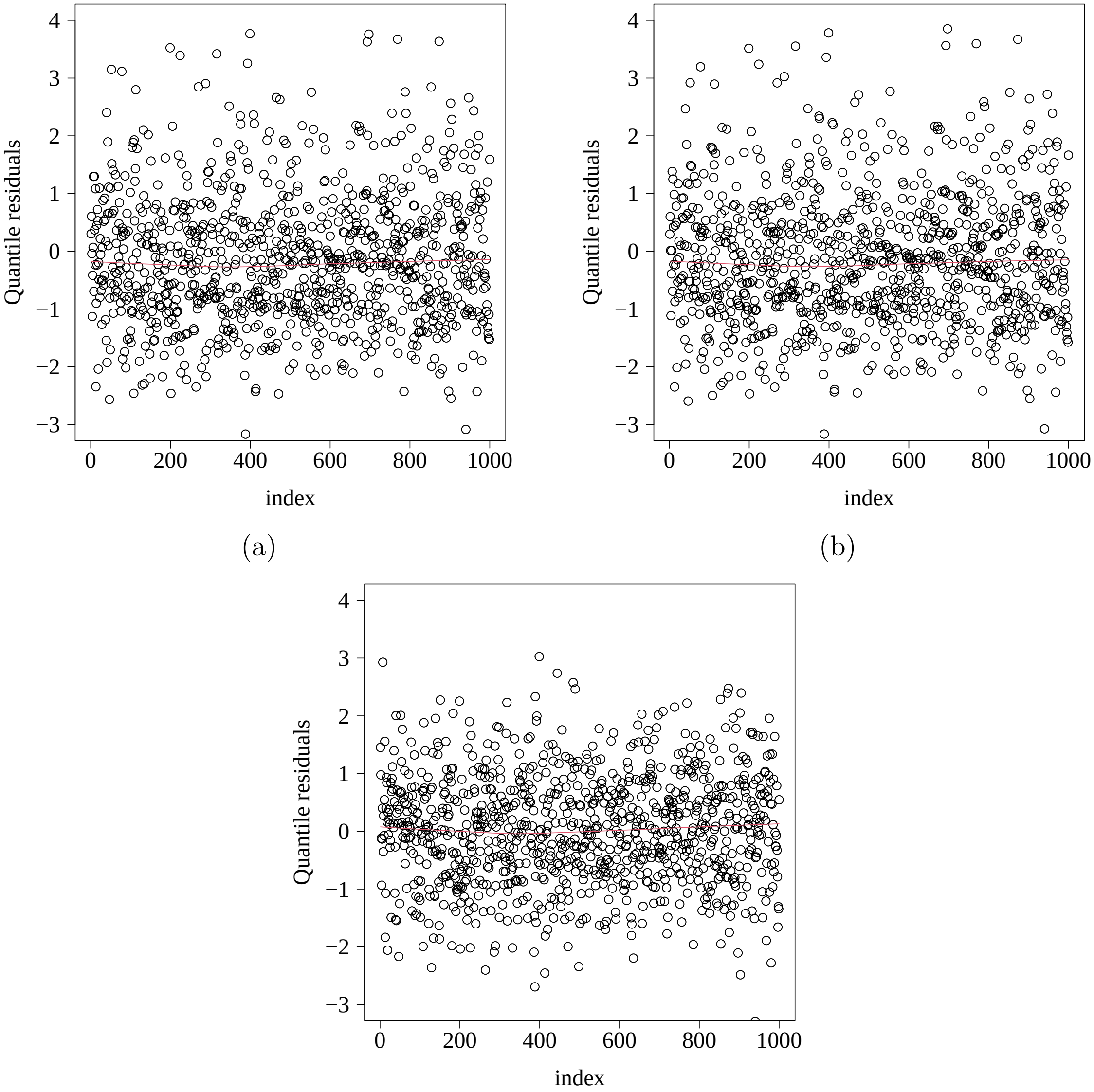


(a)

(b)

(c)

Figure 7-6: RQRs against scale parameter: (a) Model 2; (b) Model 1; (c) Model 0. The red line represents the smoothed trend.

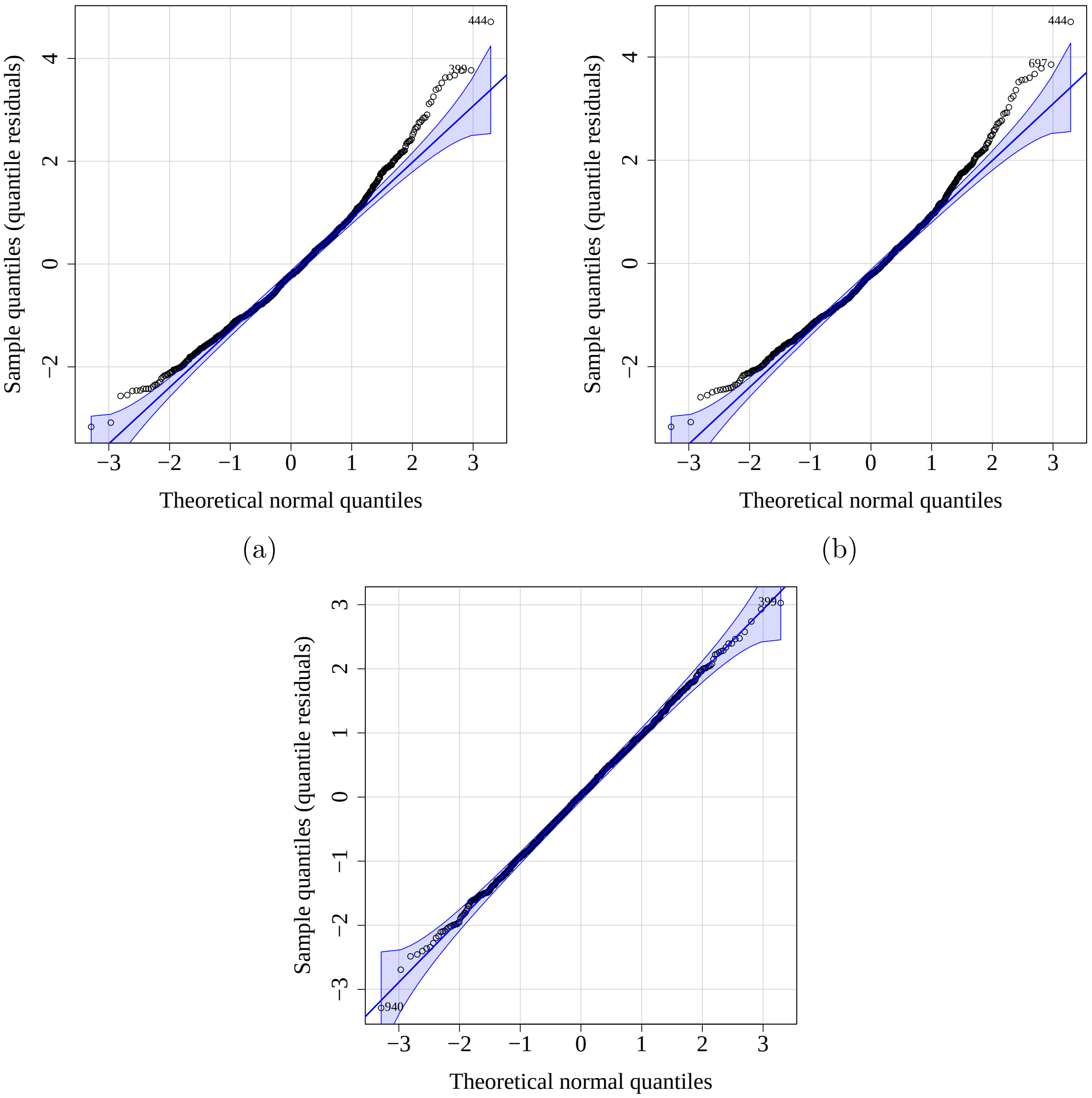


(a)

(b)

(c)

Figure 7-7: RQRs for scale parameter against index: (a) Model 2; (b) Model 1; (c) Model 0. The red line represents the smoothed trend.



(a)

(b)

(c)

Figure 7-8: Normal Q-Q plot of RQRs for scale parameter: (a) Model 2; (b) Model 1; (c) Model 0. The red line represents the smoothed trend.

7.3. Computational issues

Both Cox-Snell residuals and martingale residuals, as well as the censored deviance residuals, depend on the computation of the cumulative hazard function (CHF) of the fitted model. Therefore, we implemented a function to compute CHF given a probability density function, as well as a method to perform the same calculation for `maxlogL` objects. We will briefly examine them below.

7.3.1. `cum_hazard_fun`

```
cum_hazard_fun(
  distr,
  support = NULL,
  method = c("log_sf", "integration"),
  routine = NULL
)
```

This function takes the name of a distribution and computes its CHF using two methods

- `method = "log_sf"` computes the CHF by exploiting the mathematical relation it holds with the survival function, i.e, it uses the following expression:

$$H(t) = -\log\left(S(t|\hat{\theta})\right),$$

  where $S(t|\hat{\theta})$ is the survival function using the estimated parameters and $H(\cdot)$ is the cumulative hazard function.

- `method = ïntegration"` just integrates the hazard function,

$$H(t) = \int_0^t h(s)ds.$$

The `support` input parameter is useful when `method = ïntegration"` is set up. For example, consider the Weibull distribution, then

$$h(t) = \frac{\alpha}{\sigma}\left(\frac{t}{\sigma}\right)^{\alpha-1}, \tag{7-24}$$

$$H(t) = \left(\frac{t}{\sigma}\right)^{\alpha}, \tag{7-25}$$

where σ, $\alpha > 0$ and $t > 0$. Let's compare the value of the closed expression against that obtained by `cum_hazard_fun()`

```r
Hweibull1 <- function(x, scale, shape){
  (x/scale)^shape
}

Hweibull2 <- cum_hazard_fun(
  distr = ’dweibull’,
  method = "log_sf"
)

Hweibull1(0.2, shape = 2, scale = 1)

## [1] 0.04

Hweibull2(0.2, shape = 2, scale = 1)

## [1] 0.04
```

7.3.2. cum_hazard.maxlogL

```r
cum_hazard.maxlogL(object, ...)
```

This is a wrapper function specifically designed to provide `cum_hazard_fun()` with the CDF from the R environment and optionally the support stored into `maxlogL` models to compute the CHF.

7.4. Illustrative example

Let’s examine the Head and neck cancer application example presented in the previous chapter in section 6.6.1.

Recall that we fitted a Weibull and an exponentiated Weibull. Likewise, we confirmed that the exponentiated Weibull model is more adequate according to the AIC. Let’s check the models for or computing and visualising the corresponding residuals.

Let’s compare their Cox-Snell residuals (figures **7-9** and **7-10**). The top left, top right and bottom right plots reveal Cox-Snell residuals much closer to the standard exponential distribution for the exponentiated Weibull model (whose residuals are represented in **7-10**), hence it is more adequate to represent the dataset.

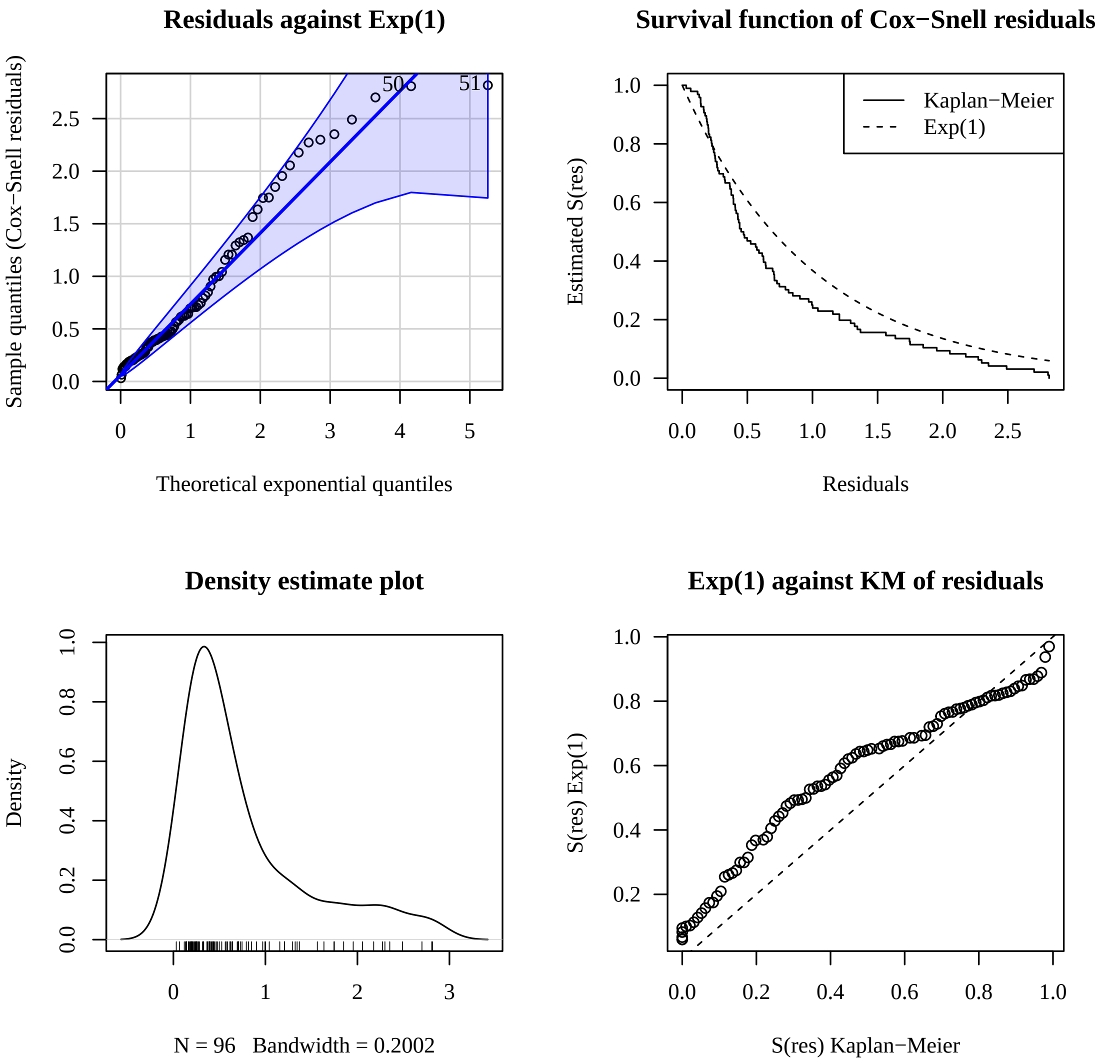


Figure 7-9: Diagnostic plots for the Weibull model of the head and neck cancer dataset using Cox-Snell residuals. Top left: standard exponential Q-Q plot of residuals, top right: Kaplan-Meier estimator of residuals and theoretical exponential survival function, bottom left: density plot of residuals, bottom right: comparison of the standard exponential survival function with the Kaplan-Meier estimator of residuals.

