MASTER'S THESIS

Interuniversity Master in Statistics and Operations Research UPC-UB

Title: Goodness-of-fit R package for Right-censored data

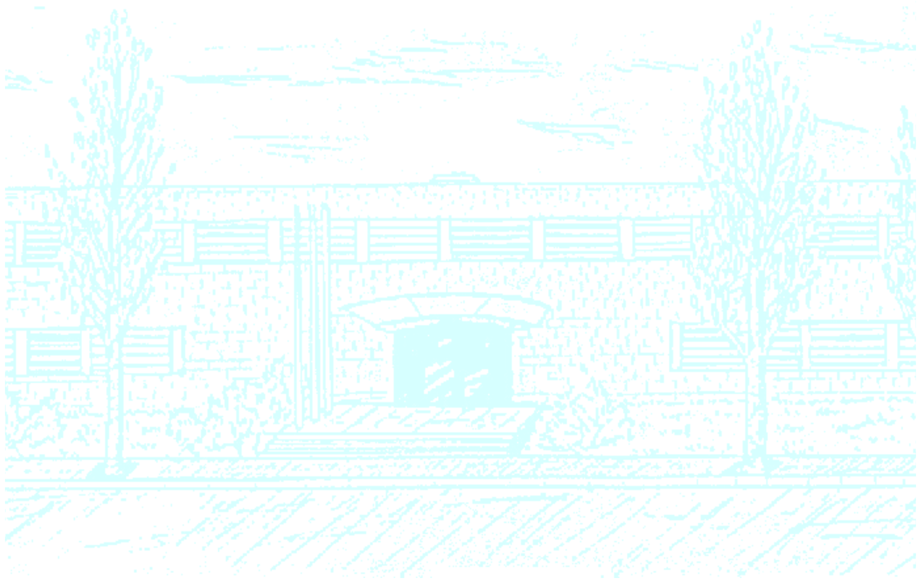
Author: Daniel García Carrasco

Advisor: Guadalupe Gómez Melis

Co-advisors: Klaus Langohr and Mireia Besalú i Mayol

Department: Departament d'Estadística i Investigació Operativa

University: Universitat Politècnica de Catalunya



UNIVERSITAT POLITÈCNICA DE CATALUNYA
BARCELONATECH
Facultat de Matemàtiques i Estadística

UNIVERSITAT DE BARCELONA

Goodness-of-fit R package for Right-censored data

Daniel García Carrasco
Universidad Politécnica de Cataluña
Departamento de Estadística e Investigación Operativa

Gracias a mis tutores, Klaus, Lupe y Mireia, por toda su ayuda y paciencia. Gracias, H., por acompañarme y darme mucho cariño en esta época. Gracias a mis padres por todo. Sin vosotros, esta tesis no sería más que una elucubración.

Abstract

The main objective of this master's degree thesis is to build a R package for the goodness-of-fit techniques for right-censored data. In this work, we'll cover a survey of the survival analysis, especially the right-censoring and the graphical techniques of goodness-of-fit, the state of the art of the survival analysis in R, a description of our package, with its two functions, related to the graphical techniques of goodness of fit, `prob.plot` and `cum.haz.plot`, and its two datasets, and a testing of the functions of our package.

Keywords: R package, goodness of fit, survival analysis, programming, right-censored data

CONTENTS

Chapter 1

INTRODUCTION

1.1 State of the Art

1.1.1 Survival Analysis: an introduction

Survival analysis is a branch of statistics with the purpose of analyzing the duration until the happening of a certain event (or events), under an interval of time, with two main periods, t_0, the time origin, and t_n, the end time of our event or failure time, also symbolized as ε. Now, for precision, we need to state three requirements for the failure event ε: 1) unambiguosly defined time origin; 2) scale for measuring elapsed time; 3) well specified meaning for ε.

This interval of time symbolizes a random non-negative variable, the survival time, with a continuous probability distribution, although under some cases can be discrete. Usually, it's called *survival analysis*, but we have other names as *event history analysis* (Broström, 2012), *event analysis* or *life time data analysis*.

We have some examples of the survival analysis:

- measuring the survival probabilities under different cancer treatments.
- knowing the failure rate of a certain line of machines in t_n
- following up the patients for a certain disease and comparing the survival times of two groups, treatment and control.