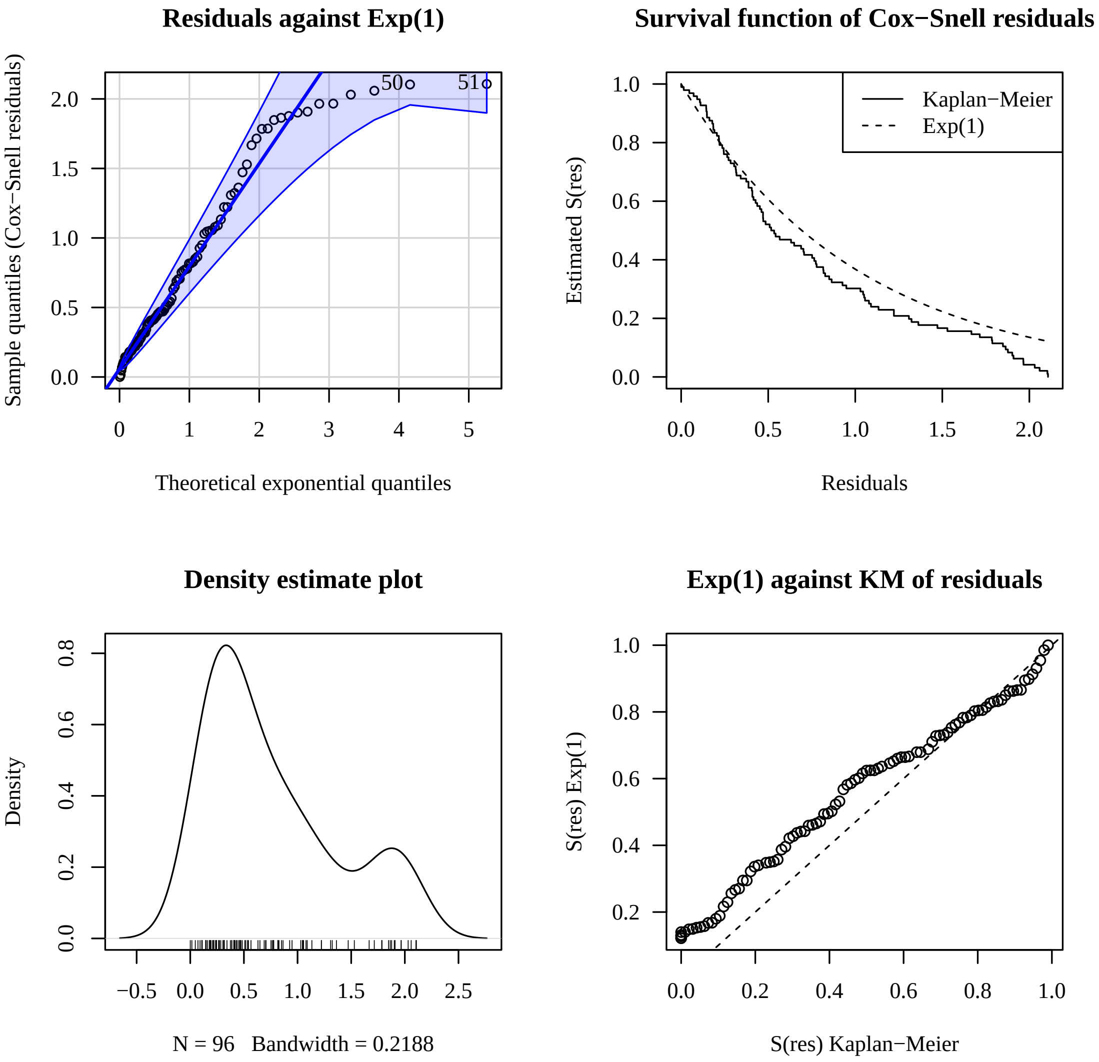
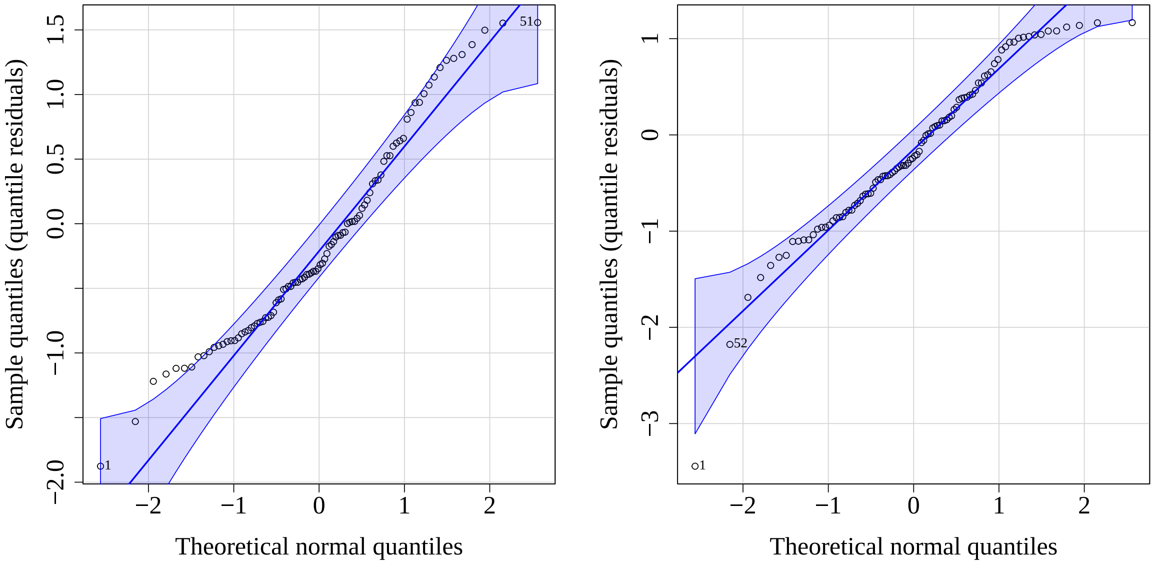


Figure 7-10: Diagnostic plots for the exponentiated Weibull model of the head and neck cancer dataset using Cox-Snell residuals. Top left: standard exponential Q-Q plot of residuals, top right: Kaplan-Meier estimator of residuals and theoretical exponential survival function, bottom left: density plot of residuals, bottom right: comparison of the standard exponential survival function with the Kaplan-Meier estimator of residuals.

On the other hand, let's examine the RQRs are shown in figure **7-11**. Consistently, figure **7-11**b exhibits a closer behaviour of residuals than figure **7-11**a, hence the first one is more adequate.



(a) Weibull model.

(b) Exponentiated Weibull model.

Figure 7-11: Standard normal Q-Q plot of residuals.

7.5. Conclusions

Residuals are an essential tool for evaluating the fit of parametric survival models. Cox-Snell residuals are used to assess the adequacy of a survival model, martingale residuals are useful to investigate the relationship between a response variable and covariates, and deviance residuals are used to evaluate the overall fit of a model. Randomized quantile residuals are a flexible approach that can be used to identify outlying observations and assess model fit. They are particularly useful when dealing with datasets with complex distributions.

The `residuals` method for `maxlogL` regression objects allows the computation of all the previously mentioned residuals for parametric models. Hence, with **EstimationTools** we provided a flexible and readable framework not only to fit but also diagnose regression models.

Part III

Implementation of a lifetime distribution

8. The odd Weibull distribution

> ” *The sciences do not try to explain, they hardly even try to interpret, they mainly make models. By a model is meant a mathematical construct which, with the addition of certain verbal interpretations, describes observed phenomena. The justification of such a mathematical construct is solely and precisely that it is expected to work.*
>
> — **John Von Neumann**
>
> (A true polymath)

The classical two-parameter Weibull distribution is a parsimonious and convenient way to model lifetime data because it describes skewed density shapes and fits data with monotonic hazard rates (Jiang et al., 2008), which are typical in reliability. However, there are some mechanical or electronic components with a bathtub-shaped hazard function because they have a rapidly decreasing initial hazard rate; other situations are described by unimodal failure rates. An example of the last case is observed in the course of a disease whose mortality reaches a maximum after some finite period and decreases progressively; the former case is in agreement with the behaviour of electronic equipment (Prataviera et al., 2018). Various generalisations of the Weibull distribution have been proposed to model this kind of dataset. Such is the case for the exponentiated Weibull family (Mudholkar and Srivastava, 1993), the modified Weibull distribution (Xie et al., 2002), the flexible Weibull extension distribution (Bebbington et al., 2007) and the generalized modified Weibull family (Carrasco et al., 2008). Despite these families being flexible, they do not exhibit a comfortable bathtub-shaped failure rate, which means that the second portion of the hazard curve is flat and long (Jiang et al., 2008). This property is helpful because it properly models the useful period (Bebbington et al., 2006). With the aim of solving this matter, Cooray (2006) presented a generalisation of the Weibull distribution derived from the odds of the Weibull and inverse Weibull (Fréchet) families.

Once a probability distribution family fits a response variable, it is useful to obtain expressions that explain its parameters with covariates through regression models. In this sense, many authors have developed regression models that produce estimates for one particular family, such as the Quadratic Regression Model with Non-constant Spread Parameter Meeker and Escobar (1998), the Log-generalized modified Weibull regression model (Ortega et al., 2011) and the exponentiated exponential mixture model (Mazucheli et al., 2013). Here, we implement the odd Weibull distribution in our `R` package **RelDists** (Hernandez et al., 2023) and perform parameter estimation using `maxlogL` and `maxlogLreg` from **EstimationTools**.

8.1. The distribution

Cooray Cooray (2006) proposed the odd Weibull (OW) distribution for survival analysis. This arises if a log-logistic distribution is assumed for the odds of death of variables with Weibull or Fréchet distribution. Further details are provided in Cooray (2006). The cumulative density function (CDF) of the odd Weibull (OW) family has the following expression:

$$F(t) = 1 - \left[1 + \left(e^{(\mu t)^{\sigma}} - 1\right)^{\nu}\right]^{-1} \tag{8-1}$$

where μ is the rate parameter, σ and ν are the shape parameters, with $t > 0$, $\mu > 0$ and $\sigma\nu > 0$. The probability density function (PDF) is:

$$\begin{aligned} f(t) = \left(\frac{\sigma\nu}{t}\right)(\mu t)^{\sigma} e^{(\mu t)^{\sigma}} \left(e^{(\mu t)^{\sigma}} - 1\right)^{\nu-1} \\ \left[1 + \left(e^{(\mu t)^{\sigma}} - 1\right)^{\nu}\right]^{-2}. \end{aligned} \tag{8-2}$$

In other words, we write $T \sim \mathrm{OW}(\mu,\, \sigma,\, \nu)$ if the distribution of T is (8-2). This distribution has the following particular cases (Cooray, 2006; Jiang et al., 2008):

- For $\nu = 1$, expressions (8-1) and (8-2) are those of the Weibull distribution.
- For $\nu = -1$, expressions (8-1) and (8-2) are those of the inverse Weibull (Fréchet) distribution.

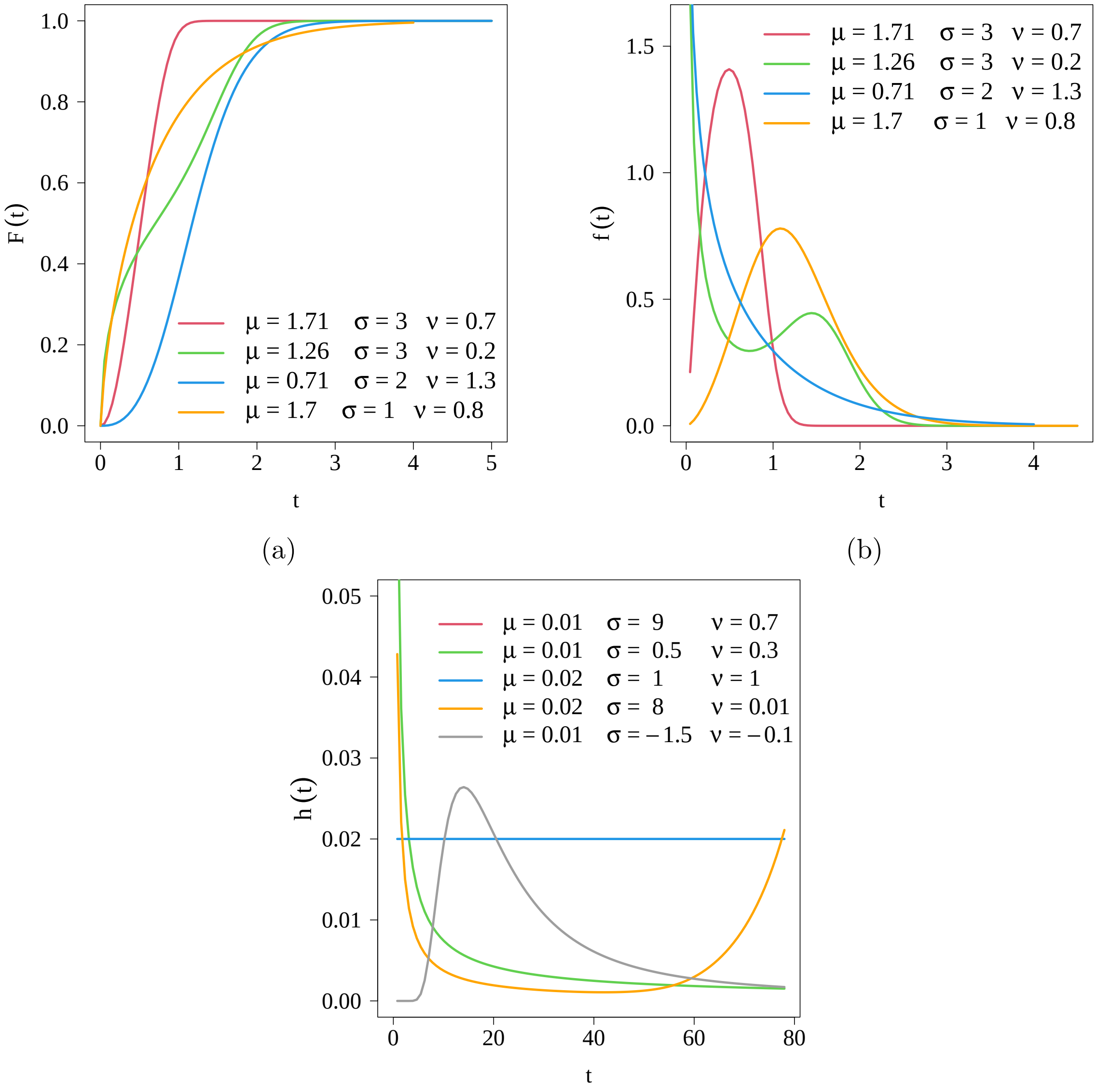
The hazard function (HF) represents the failure rate and for the OW distribution has the following expression:

$$\begin{aligned} h(t) = \left(\frac{\sigma\nu}{t}\right)(\mu t)^{\sigma} e^{(\mu t)^{\sigma}} \left(e^{(\mu t)^{\sigma}} - 1\right)^{\nu-1} \\ \left[1 + \left(e^{(\mu t)^{\sigma}} - 1\right)^{\nu}\right]^{-1}. \end{aligned} \tag{8-3}$$

According to Almalki and Nadarajah (2014) and Cooray (2006), the OW hazard function takes the following shapes:

- $h(t)$ increases if $\sigma > 1$ and $\sigma\nu > 1$.
- $h(t)$ decreases if $\sigma < 1$ and $\sigma\nu < 1$.
- $h(t)$ is unimodal shaped if $\sigma < 0$ and $\nu < 0$ or $\sigma < 1$ and $\sigma\nu \geq 1$.
- $h(t)$ is bathtub shaped if $\sigma > 1$ and $\sigma\nu \geq 1$.

In figures **8-1**a and **8-1**b are shown some different possible curves obtained for CDF and PDF. Additionally, figure **8-1**c shows four possible shapes for hazard functions: unimodal,



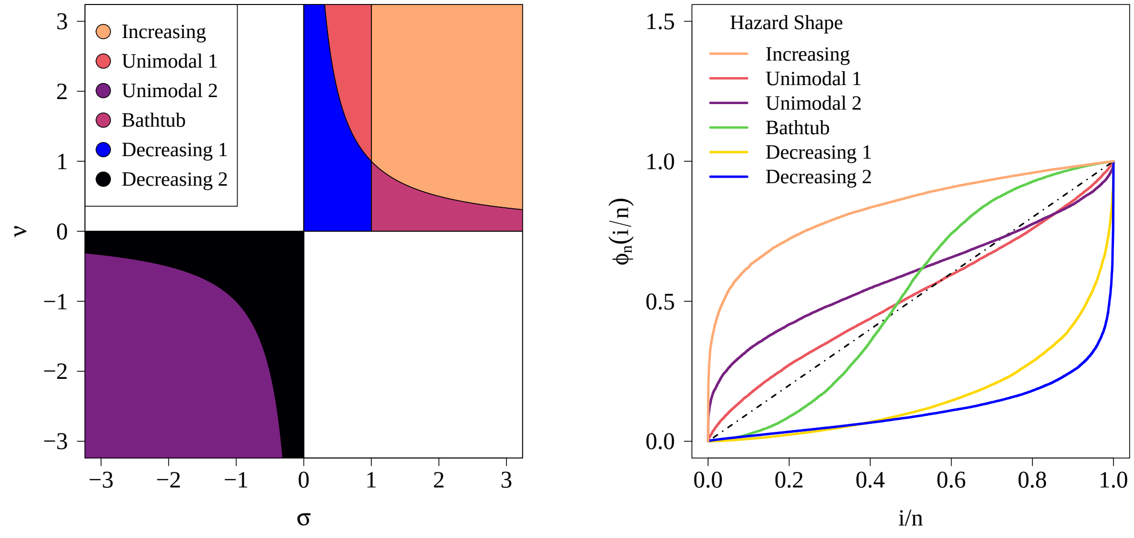
(a)

(b)

(c)

Figure 8-1: (a) Probability density function, (b) cumulative density function, and (c) hazard function for the Odd Weibull distribution.

decreasing, increasing, and bathtub. Figure **8-2**a illustrates the subspaces corresponding to different hazard shapes. It is important to note that parameter μ does not influence the hazard function.



(a) (b)

Figure 8-2: Classification of hazard function shape according to TTT plot shape: : (a) Hazard function shape according to the OW parametric space, (b) TTT plots of simulated data for a model with covariates for μ using `set.seed(345)` in R (sample size of 2000 in each case).

The quantile function is as follows:

$$Q(u) = F^{-1}(t) = \left(\frac{1}{\mu}\right) \left\{ \log \left[1 + \left(\frac{u}{1-u} \right)^{1/\nu} \right] \right\}^{1/\sigma} \tag{8-4}$$

where $0 < u < 1$ is a probability representing $F(t)$. The functions (8-1), (8-2), (8-3) and (8-4) were implemented in the **RelDists** package and can be accessed by invoking the functions `dOW`, `pOW`, `hOW` and `qOW` respectively.

According to Jiang et al. (2008) and Cooray (2015), the k^{th} moment of an odd Weibull distributed random variable is given by the following expression:

$$E(T^k) = \int_0^{\infty} t^k f(t) dt = \frac{k}{\sigma} \int_0^{\infty} \frac{x^{(k/\sigma)-1}}{1 + (e^x - 1)^{\nu}} dx. \tag{8-5}$$

This integral does not have a closed form and must be evaluated numerically. Moreover, Cooray (2015) proved that the k^{th} moment is finite if $\sigma > 0$ and $\nu > 0$, or if $\sigma < 0$ and $\nu < 0$, when $\nu\sigma > k$.

8.2. Estimation

Suppose that $T_1, T_2, ..., T_n$ is a random sample of size n for which the PDF is an OW distribution and let $\boldsymbol{\theta} = [\mu, \sigma, \nu]^\top$ be the vector of unknown parameters of equation (8-2). The log-likelihood can be shown to be Cooray (2015):

$$\begin{aligned} l(\boldsymbol{\theta}) = {} & n \log\left(\sigma\nu\mu^{-\sigma}\right) + (\sigma - 1)\sum_{i=1}^{n} \log t_i + \sum_{i=1}^{n} (\mu t_i)^\sigma \\ & + (\nu - 1) \sum_{i=1}^{n} \log\left(e^{(\mu t_i)^\sigma} - 1\right) \\ & - 2\sum_{i=1}^{n} \log\left[1 + \left(e^{(\mu t_j)^\sigma} - 1\right)^\nu\right], \end{aligned} \tag{8-6}$$

The maximum likelihood (ML) estimator of $\boldsymbol{\theta}$ is $\hat{\boldsymbol{\theta}}$. It can be obtained by solving the following nonlinear system of equations:

$$\left[\frac{\partial l(\boldsymbol{\theta})}{\partial \mu}, \frac{\partial l(\boldsymbol{\theta})}{\partial \sigma}, \frac{\partial l(\boldsymbol{\theta})}{\partial \nu}\right]^\top = \mathbf{0}, \tag{8-7}$$

where the derivatives can be expressed as follows:

$$\begin{aligned} \frac{\partial l(\boldsymbol{\theta})}{\partial \mu} = {} & \frac{n}{\sigma} + \sum_{i=1}^{n} \log\left(e^{(\mu t_i)^\sigma} - 1\right) \\ & - 2\sum_{i=1}^{n} \frac{\left(e^{(\mu t_i)^\sigma} - 1\right)^\nu \log\left(e^{(\mu t_i)^\sigma} - 1\right)}{1 + \left(e^{(\mu t_i)^\sigma} - 1\right)^\nu}, \end{aligned} \tag{8-8}$$

$$\begin{aligned} \frac{\partial l(\boldsymbol{\theta})}{\partial \sigma} = & - n\mu\sigma - \mu\sigma \sum_{i=1}^{n} (\mu t_i)^\sigma \\ & - \mu\sigma(\nu - 1)\sum_{i=1}^{n} \frac{(\mu t_i)^\sigma e^{(\mu t_i)^\sigma}}{e^{(\mu t_i)^\sigma} - 1} \\ & + 2\mu\nu\sigma \sum_{i=1}^{n} \frac{(\mu t_i)^\sigma e^{(\mu t_i)^\sigma}\left(e^{(\mu t_i)^\sigma} - 1\right)^{\nu - 1}}{1 + \left(e^{(\mu t_i)^\sigma} - 1\right)^\nu}, \end{aligned} \tag{8-9}$$

$$
\begin{aligned}
\frac{\partial l(\boldsymbol{\theta})}{\partial \nu} =& \frac{n}{\sigma} + \frac{1}{\sigma}\sum_{i=1}^{n}\log(\mu t_i)^{\sigma} + \frac{1}{\sigma}\sum_{i=1}^{n}(\mu t_i)^{\sigma}\log(\mu t_i)^{\sigma} \\
&+ \frac{1}{\sigma}(\nu-1)\sum_{i=1}^{n}\frac{(\mu t_i)^{\sigma}e^{(\mu t_i)^{\sigma}}\log(\mu t_i)^{\sigma}}{e^{(\mu t_i)^{\sigma}}-1} \\
&- 2\frac{\nu}{\sigma}\sum_{i=1}^{n}\frac{(\mu t_i)^{\sigma}e^{t_j}\left(e^{(\mu t_i)^{\sigma}}-1\right)^{\nu-1}\log(\mu t_i)^{\sigma}}{1+\left(e^{(\mu t_i)^{\sigma}}-1\right)^{\nu}},
\end{aligned}
\tag{8-10}
$$

On the other hand, $\hat{\boldsymbol{\theta}}$ can be computed using the `maxlogL` function to estimate distribution parameters or `maxlogLreg` to implement distributional regression models and the **RelDists** (Hernandez et al., 2023) package to obtain the OW distribution in R (R Core Team, 2023).

8.3. Computational implementation

As stated before, the PDF, CDF and hazard function were implemented as the R functions `dOW`, `pOW` and `hOW` respectively. Once these functions are available, they can be used within the **EstimationTools** framework to perform maximum likelihood estimation. Additionally, we implemented the quantile function in `qOW` function and the random number generator was implemented in `rOW` function. Visit the distribution script in **RelDists** package for further computational details.

`maxlogLreg` has been conceived originally to solve maximum likelihood estimation with box-constrained supports, whereas the OW distribution has some convenient subregions with hyperbolic boundaries. We tackled this problem by adding a penalisation term to the log-likelihood computed under the hood, as follows:

$$
l(\boldsymbol{\theta}(\boldsymbol{\beta})|\boldsymbol{y}) = \left(\sum_{i=1}^{n}\log R_{i1}\cdot f(y_i|\boldsymbol{\theta}(\boldsymbol{\beta})) + R_{i2}\cdot S(y_i|\boldsymbol{\theta}(\boldsymbol{\beta})) + R_{i3}\cdot F(y_i|\boldsymbol{\theta}(\boldsymbol{\beta}))\right) - \lambda\mathbf{1}_A(\boldsymbol{\theta}),
\tag{8-11}
$$

where $\boldsymbol{\beta}$ is a vector of regression parameters, $\boldsymbol{\theta}(\boldsymbol{\beta})$ are the distribution parameters (which can be expressed in terms of regression parameters and covariates) and R_{il} is an element of the status matrix defined as follows:

$$
R_{il} = \begin{cases} 1, & \text{if sample unit } i \text{ has status } l, \\ 0, & \text{in other case,} \end{cases}
\tag{8-12}
$$

with $i = 1, 2, \ldots, n$ and $l = 1, 2, 3$ ($l = 1$: observation status, $l = 2$: right censorship status, $l = 3$: left censorship status). $\mathbf{1}_A(x)$ is the indicator function of a subset A of the form $A =$

$\{\boldsymbol{\theta}|h(\boldsymbol{\theta}) > c\}$, or $A = \{\boldsymbol{\theta}|h(\boldsymbol{\theta}) = c\}$ or $A = \{\boldsymbol{\theta}|h(\boldsymbol{\theta}) < c\}$; with c a constant. The equation (8-11) includes a penalty term with a penalisation parameter $\lambda \to \infty$ (which is set to 1×10^{10} in practise). The penalisation parameter significantly lowers the log-likelihood function when the optimal values of $\boldsymbol{\theta}(\boldsymbol{\beta})$ fail to satisfy the constraints in A. This mechanism ensures that the distribution parameters remain within the appropriate region of the parameter space.

8.3.1. Estimation without covariates

Estimation of the OW distribution is sensitive to initial values. From our own experience, a good initial guess for the scale parameter μ is the inverse mean. For σ and ν there are more possibilities. It can be distinguished from five regions related to the shape of the hazard function Cooray (2006). These regions are identified in figure **8-2**a.

Therefore, the parametric space can be bounded by knowing the hazard shape. To estimate the hazard shape, we calculate the scaled empirical time on test transform (TTT). Moreover, we use the fact that the hazard function shape may be decreasing, increasing, bathtub, and unimodal if the TTT transform is convex, concave, convex then concave, and concave then convex, respectively. Some examples with simulated data are presented in figure **8-2**b.

Thus, for non-censored data the scaled empirical TTT can be computed based on the empirical cumulative distribution function (check Aarset's paper (Aarset, 1987)). The final expression is given by

$$\phi_n\left(\frac{i}{n}\right) = \frac{\left(\sum\limits_{j=1}^{i} T_{(j)}\right) + (n-i)T_{(i)}}{\sum\limits_{i=j}^{n} T_j}, \tag{8-13}$$

where $T_{(r)}$ is the r^{th} order statistic, with $T_{(0)} = 0$, $r = 1, 2, \ldots, n$, and n being the sample size. For censored data, a scaled empirical TTT can be computed using the expression of Kaplan-Meier survival estimator, based on expressions by Westberg and Klefsjö (1994)

$$\phi_n\left(\frac{i}{n}\right) = \frac{\sum\limits_{j=1}^{i}\left[\prod\limits_{k=1}^{j}\left(1 - \frac{d_k}{n_k}\right)\right]\left(T_{(j)} - T_{(j-1)}\right)}{\sum\limits_{j=1}^{n}\left[\prod\limits_{k=1}^{j}\left(1 - \frac{d_k}{n_k}\right)(T_j - T_{j-1})\right]}, \tag{8-14}$$

where d_i are the number of events and is the n_i the number of units at risk. These equations were implemented in `TTTE_Analytical` function from **EstimationTools**.

We implemented `initValuesOW`, which uses `TTTE_Analytical` to compute the empirical TTT and estimates a LOESS curve from it. Then, it calculates the inflexion point to determine the empirical TTT shape and therefore the hazard shape from the sample. Through this calculation, `initValuesOW` suggests fixed values as initial guesses, but allows the establishment of a subspace for parameter search corresponding to the hazard shape in the `inequalities` input parameter of `maxlogLreg` function. We also implemented `param.start`, which takes an `initValuesOW` object and extracts the initial guesses to set `sigma.start` and `nu.start` arguments in `gamlss` function.

By default, `initValuesOW` function sets the bounds in the first quadrant if the data correspond a to a hazard unimodal shape. If the estimation routine fails (e.g. `gamlss` or `maxlogLreg`) and the user suspects that the hazard function is unimodal, we suggest manually changing the initial guesses and valid regions with values located on the third quadrant.

There are cases where several inflexion points may be detected because the LOESS curve is not smooth enough. Additionally, there exists some extent of a lack of identifiability near the boundary of the parametric regions that define the available hazard function shapes. In figures **8-3** can be seen that points on the left side of the plot correspond to TTT plots similar to those points on the right side of the plot. Additional examples illustrating location in parametric space and TTT plot shape can be found in Appendix E.

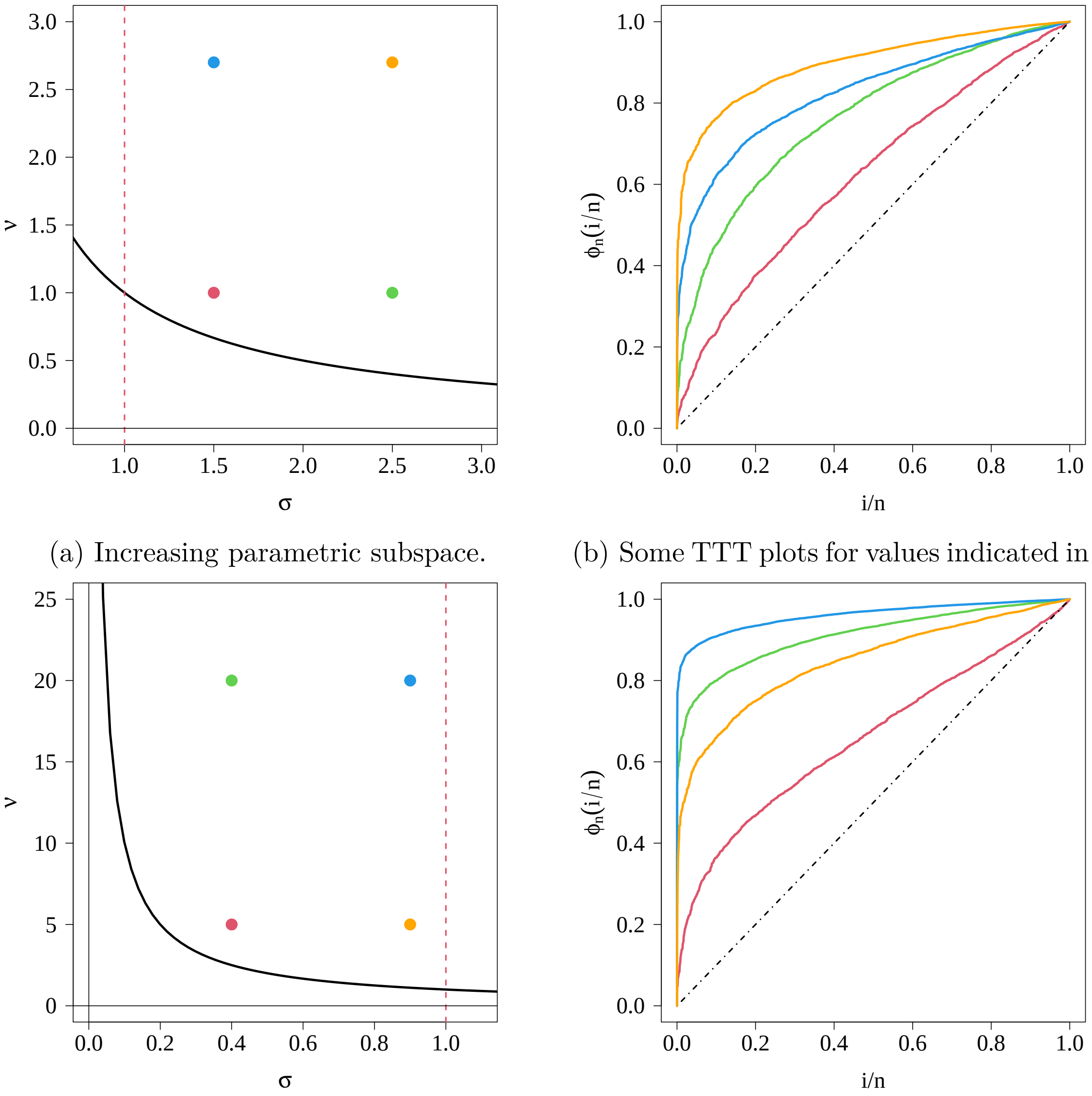
To overcome these issues, we encourage potential users to use `initValuesOW` and its `plot` method in their exploratory analysis. For example, one can tune the LOESS parameters, such as the `span` argument. Tuning initial guesses may be necessary.

8.3.2. Estimation using covariates

Another potential pitfall is that the TTT plot does not allow control of the effect of covariates that explain the parameters. Let's take a particular case as an example and consider two data sets simulated from the following model:

$$\begin{aligned} y &\sim OW(\mu,\, \sigma,\, \nu), \\ \log(\mu) &= \alpha_0 + \alpha_1 X, \\ \sigma &= 2, \\ \nu &= 2, \\ X &\sim U(0,1), \end{aligned} \tag{8-15}$$

with $\boldsymbol{\Theta} = (\alpha_0 = 0.05,\ \alpha_1 = -8)^\top$, $i = 1, 2, \ldots, n$; we tested $n = 50$ and $n = 10000$. The selected parameter values for σ and ν correspond to an increasing hazard case (see Figure **8-2**a), however TTT plots in figure **8-4** suggests a decreasing pattern, and the models correspond to a decreasing hazard rate. This happens because μ has a decreasing behaviour



(a) Increasing parametric subspace.

(b) Some TTT plots for values indicated in (a).

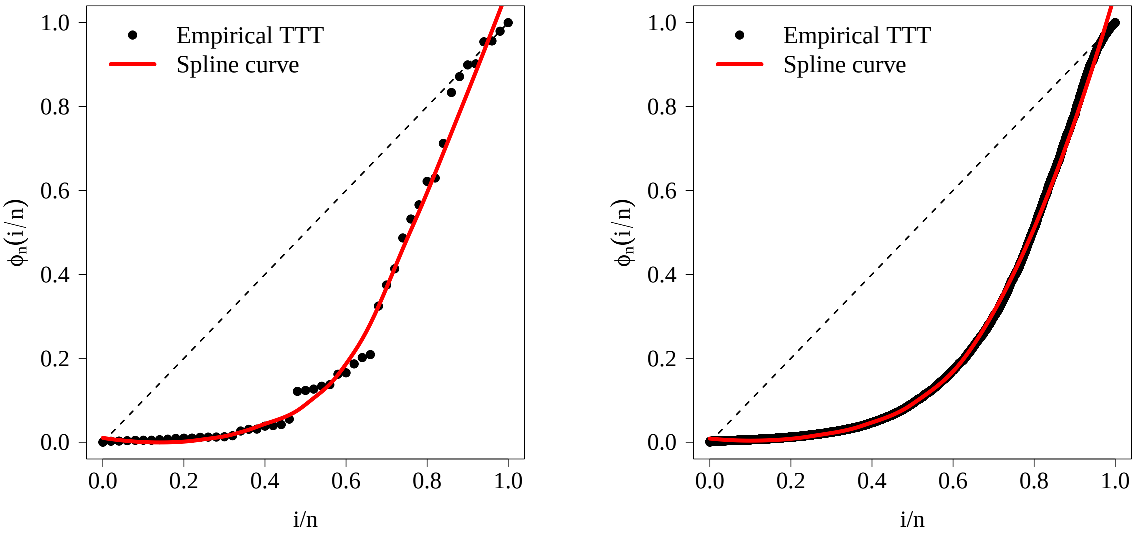
(c) Unimodal (positive) parametric subspace.