1.1.2 Survival, distribution and density functions

As a collection of statistical procedures, survival analysis has its functions. We define the first of our two main functions, the **survival function**:

$$S(t) = P(T > t), \quad t \geq 0 \tag{1.1}$$

In other words, is the probability of an individual to survive longer than a time t. The distribution function is $F(t) = P(T \leq t)$, with a predefined time t^1, so the survival function can be expressed, for a discrete time variable T with $t_1 < t_2 < \cdots < t_n$, as:

$$S(t) = 1 - F(t) = \sum_{t_j > t} P[T = t_j] \tag{1.2}$$

For the continuous case and $f(t)$ as density function:

$$S(t) = 1 - F(t) = 1 - \int_0^t f(u)du = \int_t^\infty f(u)du \tag{1.3}$$

The survival curves can take many different forms with the same properties, monotone and decreasing functions with these two conditions satisfied, $S(0) = 1$ and $\lim_{t\to\infty} S(t) = 0$. The rate of decline is dependent of the risk of experiencing the event at time t.

In last place, we have the **density functions**, for continuous variables, denoted as f, with the following limit:

$$f(t) = \lim_{\Delta t \to 0} \frac{1}{\delta t} P[t \leq T < t + \Delta t] \tag{1.4}$$

Thiss limit measures the intensity of the probability. We can interpret $f(t)\Delta t$ as the probability of happening in $(t, t + \Delta t)$ for the event ε.

1.1.3 Hazard functions

The second main function is the **hazard function**, important for knowing the pattern of the failures in our survival analysis and denoted as $\lambda(t)$, depending of the distribution. We can divide that for discrete, $\lambda(t)$, and continuous cases, $\lambda(t)\Delta t$:

Type of time variable (T)	**Values**	**Hazard Function**	**Range**
Discrete	$t_1 < t_2 < \cdots < t_n$	$\lambda(t_j) = P[T = t_j \mid T \geq t_j] = P[T = t_j \mid T > t_{j-1}]$	$\lambda(t) \in [0, 1]$
Continuous	Continuous	$\lambda(t) = \lim_{\Delta t \to 0} \frac{P[t \leq T < t + \Delta t \mid T \geq t]}{\Delta t}$	$\lambda(t) \in [0, \infty)$

Table 1.1: Hazard functions by type of time variable

Both functions, $\lambda(t)$ and $\lambda(t)\Delta t$, can be seen as the probability of an individual that has not failed by t. With that, we can determine the failure distribution, under many shapes: hump, constant, decreasing, bathtub or any other characteristic, depending on the hazard rate of ocurrence for any event.

[1] The probability of an individual to survive less than time t

1.1.3.1 Cumulative hazard function

It′ is a measure of risk, in where we measure the risk of the failure by time t depending on the value of $\Lambda(t)$, with a positive relationship between both variables[2]. For the discrete time T with values $t_1 < t_2 < \cdots < t_n$, we have:

$$\Lambda(t) = \sum_{t_j \leq t} \lambda(t_j) \tag{1.5}$$

And for the continuous time T, we have:

$$\Lambda(t) = \int_0^t \lambda(u) du \tag{1.6}$$

The cumulative hazard function (CHF) is a non-negative function and monotically increasing. Also, the CHF have a relationship with the hazard and survival functions. For the hazard functions, the relationship for continuous cases are $\Lambda(t) = -\ln S(t)$ or $S(t) = e^{-\Lambda(t)}$.

For the survival functions, under the continuous case, $S(t) = e^{-\int_0^t \lambda(u)du}$ and $f(t) = \lambda(t)S(t)$. For the discrete case, we have the following relationship, taking $t_0 = 0$: $S(t_j) = S(t_{j-1})(1 - \lambda(t_j)), \quad j = 1, 2, \ldots, n$

1.1.4 Censoring

The sequential nature of the response times can lead us to censoring, because we have a probability $1 - p$ to have incomplete information about any of our individuals X_i

Esentially, we have two types of censoring, left and right. In this chapter, we'll talk in detail about the latter.

1.1.4.1 Right censoring

Any individual X_i with a survival time T_i is followed until the ocurrence of event ε. If we observe ε when it happens, then the time until ε is known. If ε doesn't happens during the study, then our individual X_i have right-censoring, because of the following causes:

1. When we finish the study at a predefined t_n. In this case, we have some individuals survived.

2. Lost to follow-up from some individuals (we observe them in any moment less than t_n): $t < t_n$

[2] If we increase a unit of $\Lambda(t)$, we increase the risk of failure too

3. Dropout. We interrupt the treatment for some individuals.

In this case, the censored observation must be recorded, because can be seen as an event (for example, the time 35+ for an individual in a treatment, a censored observation modelled as a non-negative random variable C). There, we observe a minimum between the failure time and the censoring time. We must know if the minimum corresponds to the failure or the censoring time. If we don't have censoring time, any individual have a failure time T_i, distributed as a unknown distribution function F. A period of observation is needed too: it's stopped at C_i if failure has not ocurred by then. The observation is $Y_i = \min(T_i, C_i)$. Then, we model here an indicator variable, δ, with two possible outcomes, censorship, equal to 0, or failure, equal to 1, and represented by the pairs (Y_i, δ_i), denoted as:

$$\delta_i = \begin{cases} 1 & \text{if } T_i \leq C_i, \\ 0 & \text{if } T_i > C_i \end{cases} \tag{1.7}$$

We have different types of right-censoring, depending on the pattern of the censoring times:

- Type I: Fixed, Progressive, Generalized
- Type II: Random

1.1.4.1.1 Type I of censoring **Fixed:** We have a fixed censoring time, from a preassigned observation time for all the individuals which enter at the study at the same time, C_R. Our variant of (Y_i, δ_i) is:

$$Y_i = \min(T_i, C_R) \qquad \text{and} \qquad \delta_i = \begin{cases} 1 & \text{if } T_i \leq C_R, \\ 0 & \text{if } T_i > C_R \end{cases} \tag{1.8}$$

Progressive: It happens when the censoring time is different for each individual (or groups). We establish a finite number of censoring times $C_1, \ldots, C_n$, usually inferior to n. The variables Y and δ are equal.

Generalized: What happens when we consider the different times of entry for each individual in our study. We establish the end of the study in C_R and the time $\mathcal{O}_i$ is different for each individual. If ε ocurrs before C_R, we observe the potential time to failure F_i. We model the entry time for each individual as 0 for purposes of data analysis. Then, we rescale the variables. If we define $T_i = F_i - \mathcal{O}_i$ and $C_i = C_R - \mathcal{O}_i$, we have a censoring time C_i for each individual, albeit a fixed period of observation C_R. We observe (Y_i, δ_i) for i in $1, \ldots, n$:

$$Y_i = \min(T_i, C_i) \qquad \text{and} \qquad \delta_i = \begin{cases} 1 & \text{if } T_i \leq C_i, \\ 0 & \text{if } T_i > C_i \end{cases} \tag{1.9}$$

1.1.4.1.2 Type II of censoring We don't have a preset end for our study. When a number $r(r < n)$ of failures has ocurred, our study ends. In this censoring, the number of observed events is fixed. Then, the censoring time C is random.

Random: The failure and censoring times are treated as random variables with unknown distributions. The $T_1, \ldots, T_n$ failure times are i.i.d. with unknown distribution function F. The same case for the censoring times C_i, with a distribution function G. We observe the pairs (Y_i, δ_i) for i for 1 to n where $Y_i = \min(T_i, C_i)$, where C_i are the censoring times for each individual and

$$\delta_i = \begin{cases} 1 & \text{if } T_i \leq C_i, \\ 0 & \text{if } T_i > C_i \end{cases} \tag{1.10}$$

We need independence between T_i and C_i for assuming the random censoring.

1.1.4.2 The likelihood function

For building any likelihood function for the right-censored data[3], we must consider the type of data and its censoring, truncation and independence or not between T and the censored data. If that doesn't happen, then we need much more information about the behavior of the censoring times in our data.