(d) Some TTT plots for values indicated in (c).

Figure 8-3: Parametric sub-spaces and TTT plots corresponding to data with increasing and unimodal (quadrant I) hazard function shapes. TTT plots are computed with simulated OW variables (sample size of 2000).

and has a great influence on the hazard function in this case (recall equation 8-3). Actually, the TTT plot does not allow visualisation of the effect of covariates in several cases. This is a typical case of Simpson's paradox, Carlson (2020); Wang et al. (2018) for further details.

Despite these issues, implementation of `expm1` and `log1p` is enough to achieve successful estimation using covariates and linear predictors for the distribution parameters.



(a) $n = 50$.

(b) $n = 10000$.

Figure 8-4: TTT plots of simulated data for a model with covariates for μ using `set.seed(345)` in R.

8.4. Application examples

In this section, we supply two applications to real datasets and one simulated dataset to illustrate the importance of OW distribution. Estimation is performed as stated in sections 8.2 and 8.3. In the first two examples, the original ML estimators are compared with those computed with `maxlogLreg`.

For estimation, it is necessary to load **RelDists** package in order to set the `dOW` and `pOW` functions and **EstimationTools** for fitting models. The packages can be downloaded by typing the following instructions in R:

```
install.packages("RelDists")
require("RelDists")
```

8.4.1. Time to failure on electronic equipment

Cooray (2015) used the following data provided by Wang (2000) in order to fit an OW distribution through ML estimation:

5, 11, 21, 31, 46, 75, 98, 122, 145, 165, 195, 224, 245, 293, 321, 330, 350, 420.

The data above are the time to failure of an electronic device in hours. They can be loaded as follows.

```r
# Data set input
data("equipment")
```

Results

The estimation of OW distribution parameters can be performed using an only-intercept model, as follows:

$$
\begin{aligned}
&y \sim OW(\mu,\, \sigma,\, \nu)\\
&\mu = \alpha_0\\
&\sigma = \beta_0\\
&\nu = \gamma_0.
\end{aligned}
\tag{8-16}
$$

Using our `initValuesOW`, we detected a bathtub hazard shape, which corresponds to the convex-then-concave shape of the TTT plot (see figure **8-5**).

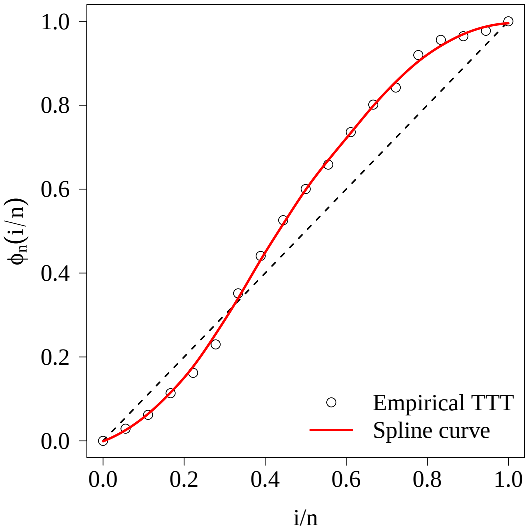


Figure 8-5: TTT plot of electronic equipment data.

We obtained the estimates shown in the third column of table **8-1**.

Code implementation

The R code to reproduce the results from section 8.4.1 is displayed in this section. The `initValuesOW` function produces an object that stores initial guesses for σ, ν and the valid region for both parameters. It can be computed and as follows:

Parameter	ML Cooray (2015)	ML `maxlogLreg`
μ	0.00535	0.0054
σ	3.22388	3.2213
ν	0.28424	0.2846

Table 8-1: Estimates for time to failure on electronic equipment data. The ML estimates in the second column are from Cooray Cooray (2015), and the third column ones are our estimates.

```r
# Data and TTT plot ------------------------------
my_initial_guess <- initValuesOW(formula = equipment ~ 1)
summary(my_initial_guess)
```

After that, TTT can be plotted:

```r
par(mfrow=c(1,1), family="serif", cex.lab=1.8, cex.axis=1.8, mgp=c(3.6,1.1,0),
    mai=c(1,1.2,0.1,0.1))
plot(
  my_initial_guess,
  curve_options = list(lwd=3, col="red"),
  cex = 1.6, las = 1
)

legend.HazardShape(
  x = "bottomright", xpd = FALSE, bty=’n’, col = c(1, "red"),
  lwd = c(1, 3), pch = c(1, NA), pt.cex = 1.6, cex = 1.8
)
```

Equation (8-16) can be written in the `maxlogLreg` style. First, specify the assumed distribution and its support:

```r
distribution <- equipment ~ dOW
support <- list(interval = c(0, Inf), type = ’continuous’)
```

Then, the linear predictors,

```r
formulas <- list(mu.fo = ~ 1, sigma.fo = ~ 1, nu.fo = ~ 1)
```

the initial values,

```r
start_values <- list(
  mu = list(Intercept = 1/mean(equipment)),
  sigma = list(Intercept = my_initial_guess$sigma.start),
  nu = list(Intercept = my_initial_guess$nu.start)
)
```

and the box-constrains for each parameter

```
lower <- list(
  mu = list(Intercept = 1e-4),
  sigma = list(Intercept = 1),
  nu = list(Intercept = 1e-4)
)
upper <- list(
  mu = list(Intercept = 0.1),
  sigma = list(Intercept = 10),
  nu = list(Intercept = 1)
)
```

Note the inclusion of the non-linear constrain:

```
ineq <- my_initial_guess$nu.valid
ineq
```

Finally, run the estimation process

```
equipment_model <- maxlogLreg(
    formulas = formulas,
    y_dist = distribution,
    support = support,
    start = start_values,
    lower = lower,
    inequalities = ineq,
    control = list(iter.max = 200)
)

summary(equipment_model)
```

8.4.2. Mortality in mice exposed to radiation

A dataset with 208 data points provided by **?** with the ages at death in weeks for male mice exposed to 240r of gamma radiation.

```r
data("mice")
head(mice, n = 6)

## [1] 40 48 50 54 56 59
```

Results

We identified the hazard shape with `initValuesOW`, which increased in accordance with Cooray (2006) (see figure **8-6**).

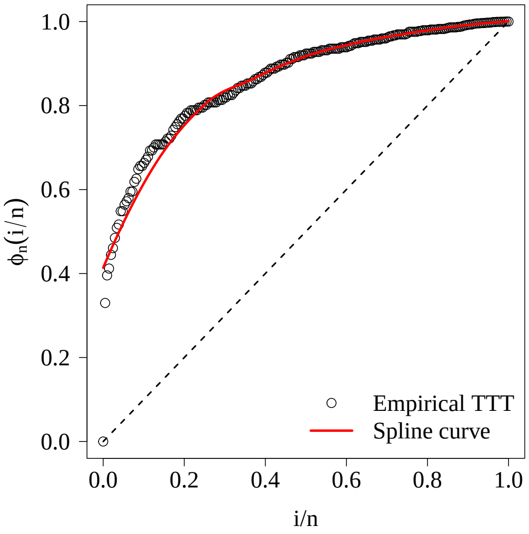


Figure 8-6: TTT plot of mortality of mice data.

Similarly, we implemented a workflow with `maxlogLreg` to get the estimates in the third column of table **8-2**.

Parameter	ML Cooray (2006)	ML `maxlogLreg`
μ	0.00761	0.0076
σ	6.22780	6.2117
ν	0.74950	0.7514

Table 8-2: Estimates for mortality of mice mortality data.

Code implementation

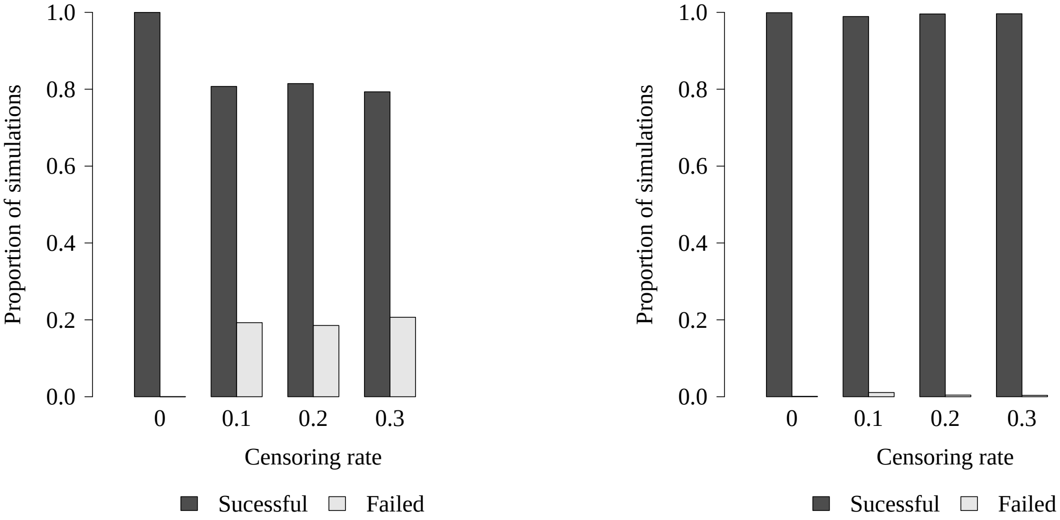
Here, we display the code to reproduce the results from section 8.4.2.

```
# Data and TTT plot ------------------------------
data("mice")
init_vals <- initValuesOW(formula = mice ~ 1)

plot(
  init_vals,
  curve_options = list(lwd=3, col="red"),
  cex = 1.6, las = 1
)

legend.HazardShape(
  x = "bottomright", xpd = FALSE, bty=’n’, col = c(1, "red"),
  lwd = c(1, 3), pch = c(1, NA), pt.cex = 1.6, cex = 1.8
)

# Fit the model ------------------------------
distribution <- mice ~ dOW
support <- list(interval = c(0, Inf), type = ’continuous’)

formulas <- list(
  mu.fo = ~ 1,
  sigma.fo = ~ 1,
  nu.fo = ~ 1
)

start_values <- list(
  mu = list(Intercept = 1/mean(mice)),
  sigma = list(Intercept = init_vals$sigma.start),
  nu = list(Intercept = init_vals$nu.start)
)
lower <- list(
  mu = list(Intercept = 1e-4),
  sigma = list(Intercept = 1),
  nu = list(Intercept = 1e-4)
)
upper <- list(
  mu = list(Intercept = 0.1),
  sigma = list(Intercept = 10),
  nu = list(Intercept = 1)
)
ineq <- init_vals$nu.valid

mice_model <- maxlogLreg(
    formulas = formulas,
    y_dist = distribution,
    support = support,
    start = start_values,
    lower = lower,
    inequalities = ineq
)

summary(mice_model)
```

8.5. Simulation study

Cooray (2006) fitted an OW distribution through ML estimation using the time-to-failure data set Aarset (1987). Estimation leads to $\mu_e \approx 0.0187$, $\sigma_e \approx 7$ and $\nu_e \approx 0.09$ for our parametrization. We used these values and conducted a simulation study to obtain a better understanding of the effect of the censoring rate in the OW fitting process. Therefore, we generated 1000 times OW(μ_e, σ_e, ν_e) variates with sample sizes $n = 10, 30, 40, ..., 300$ and censoring percentage $c_p = 0\,\%, 10\,\%, ..., 40\,\%$, i.e, we simulated 50000 random samples OW distributed.

The simulation results are reported in figure **8-8**. Through the mean square error (MSE), we verified the correct performance of the OW estimation process and `initValuesOW` since the error of estimates decreases as the censoring rate decreases and tends to zero as the sample size increases, as expected by the consistency of maximum likelihood estimates. It seems that the MSE stabilises quite quickly, and for a sample size of approximately between 50 and 100, it becomes quite stable across all censoring percentages. Furthermore, our implementation demonstrates greater computational efficiency. In figure **8-7** we can observe that our implementation increases the rate of successful estimations. The behaviour of the mean value, median value, and bias of the estimates can be found in Appendix E.



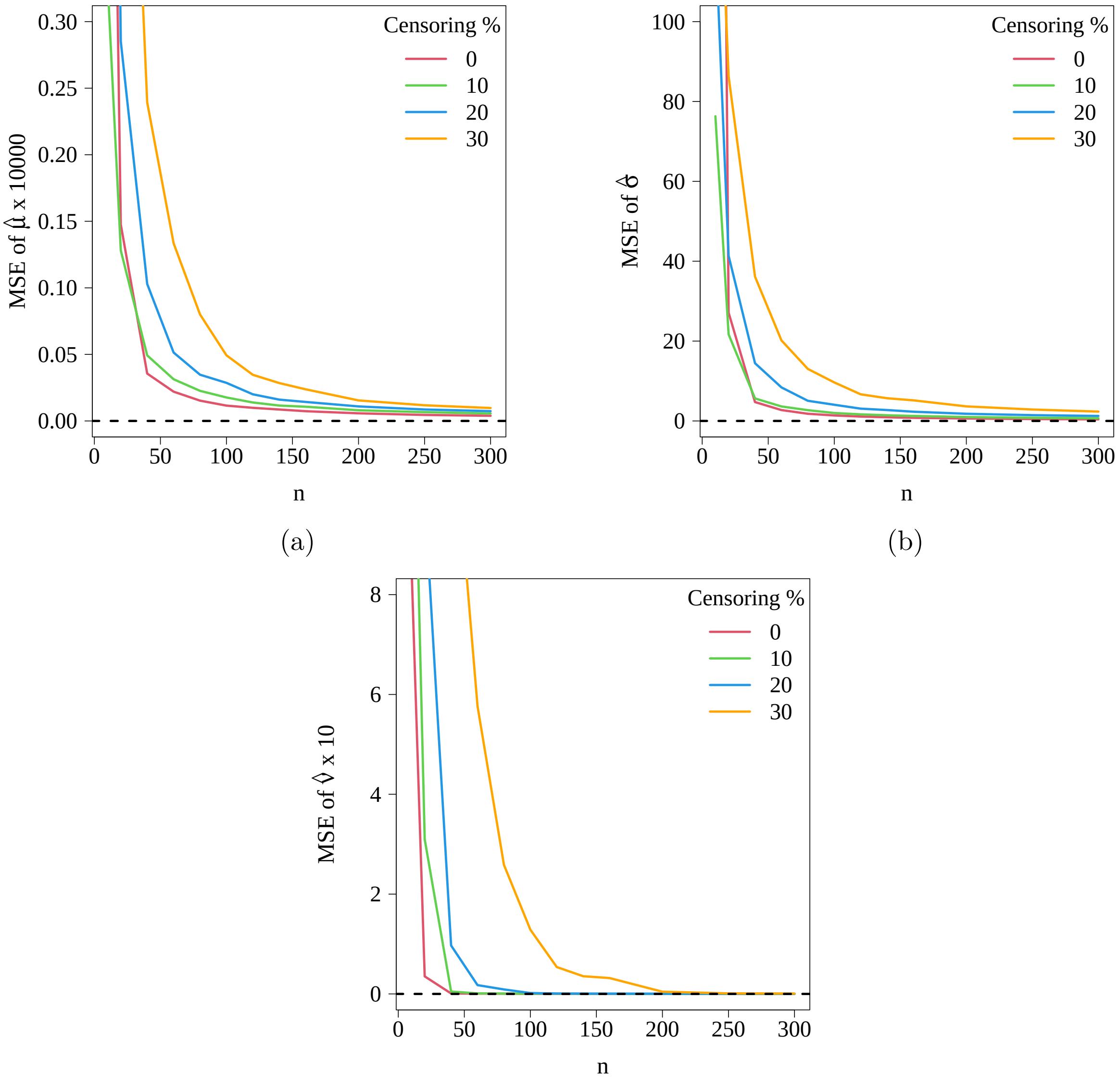
(a)

(b)

Figure 8-7: Proportion of successful and failed simulations for (a) the standard estimation, (b) estimation enhanced with `initValuesOW` in failure on electronic equipment data.

8.6. Conclusions

In this chapter, we implemented the odd Weibull in a set-up in which the parameters can vary across observations through linear predictors. Existing implementations of optimisation algorithms can be used to compute the maximum likelihood estimators of the odd Weibull distribution, and `maxlogLreg` automates the log-likelihood computation. We implemented an algorithm to obtain initial guesses for parameters based on the parametric space described by Cooray (2006). Moreover, the implementation allows the fit of regression models. We fitted real data in two application examples and conducted a simulation study to demonstrate the usefulness of the implementation. As expected, the results in the application examples are close to those previously obtained by Cooray (2006, 2015), and the estimates are better as the sample size increases in the simulation studies.



(a)

(b)

(c)

Figure 8-8: Mean squared error (MSE) for estimated parameters, (a) for $\hat{\mu}$, (b) for $\hat{\sigma}$ and (c) for $\hat{\nu}$. The dashed line indicates MSE = 0.

Part IV

Concluding remarks

9. Future directions and conclusions

> *The future is already here – it’s just not evenly distributed*
>
> — **William Gibson**
>
> (Fiction writer and essayist)

This research work aimed to create and release **EstimationTools**, a package intended to provide the statistics and data science community with general-purpose tools useful for estimating the parameters of probability distributions. Although our proposal is useful for the entire scientific community, it was endowed with features to estimate parametric models in survival analysis.

In the following sections, we will reflect on the limitations and opportunities that the findings of this study bring to the academic community and the R community.

9.1. Advantages and limitations

EstimationTools, and specifically `maxlogLreg` function offers a grammar for flexible distribution regression model declaration that resembles the mathematical syntax used by statisticians. To illustrate the concept, we consider the following heteroskedastic model:

```
dist <- y ˜ dnorm
support <- list(
  interval = c(-Inf, Inf),
  type = ’continuous’
)
link <- list(
  over = "sd", fun = "log_link"
)
formulas <- list(
  mean.fo = ˜ x, sd.fo = ˜ x
)
```

$$
\begin{aligned}
X_i &\sim U(-5,\, 6),\\
y_i &\sim N(\mu_i,\, \sigma_i),\\
\mu_i &= \alpha_0 + \alpha_1 X_i,\\
\log(\sigma_i) &= \beta_0 + \beta_1 X_i
\end{aligned}
\qquad (9\text{-}1)
$$

Our approach also allows users to control every aspect of log-likelihood optimisation through algorithms implemented in R. However, this syntax can be complex for novice users of R (recall section 8.4.1, where non-linear restrictions and initial values should have been set in addition to the distribution, its support, and formulas).

9.2. Potential research directions

Software development is a never ending endeavour. The previously mentioned limitations provide both us and computational statistics developers with opportunities to explore and suggest alternative solutions. These new approaches could, on one hand, be grounded in fundamental concepts such as Object-Oriented Programming and non-linear constrained optimisation On the other hand, they could be developed on the basis of other distributional regression approaches, such as Generalized Additive Models for Location, Scale, and Shape (GAMLSS) (Rigby and Stasinopoulos, 2005). Finally, we found that it is possible to propose detection functions based on survival functions for estimating population density.

9.2.1. Object-Oriented approach

Our proposal is probably very appealing to users who are not oriented towards software engineering, as all the necessary elements for model declaration are provided one by one before model fitting. However, from a software engineering perspective, it would be more convenient to take advantage of encapsulation. Therefore, elements such as `d`-functions (densities), `p`-functions (cumulative densities), `q`-functions (quantile), `support`, and `fixed`, which are provided to `maxlogLreg`, could be stored in an object.

Some packages have made progress in this direction. For example, **distributions3** provides PDF, CDF, quantile, and random generator generic methods for probability distributions using the `S3` object system (Hayes et al., 2022). Similarly, **distributional** implements distributions as vectorised `S3` objects and additionally offers methods to compute summary statistics such as mean, variance, kurtosis, and skewness (O'Hara-Wild et al., 2023); methods for statistical inference such as `hilo` for confidence intervals or mean/median distribution computation; and methods for computing the likelihood. On the other hand, **distrEx** (Ruckdeschel et al., 2006)and **distr** (Ruckdeschel and Kohl, 2014) serve similar interfaces with the `S4` system. Perhaps the most comprehensive implementation is **distrMod**, it uses **distr** and **distrEx** methods to extend the support of object-oriented implementation of probability models via several new S4 classes and methods in base R for distributions and parametric modelling (Kohl and Ruckdeschel, 2010).

The **gamlss** package has created an `S3` object of class `gamlss.family`, which is a halfway solution between an object-oriented approach and an informal declaration. All of these alternatives are interesting starting points, but they may move potential users away from simpler approaches. In this scenario, the estimation function (e.g., `maxlogLreg`) could be a method for these distributions, or the distributions could be aggregated into estimation objects/functions, similar to the **gamlss** workflow.

Despite the aforementioned advantages, these methods are only feasible for intermediate/advanced users or even R developers. To overcome this situation, a generic method to populate `gamlss.family` objects could be implemented. Our **gamlss.extra** package closes this gap.

9.2.2. Non-linear constrains

`maxlogLreg` works with `nlminb` routine by default and was designed to run specifically with box-constrained routines such as `optim`, `ga` or `DEoptim`. Additional optimisation routines can be tested using the `set_optimizer` adapter function. In this study, we addressed non-linear constraints using a penalisation term (recall equation (8-11)). However, modern maximum likelihood estimation problems have much to do with non-linear boundaries. In particular, we dealt with hyperbolic constraints on the estimation of OW parameters (recall figure **8-2**a). Regarding this fact, it is worth mentioning that the special routines designed for non-linear constrained optimisation have not been explored. One particularly interesting alternative could be **nloptr** (Johnson, 2022).

9.2.3. Contributions to gamlss project

gamlss package is a huge and high-impact project that implements the GAMLSS framework in R. A potential research topic could be the odd Weibull family amending the GAMLSS framework. Implementation of OW's pdf (`dOW`), cdf (`pOW`), hazard (`hOW`) and quantile function (`qOW`) in R is required. The `gamlss.family` structure needs to be adapted to add the OW distribution and put in a function with the family information needed for fitting the distribution, such as the header, the first and second derivatives used in estimation, and initial values. Additionally, the effect of passing non-linear constraints to the `RS()` algorithm should be investigated, as we did with `maxlogLreg` with the penalised log-likelihood (equation (8-11)).

We have already explored these issues in part. Consider the following regression model:

$$\begin{aligned} y_i &\sim OW(\mu_i,\, \sigma_i,\, \nu_i), \\ \log(\mu_i) &= \alpha_0, \\ \log(\sigma_i) &= \beta_0 + \beta_1 X_{1i}, \\ \log(\nu_i) &= \gamma_0 + \gamma_1 X_{1i}, \end{aligned} \tag{9-2}$$

where X_{1i} is binary (balanced) covariate and the vector parameter is $\boldsymbol{\Theta} = (\alpha_0 = 0.05, \beta_0 = -0.361, \beta_1 = 1.460, \gamma_0 = 0.694, \gamma_1 = -1.61)\top$. Specifically, when $X_{1i} = 0$, the hazard shape is unimodal, whereas the behaviour increases when $X_{1i} = 1$. This model does not successfully converge with standard optimisation routines implemented in R, however `gamlss()` function is able to find plausible estimators

```r
library(gamlss)
con.out <-gamlss.control(n.cyc = 500, trace = FALSE)
con.in <- glim.control(cyc = 600)

set.seed(0404)
mu <- 0.05; beta0 <- -0.361; beta1 <- 1.460
gamma0 <- 0.694; gamma1 <- -1.61
n <- 1000
X1 <- c(rep(0, n/2), rep(1, n/2))
sigma <- exp(beta0 + beta1*X1)
nu <- exp(gamma0 + gamma1*X1)
y <- rOW(n, mu=mu, sigma=sigma, nu=nu)

OW_model <- gamlss(y~1, sigma.fo=~X1, nu.fo=~X1,
                    control=con.out, i.control=con.in,
                    family=OW)

cat(paste0("mu hat:\n", coef(OW_model, what = "mu") |> exp()))
cat("sigma hat:\n")
coef(OW_model, what = "sigma")
cat("nu hat:\n")
coef(OW_model, what = "nu")
```

```
## mu hat:
## 0.049307351400222
## sigma hat:
## (Intercept)          X1
##   -0.347746    1.395705
## nu hat:
## (Intercept)          X1
##   0.7333091  -1.5700118
```

In our experience, **gamlss** has a robust and effective two-step algorithm for computing maximum likelihood estimators. An in-depth discussion is out of the scope of this document, visit the family script and the RelDists package vignette for further information.

Finally, we can leverage our previous experience to provide some additional features to **gamlss**. Some unique features from **EstimationToos** and `maxlogLreg` such as moments estimation, hazard function computation, and survival analysis residuals computation, could be adapted for `gamlss` objects.

9.2.4. Estimation of detection functions

Population density estimation in distance sampling requires fitting a probability density function denoted by $f(y|\theta)$, where y represents the perpendicular (or radial) distance from a detected animal (or object) to a transect line (or point), and θ represents the vector parameter indexing this family of probability density functions. The most popular approach to estimate $f(\cdot)$, is based on a semi–parametric methodology proposed by Buckland (1992); Buckland et al. (1993) using typical forms such as half-normal and hazard-rate functions (see figure **9-1**). The main idea is to find the maximum likelihood estimator for θ using a parametric functional form combined with a series expansion.

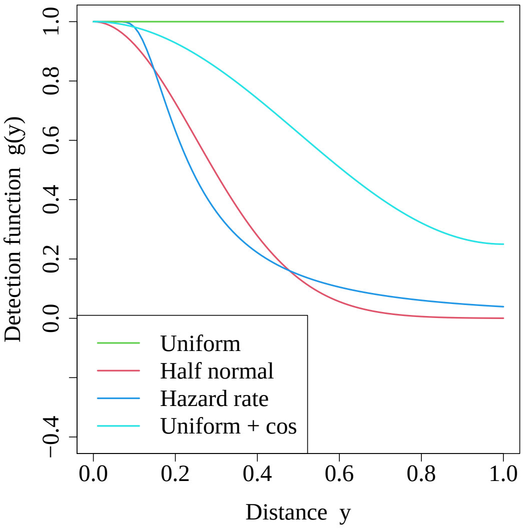


Figure 9-1: Usual detection functions used in distance framework.

Regarding the decreasing nature of detection functions, another interesting approach is to propose detection functions based on survival functions from novel distributions currently used in parametric survival analysis (Almalki and Nadarajah, 2014) and already implemented in **RelDists** (Hernandez et al., 2023). Some exploration has been made by Alanzi et al. (2023), however an official computational implementation would allow further investigation of the statistical properties and numerical behaviour of these functions.

9.3. Conclusion

We can say that our tool is far from perfect and is not the definitive solution to parameter estimation problems for distributions. However, during the development process, we found an opportunity to learn about special topics in maximum likelihood estimation, such as high-precision arithmetic, and gained a better understanding of existing frameworks, especially the generalised additive model for location, scale, and shape (GAMLSS).

Part V

Appendix

A. Appendix: Expected value as integral of the survival function

> *There is a harmony in the universe, if you can only find it.*
>
> — **Ralph Waldo Emerson**
>
> (Philosopher)

Let's prove that

$$\int_0^\infty S(t)dt = \mathbf{E}(T),$$

where T is a positive random variable which follows a lifetime distribution.

Proof. This result has been proved regarding the **Corollary 1** of James (2010). Consider the definition of the expected value using the Riemann-Stieltjes integral

$$\mathbf{E}(T) = \int_{-\infty}^\infty tdF(t) = \int_0^\infty tdF(t),$$

where $F(\cdot)$ is a cumulative density function. Let's prove that

$$\int_0^\infty tdF(t) = \int_0^\infty S(t)dt = \mathbf{E}(T).$$

Recall the formula for integration by parts of the Riemann-Stieltjes integral

$$\int_a^b g(t)d\alpha(t) + \int_a^b \alpha(t)dg(t) = g(b)\alpha(b) - g(a)\alpha(a)$$

Clearing, we get

$$\int_a^b \alpha(t)dg(t) = g(b)\alpha(b) - g(a)\alpha(a) - \int_a^b g(t)d\alpha(t)$$

Regarding the expected value definition, we have $g(t) = F(t)$, $\alpha(t) = t$, $a = 0$, and $F(t = a) = 0$ because $F(\cdot)$ is a cumulative density function. Hence

$$\forall b > 0,\ \int_0^b tdF(t) = bF(b) - \int_0^b F(t)dt = \int_0^b [F(b) - F(t)]dt.$$

Since $F(b) < 1$ and $1 - F(t) \geq 0$, we have

$$\int_0^b tdF(t) = \int_0^b [F(b) - F(t)]dt \leq \int_0^\infty [1 - F(t)]dt,\ \forall b > 0,$$

so

$$\int_0^\infty tdF(t) = \lim_{b\to\infty} \int_0^b tdF(t) \leq \int_0^\infty [1 - F(t)]dt.$$

On the other hand, let $\lambda > 0$. If $b > \lambda$, then

$$\begin{aligned}\int_0^b [F(b) - F(t)]dt &\geq \int_0^\lambda [F(b) - F(t)]dt \\ &= \int_0^\lambda [F(b) - 1]dt + \int_0^\lambda [1 - F(t)]dt \\ &= \lambda[F(b) - 1] + \int_0^\lambda [1 - F(t)]dt,\end{aligned}$$

and therefore,

$$\begin{aligned}\int_0^\infty tdF(t) = \lim_{b\to\infty} \int_0^b tdF(t) &= \lim_{b\to\infty} \int_0^b [F(b) - F(t)]dt \\ &\geq \int_0^\lambda [1 - F(t)]dt + \lim_{b\to\infty} \lambda[F(b) - 1] \\ &= \int_0^\lambda [1 - F(t)]dt.\end{aligned}$$

Since this is true for all $\lambda > 0$, we have

$$\int_0^\infty tdF(t) = \lim_{\lambda\to\infty} \int_0^\lambda [1 - F(t)]dt = \int_0^\infty [1 - F(t)]dt,$$

which is equivalent to the survival function. Finally

$$\mathbf{E}(T) = \int_0^\infty t dF(t) = \int_0^\infty S(t) dt,$$

□

B. Appendix: Derivative of the TTT transform

> *The ability to simplify means to eliminate the unnecessary so that the necessary may speak.*
>
> — **Hans Hofmann**
>
> (Expressionist painter)

Let's show that

$$\frac{d}{du}H_{F^{-1}}(u) = \frac{1}{h(t)}.$$

Proof. Firstly, let's compute the derivative of the TTT transform,

$$\begin{aligned}\frac{d}{du}H_{F^{-1}}(u) &= \frac{d}{du}\int_0^{F^{-1}(u=1)} S(t)dt \\ &= \underbrace{\left[\frac{d}{du}F^{-1}(u)\right]}_{\text{Member A}} \underbrace{\left[S(F^{-1}(u=1))\right]}_{\text{Member B}} - \cancelto{0}{\left(\frac{d}{du}0\right)} S(0) + \cancelto{0}{\int_0^{F^{-1}(u=1)} \frac{d}{du}S(t)dt}\end{aligned}$$

The second and third terms have derivatives equals to zero. Let's develop in more detail the first one.

Member A Let's say that

$$t = F^{-1}(u)$$

so,

$$F(t) = u \quad \text{and} \quad \frac{dt}{du} = \frac{d}{du}F^{-1}(u) \tag{B-1}$$

then, regarding B-1

$$\frac{dt}{du} = 1\Big/\frac{du}{dt} = 1\Big/\frac{d}{dt}F(t)$$

Recall from the definition of the cdf that

$$f(t) = \frac{d}{dt}F(t), \tag{B-2}$$

whence, B-1 can be rewritten as follows

$$\frac{dt}{du} = \frac{1}{f(t)}. \tag{B-3}$$

Therefore,

$$\frac{d}{du}F^{-1}(u=1) = \frac{1}{f(F^{-1}(u))}.$$

Member B The expression can be rewritten as follows

$$S(F^{-1}(u=1)) = 1 - F(F^{-1}(u=1)) = 1 - u.$$

Final result Combining the two previous results and evaluating the cdf in any time yields the following expression

$$\frac{d}{du}H_{F^{-1}}(u) = \left.\frac{1-u}{f(F^{-1}(u))}\right|_{u=F(t)} = \frac{S(t)}{f(t)} = \frac{1}{h(t)}.$$

□

C. Appendix: Scaled TTT transform for censored samples

> *God is in the details.*
>
> — **Ludwig Mies van der Rohe**
>
> (Architect and furniture designer)

Recall the definition of the TTT transform

$$\phi_n\left(\frac{i}{n}\right) = \frac{H_{F^{-1}}(i/n)}{H_{F^{-1}}(1)}. \tag{C-1}$$

Using the Kaplan-Meier estimator (KME), the TTT transform could be expressed as follows

$$H_{F^{-1}}\left(\frac{i}{n}\right) = \int_0^{F^{-1}(i/n)} S(t)dt = \int_0^{T_i}\left[\prod_{k:T_k \leq T}\left(1-\frac{d_k}{n_k}\right)\right]dt \tag{C-2}$$

where d_k are the number of events and n_k the number of units in risk up to time k. Since KME is a step function, the integral can be written as a Riemann summation,

$$H_{F^{-1}}\left(\frac{i}{n}\right) = \sum_{j=1}^{i}\left[\prod_{k=1}^{j}\left(1-\frac{d_k}{n_k}\right)(T_j - T_{j-1})\right] \tag{C-3}$$