The first function is the following, an individual censored at the right (for any type). We have a discrete random variable Y. Now, the contribution of our individual at our likelihood function depends of the observation and censoring times. We represent that as (Y, δ). We have two situations:

1. Not censored individual: $(Y, \delta) = (y, 1)$. The contribution to $\mathcal{L}$ is given:

$$\begin{split} P\{y, \delta = 1\} = P\{Y = y, T \leq C\} = P\{T = y, T \leq C\} = \\ P\{T = y, y \leq C\} = \boxed{P\{T = y\}P\{C \geq y\}} \end{split} \tag{1.11}$$

2. Censored individual: $(Y, \delta) = (y, 0)$. The contribution to $\mathcal{L}$ is given:

$$\begin{split} P\{y, \delta = 0\} = P\{Y = y, T > C\} = P\{C = y, T > C\} \Rightarrow \\ P\{C = y, T > y\} = P\{C = y\}P\{T > y\} \end{split} \tag{1.12}$$

We suppose that T is independent of C, so we calculate the contribution of an individual with observed data (y, δ) by $P\{y, \delta\} = (P\{T = y\}P\{y \leq C\})^{\delta}(P\{C = y\}P\{T >$

[3] For a detailed explanation of the likelihood function, read Rohde (2014)

$y\}^{1-\delta}$. For the continuous case, the contribution of an individual (y, δ) is $P\{y, \delta\} = (f(y)G(y))^{\delta}(g(y)S(y))^{1-\delta}$.

A sample of individuals censored at right, is written as a product for each individual and its contribution, with a sample $(y_1, \delta_1), \ldots, (y_n, \delta_n)$.

$$\prod_{i=1}^{n} P\{y_i, \delta_i\} = \prod_{i=1}^{n} \{[f(y_i)]^{\delta_i}[S(y_i)]^{1-\delta_i}\} \prod_{i=1}^{n} \{G(y_i)^{\delta_i}[g(y_i)]^{1-\delta_i}\} \tag{1.13}$$

If C has a non-informative censoring to T and both variables are independent, then the estimation for the distribution of T via likelihood function will not be dependent to C and, then, we can work with one of the following likelihood functions:

$$\prod_{i=1}^{n} \{[f(y_i)]^{\delta_i}[S(y_i)]^{1-\delta_i}\} \tag{1.14}$$

Or because $f(y) = \lambda(y)S(y)$, that:

$$\prod_{i=1}^{n} \{[\lambda(y_i)]^{\delta_i}[S(y_i)]\} \tag{1.15}$$

1.1.5 Non-parametric methods

With complete datasets, we can estimate the survival function: $\hat{S}(t) = 1 - \hat{F}(t)$, where $\hat{F}$ is the empirical distribution function (EDF). But if we have, at least, one censored observation, then the EDF will not be a consistent estimator. Then this survival function will be useless. We must use other ways, but in first place, some notation:

- The r different times: $Y_{(1)} < \cdots < Y_{(i)} < \cdots < Y_{(r)}$
- Number of individuals at risk before $Y_{(i)}$: n_i
- Number of individuals that fail at moment $Y_{(i)}$: d_i

1.1.5.1 Kaplan-Meier estimator

The most known estimator is Kaplan-Meier estimator ($\hat{S}_{KM}$). There, we compute the survival probabilities with a product limit formula, denoted as:

$$\hat{S}_{KM}(t) = \begin{cases} 1 \quad \text{if } t < Y_{(1)} \\ \prod_{i:Y_{(i)} \leq t} \left(1 - \frac{d_i}{n_i}\right) \qquad \text{if } t \geq Y_{(1)} \end{cases} \tag{1.16}$$

If the last observation is censored, then the function isn't well defined, because the survival function never takes the value 0 and $\lim_{t\to\infty} \hat{S}_{KM}(\infty) > 0$. How to deal with that? We have two suggestions, from Efron and Gill, with the same behavior, asymptotically speaking, but the latter has a better behaviour for small samples. We'll show them below:

- **Efron:** Redefine $\hat{S}_{KM}(t) = 0$ for all $t \geq Y_{(n)}$
- **Gill:** Maintain $\hat{S}_{KM}(t) = \hat{S}_{KM}(Y_{(n)})$ when $\delta_{(n)} = 0$ for all $t > Y_{(n)}$

This estimator is a step function, broken by observed event times and a variable size, depending on the number of events at each time $Y_{(i)}$ and the pattern of the censored observations prior to $Y_{(i)}$.