Therefore, when $i = n$,

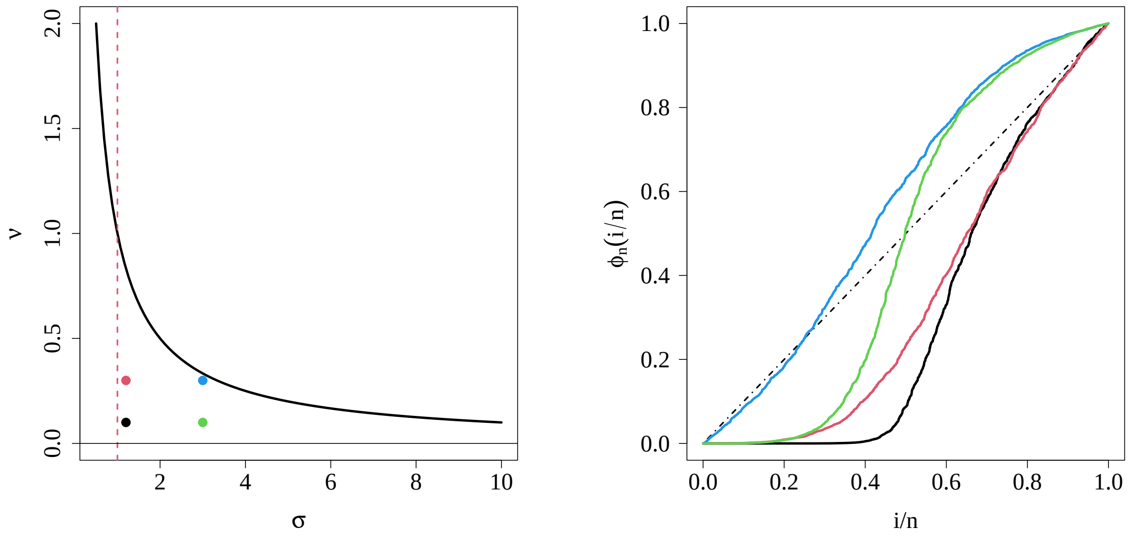
$$H_{F^{-1}}(1) = \sum_{j=1}^{n}\left[\prod_{k=1}^{j}\left(1-\frac{d_k}{n_k}\right)(T_j - T_{j-1})\right] \tag{C-4}$$

Then, the equation C-1 leads to

$$\phi_n\left(\frac{i}{n}\right)=\frac{\sum\limits_{j=1}^{i}\left[\prod\limits_{k=1}^{j}\left(1-\frac{d_k}{n_k}\right)(T_j-T_{j-1})\right]}{\sum\limits_{j=1}^{n}\left[\prod\limits_{k=1}^{j}\left(1-\frac{d_k}{n_k}\right)(T_j-T_{j-1})\right]} \tag{C-5}$$

D. Appendix: odd Weibull parametric sub-spaces and TTT shapes

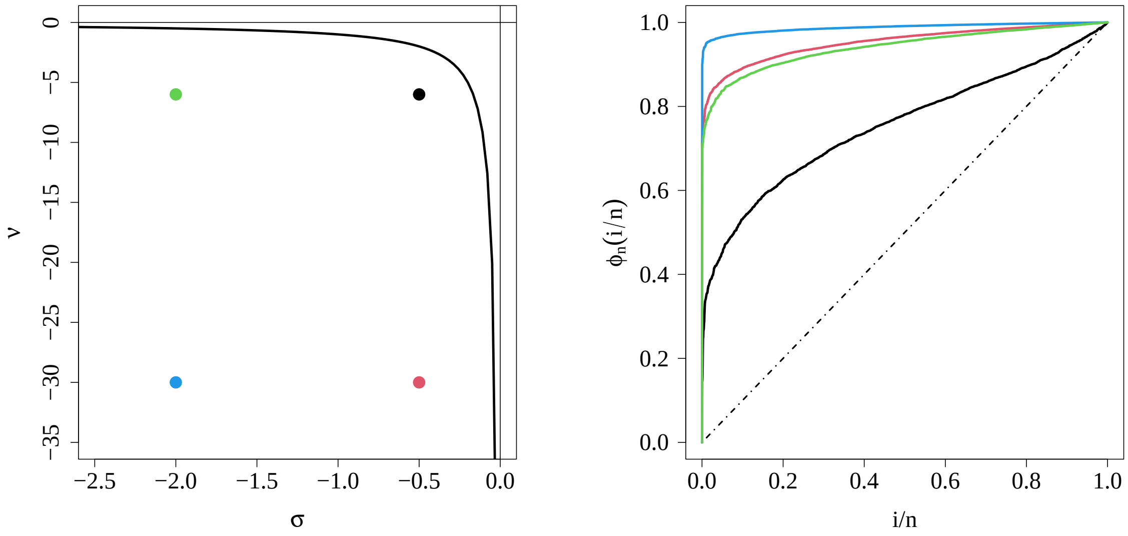
> *A picture is worth a thousand words.*
>
> — **Popular adage**



(a) Bathtub parametric subspace.

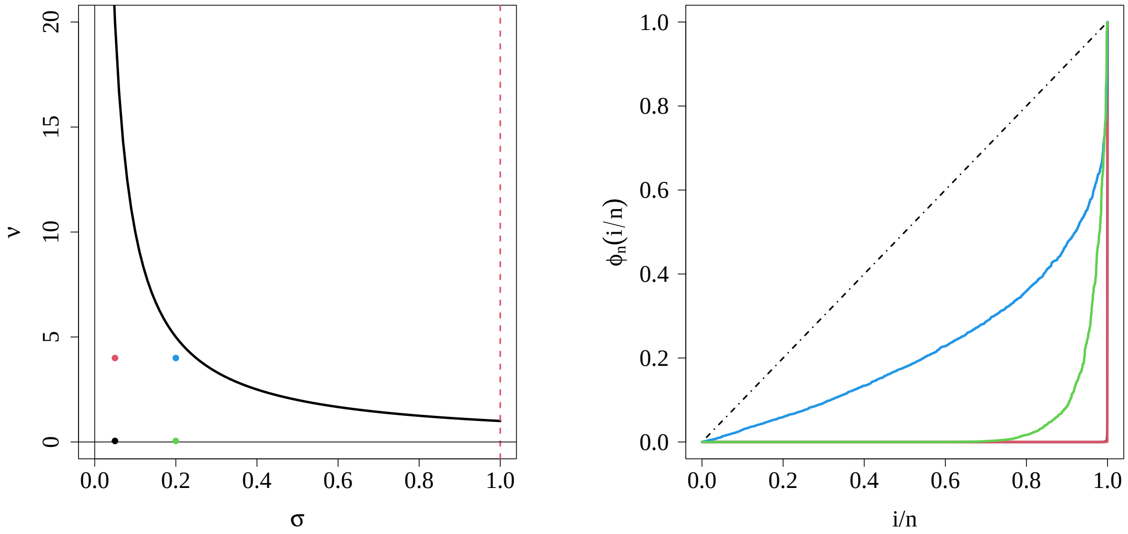
(b) Some TTT plots for values indicated in (a).

Figure D-1: Parametric sub-spaces and TTT plots corresponding to data with bathtub hazard function shapes. TTT plots are computed with simulated OW variables (sample size of 2000).



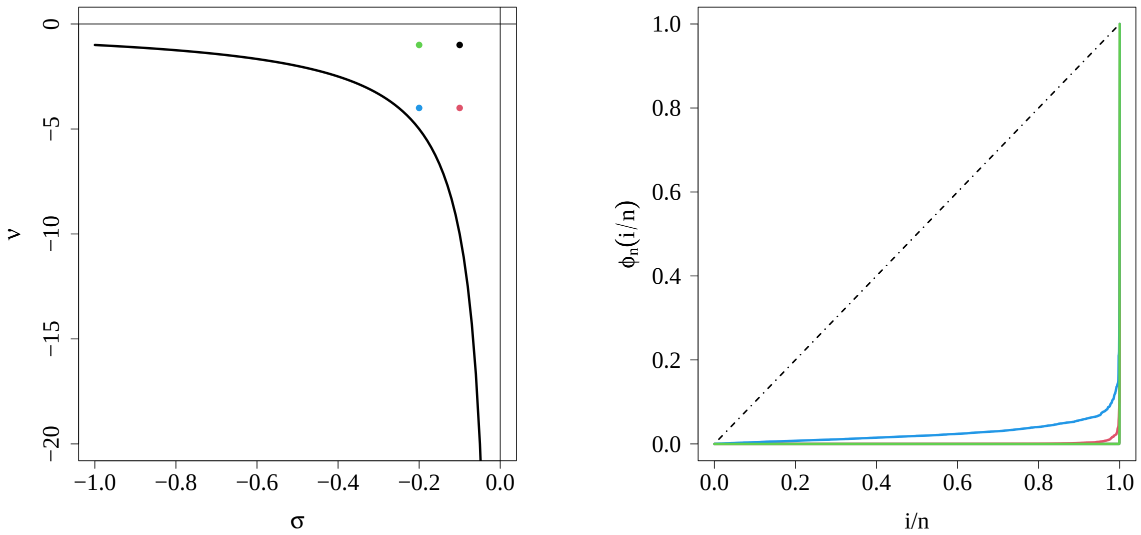
(a) Unimodal (negative) parametric subspace. (b) Some TTT plots for values indicated in (a).

Figure D-2: Parametric sub-spaces (quadrant III) and TTT plots corresponding to data with unimodal hazard function shapes. TTT plots are computed with simulated OW variables (sample size of 2000).



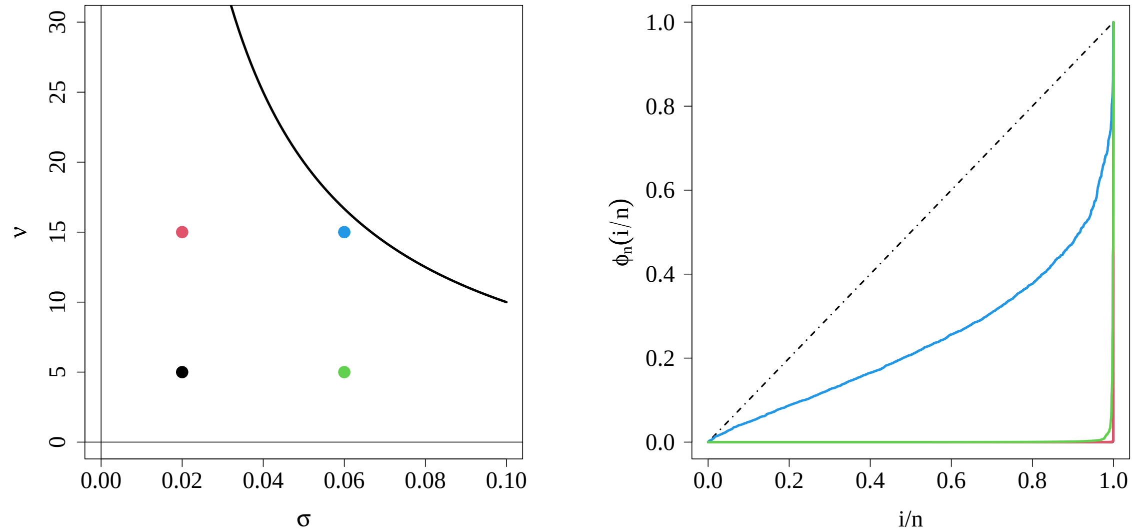
(a) Decreasing (postive) parametric subspace. (b) Some TTT plots for values indicated in (a).

Figure D-3: Parametric sub-spaces (quadrant I) and TTT plots corresponding to data with decreasing hazard function shapes. TTT plots are computed with simulated OW variables (sample size of 2000).



(a) Decreasing (negative) parametric subspace. (b) Some TTT plots for values indicated in (a).

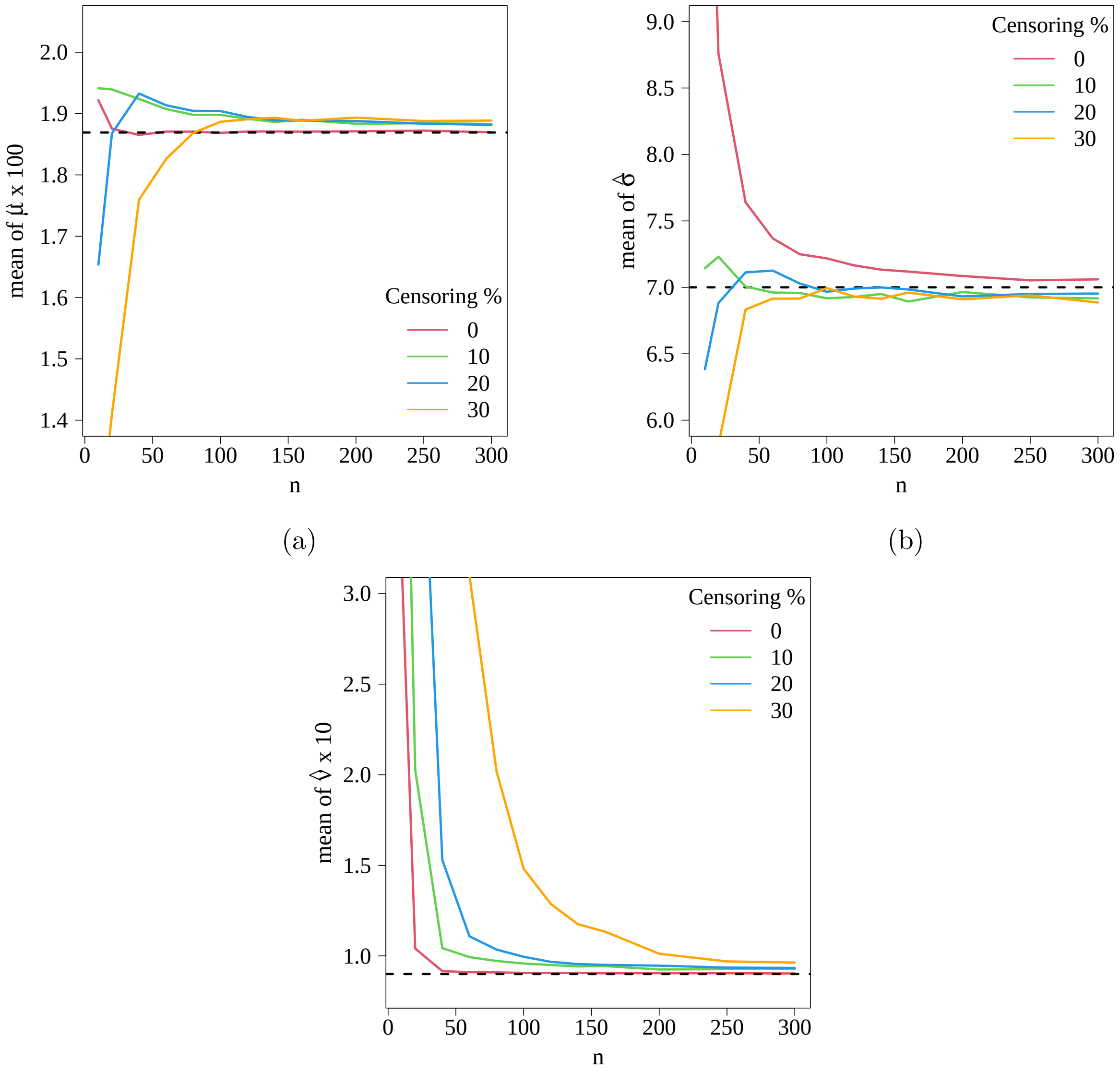
Figure D-4: Parametric sub-spaces (quadrant III) and TTT plots corresponding to data with decreasing hazard function shapes. TTT plots are computed with simulated OW variables (sample size of 2000).



(a) Decreasing (postive) parametric subspace. (b) Some TTT plots for values indicated in (a).

Figure D-5: Parametric sub-spaces (quadrant I, $\sigma < 1$) and TTT plots corresponding to data with decreasing hazard function shapes. TTT plots are computed with simulated OW variables (sample size of 2000).

E. Appendix: odd Weibull simulation study

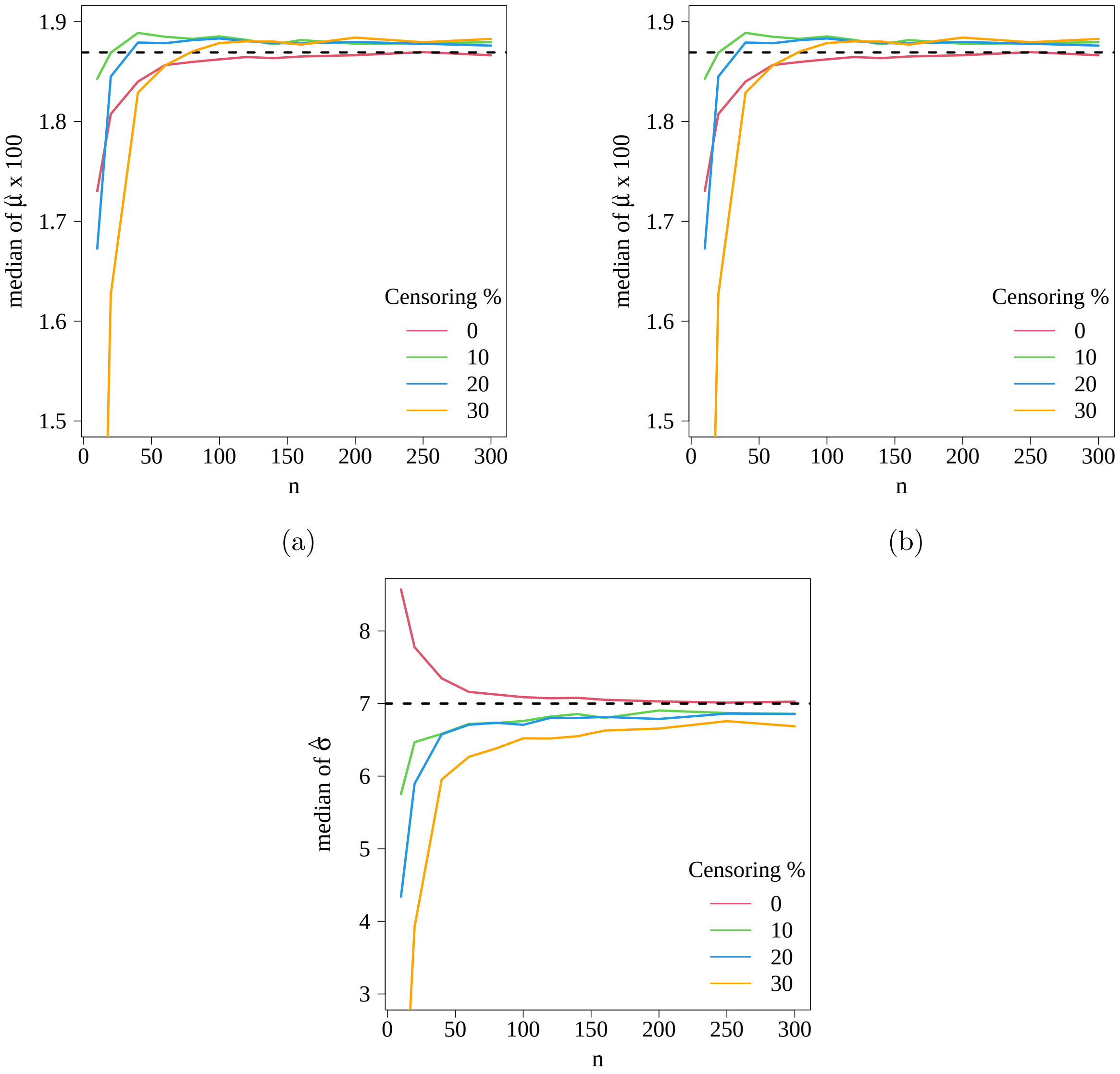


(a)

(b)

(c)

Figure E-1: Mean value for estimated parameters. The dashed line indicates the true values of the parameters.

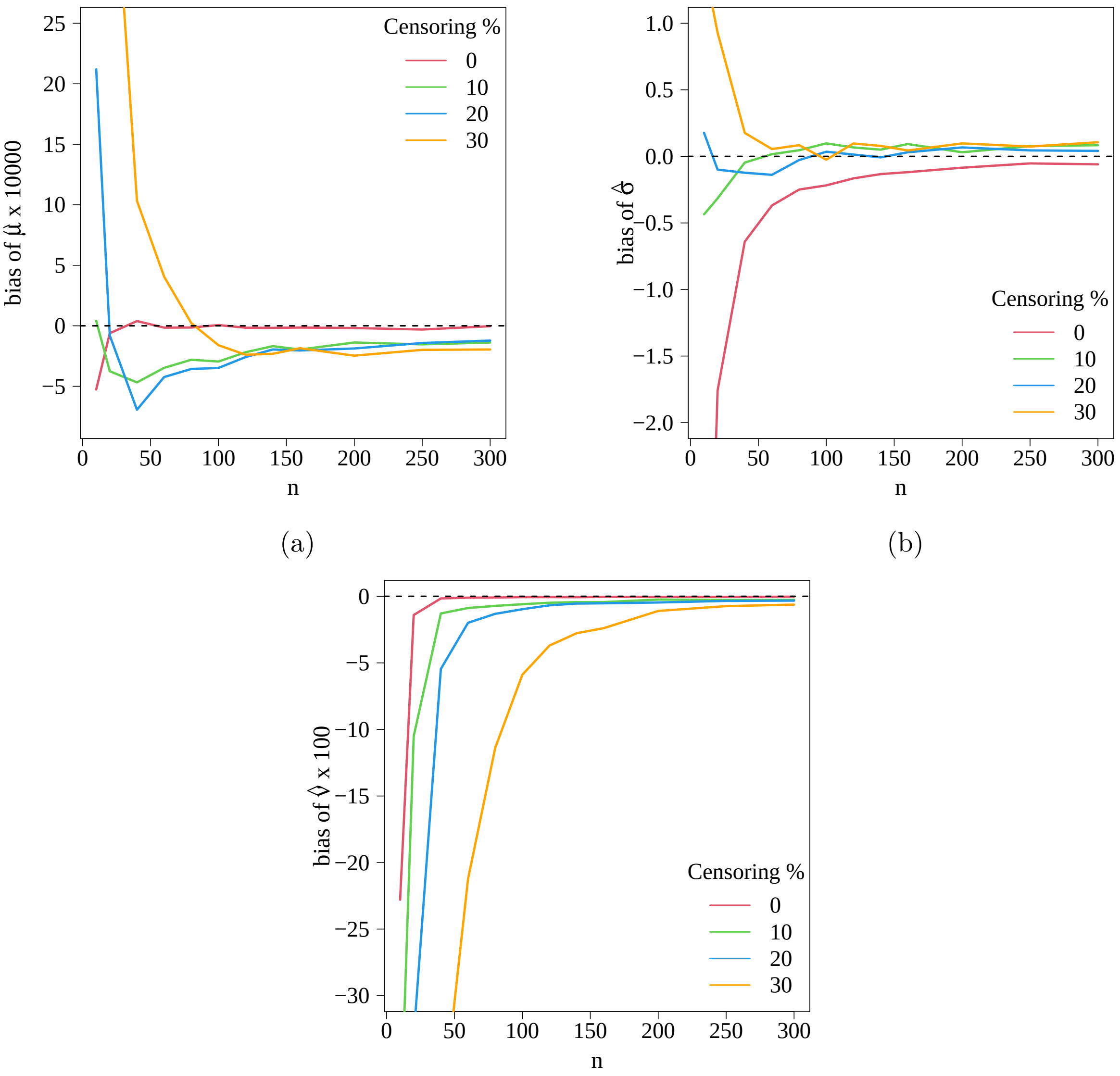


(a)

(b)

(c)

Figure E-2: Median value for estimated parameters. The dashed line indicates the true values of the parameters.



(a)

(b)

(c)

Figure E-3: Bias for estimated parameters. The dashed line indicates Bias = 0.

Part VI

References

References

Aarset, M. V. (1987). How to Identify a Bathtub Hazard Rate. *IEEE Transactions on Reliability*, R-36(1):106–108.

Abdi, M., Asgharzadeh, A., Bakouch, H. S., & Alipour, Z. (2019). A New Compound Gamma and Lindley Distribution with Application to Failure Data. *Austrian Journal of Statistics*, 48(3):54–75.

Agresti, A. (2015a). Deviance of a GLM, model comparison, and model checking. In *Foundations of Linear and Generalized Linear Models*, chapter 4, page 132. John Wiley & Sons, Inc., Hoboken, NJ, USA, 1st edition.

Agresti, A. (2015b). Likelihood-Ratio Model Comparison Uses Deviance Difference. In *Foundations of Linear and Generalized Linear Models*, chapter 4, page 134. John Wiley & Sons, Inc., Hoboken, NJ, USA, 1st edition.

Alanzi, A. R. A., Jamal, F., Tahir, M. H., Chesneau, C., Kanwal, S., & Sami, W. (2023). A New Detection Function Model for Distance Sampling Based on the Burr XII Model. *Symmetry*, 15(3):620.

Almalki, S. J. & Nadarajah, S. (2014). Modifications of the Weibull distribution: A review. *Reliability Engineering & System Safety*, 124:32–55.

Asquith, W. H. (2018). lmomco—L-moments, censored L-moments, trimmed L-moments, L-comoments, and many distributions.

Aucoin, F. (2015). FAdist: Distributions that are Sometimes Used in Hydrology.

Bailey, D. H. & Borwein, J. M. (2015). High-Precision Arithmetic : Progress and Challenges. *Mathematics*, 3:337–367.

Barlow, W. E. & Prentice, R. L. (1988). Residuals for Relative Risk Regression. *Biometrika*, 75(1):65.

Bartoš, F. (2021). *BayesTools: Tools for Bayesian Analyses.*

Bebbington, M., Lai, C.-D., & Zitikis, R. (2006). Useful periods for lifetime distributions with Bathtub shaped hazard rate functions. *IEEE Transactions on Reliability*, 55(2):245–251.

Bebbington, M., Lai, C. D., & Zitikis, R. (2007). A flexible Weibull extension. *Reliability Engineering & System Safety*, 92(6):719–726.

Bergman, B. & Klefsjo, B. (1984). Total Time on Test Concept and Its Use in Reliability Theory. *Operations Research*, 32(3):596–606.

Borchers, H. W. (2022). *numbers: Number-Theoretic Functions.*

Broström, G. (2018). *Event History Analysis with R.* CRC Press.

Broström, G. (2020). eha: Event History Analysis.

Buckland, S. T. (1992). Algorithm AS 270: Maximum Likelihood Fitting of Hermite and Simple Polynomial Densities. *Applied Statistics*, 41(1):241.

Buckland, S. T., Anderson, D. R., Burnham, K. P., & Laake, J. L. (1993). *Distance Sampling.* Springer Netherlands, Dordrecht.

Byrd, R. H., Lu, P., Nocedal, J., & Zhu, C. (1995). A Limited Memory Algorithm for Bound Constrained Optimization. *SIAM Journal on Scientific Computing*, 16(5):1190–1208.

Canty, A. & Ripley, B. D. (2017). boot: Bootstrap R (S-Plus) Functions.

Carlson, B. W. (2020). Simpson's paradox.

Carrasco, J. M. F., Ortega, E. M. M., & Cordeiro, G. M. (2008). A generalized modified Weibull distribution for lifetime modeling. *Computational Statistics and Data Analysis*, 53(2):450–462.

Casella, G. & Berger, R. L. (2002). The Sufficience Principle. In *Statistical inference*, volume 2, page 281. Duxbury Pacific Grove, CA.

Cohen, A. C. (1973). The reflected Weibull distribution. *Technometrics*, 15(4):867–873.

Colosimo, E. A. & Ruiz Giolo, S. (2006a). Censura e Dados Truncados. In *Análise de Sobrevivência Aplicada*, chapter 1, pages 8–9. Edgard Blucher, São Paulo.

Colosimo, E. A. & Ruiz Giolo, S. (2006b). Modelos de Regressão Parametricos. In *Análise de Sobrevivência Aplicada*, chapter 4, pages 123–134. Edgard Blucher, São Paulo.

Cooray, K. (2006). Generalization of the Weibull distribution: The odd Weibull family. *Statistical Modelling*, 6(3):265–277.

Cooray, K. (2015). A study of moments and likelihood estimators of the odd Weibull distribution. *Statistical Methodology*, 26:72–83.

Cordeiro, G. M. & NETO, E. d. A. L. (2004). Modelos paramétricos. *Pernambuco: UFRPE.*

Cordeiro, G. M., Ortega, E. M., & Nadarajah, S. (2010). The Kumaraswamy Weibull distribution with application to failure data. *Journal of the Franklin Institute*, 347(8):1399–1429.

Cox, D. R. (1972). Regression Models and Live-tables. *Journal of the Royal Statistical Society. Series B (Methodological)*, 34(2):187–220.

Cox, D. R. & Snell, E. J. (1968). A General Definition of Residuals. *Journal of the Royal Statistical Society. Series B (Methodological)*, 30(2):248–275.

Dai, Y. & Yuan, Y. (2001). An Efficient Hybrid Conjugate Gradient Method for Unconstrained Optimization. *Annals of Operations Research*, 103(1-4):33–47.

Dai, Y. H. & Yuan, Y. (1999). A nonlinear conjugate gradient method with a strong global convergence property. *SIAM Journal on Optimization*, 10(1):177–182.

Daniels, H. E. (1961). Efficiency of a maximum likelihood estimator. In Neyman, J., editor, *Proceedings of the Fourth Berkeley Symposium on Mathematical Statistics and Probability, Volume 1: Contributions to the Theory of Statistics*, pages 151–163, Berkeley. University of California Press.

Davison, A. C. & Hinkley, D. V. (1997). *Bootstrap Methods and Their Applications.* Cambridge University Press, Cambridge.

De Bruijn, N. G. (1981). *Asymptotic methods in analysis*, volume 4. Courier Corporation.

de Valpine, P., Turek, D., Paciorek, C., Anderson-Bergman, C., Temple Lang, D., & Bodik, R. (2017). Programming with models: writing statistical algorithms for general model structures with NIMBLE. *Journal of Computational and Graphical Statistics*, 26(2):403–413.

Delignette-Muller, M. L. & Dutang, C. (2015). fitdistrplus : An R Package for Fitting Distributions. *Journal of Statistical Software*, 64(4):1–34.

Devendra, K. & Rangaswamy, T. (2013). Strength Characterization of E-glass Fiber Reinforced Epoxy Composites with Filler Materials. *Journal of Minerals and Materials Characterization and Engineering*, 01(06):353–357.

Diaconis, P. (1987). Application of the Method of Moments in Probability and Statistics. In Landau, H., editor, *Moments in Mathematics: Proceedings of Symposia in Applied Mathematics*, volume 37 of *Proceedings of Symposia in Applied Mathematics*, pages 125–142. American Mathematical Society, Providence, Rhode Island.

Didonato, A. R. & Morris, A. H. (1992). Algorithm 708: Significant digit computation of the incomplete beta function ratios. *ACM Transactions on Mathematical Software*, 18(3):360–373.

Dorsey, R. E. & Mayer, W. J. (1995). Genetic algorithms for estimation problems with multiple optima, nondifferentiability, and other irregular features. *Journal of Business & Economic Statistics*, 13(1):53–66.

Drapella, A. (1993). The complementary Weibull distribution: Unknown or just forgotten? *Quality and Reliability Engineering International*, 9(4):383–385.

Dunn, P. K. & Smyth, G. K. (1996). Randomized Quantile Residuals. *Journal of Computational and Graphical Statistics*, 5(3):236.

Dunn, P. K. & Smyth, G. K. (2018). *Generalized Linear Models With Examples in R*. Springer Texts in Statistics. Springer New York, New York, NY.

Efron, B. (1988). Logistic Regression, Survival Analysis, and the Kaplan-Meier Curve. *Journal of the American Statistical Association*, 83(402):414.

Eisinga, R., Heskes, T., Pelzer, B., & Te Grotenhuis, M. (2017). Exact p-values for pairwise comparison of Friedman rank sums, with application to comparing classifiers. *BMC Bioinformatics*, 18(1):68.

Escobar, L. A. & Meeker, W. Q. (2006). A Review of Accelerated Test Models. *Statistical Science*, 21(4):552–577.

Famoye, F., Lee, C., & Olumolade, O. (2005). The beta-Weibull distribution. *Journal of Statistical Theory and Applications*, 4:121–136.

Fisher, R. (1912). On an Absolute Criterion for Fitting Frequency Curves. *Messenger of Mathematics*, 41(1):155–160.

Fisher, R. A. (1922). On the Mathematical Foundations of Theoretical Statistics. *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 222(594-604):309–368.

Fisher, R. A. (1958). *Statistical Methods for Research Workers*. Oliver & Boyd, Edinburgh, 13th edition.

Fletcher, R. (1964). Function minimization by conjugate gradients. *The Computer Journal*, 7(2):149–154.

Fletcher, R. (1987). *Practical Methods of Optimization*. John Wiley & Sons, New York, 2nd edition.

Fox, P. A., Hall, A. P., & Schryer, N. L. (1978). The PORT Mathematical Subroutine Library. *ACM Transactions on Mathematical Software*, 4(2):104–126.

Gelman, A., Simpson, D., & Betancourt, M. (2017). The Prior Can Often Only Be Understood in the Context of the Likelihood. *Entropy*, 19(10):555.

Ghitany, M., Al-Mutairi, D., Balakrishnan, N., & Al-Enezi, L. (2013). Power Lindley distribution and associated inference. *Computational Statistics & Data Analysis*, 64:20–33.

Goulet, V. (2008). actuar: An R Package for Actuarial Science. *Journal of Statistical Software*, 25(7).

Granzotto, D. C. T., Dos Santos, C. A., & Louzada, F. (2018). The Transmuted Weibull Regression Model: an Application to Type 2 Diabetes Mellitus Data. *International Journal of Statistics and Probability*, 7(2):1.

Gruman, J. (2021). Survival Analysis.

Gumbel, E. J. (1941). The Return Period of Flood Flows. *Annals of Statistics*, pages 163–190.

Gumbel, E. J. (1958). *Statistics of Extremes.* Columbia University Press, New York, university edition.

Gurland, J. (1954). On regularity conditions for maximum likelihood estimators. *Scandinavian Actuarial Journal*, 1954(1):71–76.

Halabi, S., Dutta, S., Wu, Y., & Liu, A. (2020). Score and deviance residuals based on the full likelihood approach in survival analysis. *Pharmaceutical Statistics*, 19(6):940–954.

Hall, D. (2021). *bignum: Arbitrary-Precision Integer and Floating-Point Mathematics.*

Haupt, R. L. & Haupt, S. E. (2003). *Practical Genetic Algorithms.* John Wiley & Sons, Inc., Hoboken, NJ, USA.

Hayes, A., Moller-Trane, R., Jordan, D., Northrop, P., Lang, M. N., & Zeileis, A. (2022). *distributions3: Probability Distributions as S3 Objects.*

Henningsen, A. & Toomet, O. (2011). maxLik: A package for maximum likelihood estimation in R. *Computational Statistics*, 26(3):443–458.

Hernandez, F., Usuga, O., Patino, C., Mosquera, J., & Urrea, A. (2023). *RelDists: Estimation for some Reliability Distributions.* Medellin, Colombia.

IEEE and Open Group (2004a). The Open Group Base Specifications Issue 6 — expm1.

IEEE and Open Group (2004b). The Open Group Base Specifications Issue 6 — log1p.

Jackson, C. (2016). flexsurv : A Platform for Parametric Survival Modeling in R. *Journal of Statistical Software*, 70(8).

James, B. R. (2010). Esperança Matemática. In *Probabilidade: um curso em nível intermediário*, chapter 3, page 112. 5th edition.

Jiang, H., Xie, M., & Tang, L. C. (2008). On the odd Weibull distribution. *Proceedings of the Institution of Mechanical Engineers, Part O: Journal of Risk and Reliability*, 222(4):583–594.

Johnson, S. G. (2022). The NLopt nonlinear-optimization package.

Kalbfleisch, J. D. & Prentice, R. L. (2002). Regression models. In *The Statistical Analysis of Failure Time Data*, page 40. John Wiley & Sons, Inc.

Kaplan, E. L. & Meier, P. (1958). Nonparametric Estimation from Incomplete Observations. *Journal of the American Statistical Association*, 53(282):457.

Karvanen, J. (2006). Estimation of quantile mixtures via L-moments and trimmed L-moments. *Computational Statistics & Data Analysis*, 51(2):947–959.

Karvanen, J. (2019). Lmoments: L-moments and quantile mixtures.

Khan, S. A. (2018). Exponentiated Weibull regression for time-to-event data. *Lifetime Data Analysis*, 24(2):328–354.

Kohl, M. & Ruckdeschel, P. (2010). R Package distrMod: S4 Classes and Methods for Probability Models. *Journal of Statistical Software*, 35(10).

Lai, C. D., Xie, M., & Murthy, D. N. P. (2003). A modified Weibull distribution. *IEEE Transactions on Reliability*, 52(1):33–37.

Lange, K. (2010). Newton's Method and Scoring. In *Numerical Analysis for Statisticians*, Statistics and Computing, chapter 11. Springer New York, New York, NY.

Lawless, J. F. (2002). Parametric Regression models. In *Statistical Models and Methods for Lifetime Data*, Wiley Series in Probability and Statistics, pages 34–35. John Wiley & Sons, Inc., Hoboken, NJ, USA, 2nd edition.

Lawless, J. F. (2003). Basic Concepts and Models. In *Statistical Models and Methods for Lifetime Data*, chapter 1, page 1. Wiley, 2nd edition.

Lee, E. T. & Wang, J. W. (2003). Type I Censoring. In *Statistical Methods for Survival Data Analysis*, chapter 1, page 2. John Wiley & Sons, Inc., New Jersey, NJ, USA, third edition.

Leung, K.-M., Elashoff, R. M., & Afifi, A. A. (1997). Censoring issues in survival analysis. *Annual Review of Public Health*, 18(1):83–104.

Loprinzi, C. L., Laurie, J. A., Wieand, H. S., Krook, J. E., Novotny, P. J., Kugler, J. W., Bartel, J., Law, M., Bateman, M., & Klatt, N. E. (1994). Prospective evaluation of prognostic variables from patient-completed questionnaires. North Central Cancer Treatment Group. *Journal of clinical oncology : official journal of the American Society of Clinical Oncology*, 12(3):601–7.

Lucas, A., Scholz, I., Boehme, R., Jasson, S., & Mächler, M. (2023). *gmp: Multiple Precision Arithmetic*.

Mächler, M. (2011). Arbitrarily Accurate Computation with R: The Rmpfr Package. Technical report, ETH Zurich.

Mächler, M. (2012). Accurately Computing log(1 - exp(-—a—)) Assessed by the Rmpfr package. Technical report.

Mächler, M. (2022). *Rmpfr: R MPFR - Multiple Precision Floating-Point Reliable*.

Marinho, P. R. D., Silva, R. B., Bourguignon, M., Cordeiro, G. M., & Nadarajah, S. (2019). AdequacyModel: An R package for probability distributions and general purpose optimization. *PLOS ONE*, 14(8):e0221487.

Marshall, A. W. & Olkin, I. (2007). *Life Distributions*. Springer Series in Statistics.

Martin, A. D., Quinn, K. M., & Park, J. H. (2011). MCMCpack: Markov Chain Monte Carlo in R. *Journal of Statistical Software*, 42(9):22.

Mazucheli, J., Coelho-Barros, E. A., & Achcar, J. A. (2013). The exponentiated exponential mixture and non-mixture cure rate model in the presence of covariates. *Computer Methods and Programs in Biomedicine*, 112(1):114–124.

McCullagh, P. & Nelder, J. A. (1989). Deviance Residuals. In *Generalized Linear Models*, chapter 2, pages 39–40. CRC Press, London, 2nd edition.

Meeker, W. Q. & Escobar, L. A. (1998). Distinguishing Features of Reliability Data. In *Statistical Methods for Reliability Data*, chapter 1, page 3. John Wiley & Sons, Inc.

Mosquera Gutiérez, J. & Hernández, F. (2023). EstimationTools.

Mudholkar, G. S. & Hutson, A. D. (1996). The exponentiated weibull family: some properties and a flood data application. *Communications in Statistics - Theory and Methods*, 25(12):3059–3083.

Mudholkar, G. S. & Kollia, G. D. (1994). Generalized weibull family: a structural analysis. *Communications in Statistics - Theory and Methods*, 23(4):1149–1171.

Mudholkar, G. S. & Srivastava, D. K. (1993). Exponentiated Weibull Family for Analyzing Bathtub Failure-Rate Data. *IEEE Transactions on Reliability*, 42(2):299–302.

Muhammad, M., Bantan, R. A. R., Liu, L., Chesneau, C., Tahir, M. H., Jamal, F., & Elgarhy, M. (2021). A New Extended Cosine—G Distributions for Lifetime Studies. *Mathematics*, 9(21):2758.

Murdock, Bennet B., J. (1961). The retention of individual items. *Journal of Experimental Psychology*, 62(6):618–625.

Murthy, P., Xie, M., & Jiang, R. (2004). Taxonomy for Weibull Models. In *Weibull Models*, pages 18–37. John Wiley & Sons, Inc.

Myung, I. J. (2003). Tutorial on maximum likelihood estimation. *Journal of Mathematical Psychology*, 47(1):90–100.

Nachlas, J. A. (2017). *Reliability Engineering.* CRC Press.

Nadarajah, S. & Rocha, R. (2016). Newdistns : An R Package for New Families of Distributions. *Journal of Statistical Software*, 69(10).

Nair, N. U., Sankaran, P. G., & Balakrishnan, N. (2018). Chapter 5 - Bathtub Distributions. In Nair, N. U., Sankaran, P. G., & Balakrishnan, N., editors, *Reliability Modelling and Analysis in Discrete Time*, pages 247–279. Academic Press, Boston.

Nakagawa, T. & Osaki, S. (1975). The Discrete Weibull Distribution. *IEEE Transactions on Reliability*, R-24(5):300–301.

Nash, J. C. (1979). *Compact Numerical Methods for Computers. Linear Algebra and Function Minimisation.* Adam Hilger, Bristol, 2nd editio edition.

Nash, J. C. (2014a). Conjugate gradient and related methods. In *Nonlinear parameter optimization using R tools*, pages 18–19. John Wiley & Sons.

Nash, J. C. (2014b). Quasi-Newton or variable metric method. In *Nonlinear parameter optimization using R tools*, chapter 2, pages 17–18. John Wiley & Sons.

Nash, J. C. (2014c). Rcgmin: Conjugate Gradient Minimization of Nonlinear Functions.

Nash, J. C. (2018). Rvmmin: Variable Metric Nonlinear Function Minimization.

Nelder, J. A. & Mead, R. (1965). A Simplex Method for Function Minimization. *The Computer Journal*, 7(4):308–313.

Nielsen, H. B. & Mortensen, S. B. (2016). ucminf: General-Purpose Unconstrained Non-Linear Optimization.

Ñíguez, T.-M., Paya, I., Peel, D., & Perote, J. (2019). Flexible distribution functions, higher-order preferences and optimal portfolio allocation. *Quantitative Finance*, 19(4):699–703.

Nikulin, M. & Haghighi, F. (2007). A chi-squared test for the genralized power Weibull family for the head-and-neck cancer censored data. *Journal of Mathematical Sciences*, 142(3):2204–2204.

O'Hara-Wild, M., Kay, M., & Hayes, A. (2023). *distributional: Vectorised Probability Distributions*.

Ortega, E. M. M., Cordeiro, G. M., & Carrasco, J. M. F. (2011). The log-generalized modified Weibull regression model. *Brazilian Journal of Probability and Statistics*, 25(1):64–89.

Padgett, W. J. & Spurrier, J. D. (1985). On Discrete Failure Models. *IEEE Transactions on Reliability*, R-34(3):253–256.

Panja, S. C. & Ray, P. K. (2007). Reliability analysis of track circuit of Indian railway signalling system. *International Journal of Reliability and Safety*, 1(4):428.

Pawitan, Y. (2013a). Continuos data (example 4.8). In *In all likelihood: statistical modelling and inference using likelihood*, chapter 4, pages 90–91. Oxford University Press.

Pawitan, Y. (2013b). *In all likelihood: statistical modelling and inference using likelihood*. Oxford University Press.

Pearson, K. (1936). Method of Moments and Method of Maximum Likelihood. *Biometrika*, 28(1/2):34.

Percontini, A., Blas, B., & Cordeiro, G. (2013). The beta Weibull Poisson distribution. *Chilean Journal of Statistics*, 4(2):3–26.

Poncet, P. (2019). modeest: Mode Estimation.

Prataviera, F., Ortega, E. M., Cordeiro, G. M., Pescim, R. R., & Verssani, B. A. (2018). A new generalized odd log-logistic flexible Weibull regression model with applications in repairable systems. *Reliability Engineering & System Safety*, 176:13–26.

Pregibon, D. (1981). Logistic Regression Diagnostics. *The Annals of Statistics*, 9(4).

Prinja, S., Gupta, N., & Verma, R. (2010). Censoring in clinical trials: Review of survival analysis techniques. *Indian Journal of Community Medicine*, 35(2):217.

R Core Team (2022). Double-Precision Vectors.

R Core Team (2023). R: A Language and Environment for Statistical Computing.

Reid, N. (1994). A Conversation with Sir David Cox. *Statistical Science*, 9(3).

Rigby, R. A. & Stasinopoulos, D. M. (2005). Generalized additive models for location, scale and shape. *Journal of the Royal Statistical Society: Series C (Applied Statistics)*, 54(3):507–554.

Rossi, P. (2023). *bayesm: Bayesian Inference for Marketing/Micro-Econometrics.*

Rossi, R. J. (2018). Likelihood-based Estimation. In *Mathematical Statistics: An Introduction to Likelihood Based Inference*, chapter 5, pages 227–228. John Wiley & Sons, Inc., Hoboken, NJ, USA, 1st edition.

Ruckdeschel, P. & Kohl, M. (2014). General Purpose Convolution Algorithm in S4 Classes by Means of FFT. *Journal of Statistical Software*, 59(4):1–25.

Ruckdeschel, P., Kohl, M., Stabla, T., & Camphausen, F. (2006). S4 Classes for Distributions. *R News*, 6(2):2–6.

Shafaei Nooghabi, M., Reza Mohtashami Borzadaran, G., & Hamid Rezaei Roknabadi, A. (2011). Discrete modified Weibull distribution. *METRON*, 69(2):207–222.

Sheynin, O. (1994). Chebyshev's lectures on the theory of probability. *Archive for History of Exact Sciences*, 46(4):321–340.

Singer, S. & Nelder, J. (2009). Nelder-Mead algorithm.

Stacy, E. W. (1962). A Generalization of the Gamma Distribution. *The Annals of Mathematical Statistics*, 33(3):1187–1192.

Stasinopoulos, D. M. & Rigby, R. A. (2007). Generalized Additive Models for Location Scale and Shape (GAMLSS) in R. *Journal of Statistical Software*, 23(7).

Stasinopoulos, M., Rigby, R. A., Heller, G. Z., Voudouris, V., & De Bastiani, F. (2017a). Diagnostics. In *Flexible Regression and Smoothing: Using GAMLSS in R.* Chapman and Hall/CRC, 1st edition.

Stasinopoulos, M., Rigby, R. A., Heller, G. Z., Voudouris, V., & De Bastiani, F. (2017b). The GAMLSS family of distributions. In *Flexible regression and smoothing using GAMLSS in R*, pages 153–189. Chapman and Hall/CRC, 1st edition.

Statisticat & LLC. (2021). *LaplacesDemon: Complete Environment for Bayesian Inference.*

Stein, W. E. & Dattero, R. (1984). A New Discrete Weibull Distribution. *IEEE Transactions on Reliability*, R-33(2):196–197.

Stigler, S. (2005). Fisher in 1921. *Statistical Science*, 20(1):32–49.

Stigler, S. M. (1977). Do Robust Estimators Work with Real Data? *The Annals of Statistics*, 5(6).

Tang, P.-T. P. (1990). Table-driven implementation of the logarithm function in IEEE floating-point arithmetic. *ACM Transactions on Mathematical Software*, 16(4):378–400.

Tang, P. T. P. (1992). Table-driven implementation of the Expm1 function in IEEE floating-point arithmetic. *ACM Transactions on Mathematical Software*, 18(2):211–222.

Therneau, T. & Grambsch, P. (2000a). *Modeling Survival Data: Extending the Cox Model*. Statistics for Biology and Health. Springer New York, New York, NY.

Therneau, T. M. & Grambsch, P. M. (2000b). Fraility Models. In *Modeling Survival Data: Extending the Cox Model*, Statistics for Biology and Health, pages 231–233. Springer New York, New York, NY.

Therneau, T. M., Grambsch, P. M., & Fleming, T. R. (1990). Martingale-Based Residuals for Survival Models. *Biometrika*, 77(1):147.

Turkson, A. J., Ayiah-Mensah, F., & Nimoh, V. (2021). Handling Censoring and Censored Data in Survival Analysis: A Standalone Systematic Literature Review. *International Journal of Mathematics and Mathematical Sciences*, 2021:1–16.

Venables, W. N. & Ripley, B. D. (2013). *Modern applied statistics with S-PLUS*. Springer Science & Business Media.

Wagh, Y. S. & Kamalja, K. K. (2018). Zero-inflated models and estimation in zero-inflated Poisson distribution. *Communications in Statistics - Simulation and Computation*, 47(8):2248–2265.

Wald, A. (1943). Tests of Statistical Hypotheses Concerning Several Parameters When the Number of Observations is Large. *Transactions of the American Mathematical Society*, 54(3):426.

Wang, B., Wu, P., Kwan, B., Tu, X. M., & Feng, C. (2018). Simpson's Paradox: Examples. *Shanghai archives of psychiatry*, 30(2):139–143.

Wang, F. K. (2000). A new model with bath tub-shaped failure rate using an additive Burr XII distribution. *Reliability Engineering & System Safety*, 70(3):305–312.

Wang, X.-L. & Li, D.-H. (2011). A modified Fletcher-Reeves-Type derivative-free method for symmetric nonlinear equations. *Numerical Algebra, Control and Optimization*, 1(1):71–82.

Weibull, W. (1951). A Statistical Distribution Function of Wide Applicability. *Journal of applied mechanics*, 103(4):293–297.

Westberg, U. & Klefsjö, B. (1994). TTT-plotting for censored data based on the piecewise exponential estimator. *International Journal of Reliability, Quality and Safety Engineering*, 01(01):1–13.

Whitmore, G. A. (1983). A regression method for censored inverse-Gaussian data. *Canadian Journal of Statistics*, 11(4):305–315.

Xie, M. & Lai, C. (1996). Reliability analysis using an additive Weibull model with bathtub-shaped failure rate function. *Reliability Engineering & System Safety*, 52(1):87–93.

Xie, M., Tang, Y., & Goh, T. N. (2002). A modified Weibull extension with bathtub- shaped failure rate function. *Reliability Engineering & System Safety*, 8320(January).

Zhang, Z. (2016). Parametric regression model for survival data: Weibull regression model as an example. *Annals of Translational Medicine*, 4(24):484–484.

Zhao, Y.-G., Zhang, X.-Y., & Lu, Z.-H. (2018). A flexible distribution and its application in reliability engineering. *Reliability Engineering & System Safety*, 176:1–12.