If the data doesn't have censored observations, then this estimator is reduced to the empirical survival function, $\hat{S}(t) = 1 - \hat{F}(t)$

For the cumulative hazard case, we must remind the relationship between the survival and cumulative hazard functions. With that, we can compute an estimator of $\Lambda(t)$, based on the Kaplan-Meier estimator:

$$\hat{\Lambda}_{KM}(t) = -\ln \hat{S}_{KM}(t) \tag{1.17}$$

1.1.5.2 Nelson-Aalen estimator

For the cumulative hazard function, we have a non-parametric estimator, called Nelson-Aalen estimator, with a better performance for small sample sizes, denoted as:

$$\hat{\Lambda}_{NA}(t) = \begin{cases} 0 & \text{if } t < Y_{(1)} \\ \sum\limits_{i:Y_{(i)} \leq t} \frac{d_i}{n_i} & \text{if } t \geq Y_{(1)} \end{cases} \tag{1.18}$$

The estimator for the survival function is $\hat{S}_{NA}(t) = e^{-\hat{\Lambda}_{NA}(t)}$

It's useful for:

1. Select between parametric models for the failure time.
2. Crude estimates of the hazard rate $\lambda(t)$, the slope of the Nelson-Aalen estimator.

In the case of simultaneous multiple deaths, we can modified the Kaplan-Meier and Nelson-Aalen estimators as distinct, even if we don't know that distinction.

Kaplan-Meier

$$\hat{S}_{KM}(t) = \begin{cases} 1 & \text{if } t < Y_{(1)} \\ \prod\limits_{i:Y_{(i)} \leq t} \prod\limits_{k=0}^{d_i-1} \left(1 - \frac{1}{n_i-k}\right) & \text{if } t \geq Y_{(1)} \end{cases} \tag{1.19}$$

Nelson-Aalen

$$\hat{\lambda}_{NA}(t) = \begin{cases} 0 & \text{if } t < Y_{(1)} \\ \sum\limits_{i:Y_{(i)} \leq t} \sum\limits_{k=0}^{d_i-1} \frac{1}{n_i-k} & \text{if } t \geq Y_{(1)} \end{cases} \tag{1.20}$$

1.1.6 Kaplan-Meier Survival Curves

With our survivor function, $S(t)$, we draw curves. In many cases, these survivor curves will be step functions (an ideal curve will be smooth), as the curve showed below:

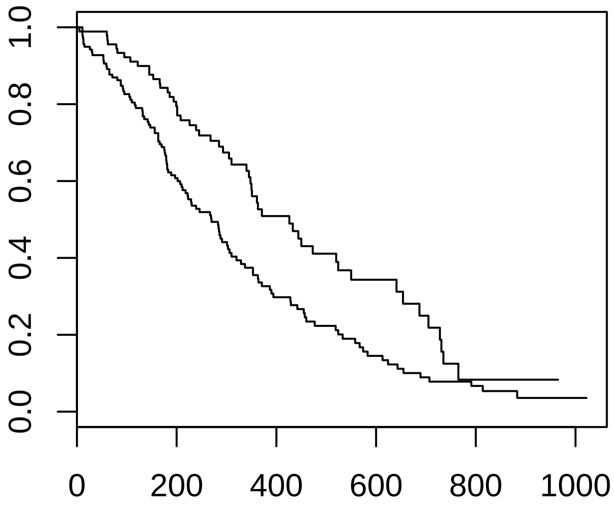


Figure 1.1: An example of curve estimated by the Kaplan-Meier method, with the data of lung cancer from the R package `survival`

The general formula for a KM survival probability at failure time $t_{(j)}$ is $\hat{S}(t_{(j-1)}) \times \hat{P}r(T > t_{(j)} | T \geq t_{(j)})$.

1.1.7 Parametric models

When we make our survival analyses, in a certain stage we'll use our data for adjusting and approximating a model. In that sense, the parametric models are useful.

A general idea about distributions in survival analysis is the following: the distributions can be adapted to our hazard function and we have the common parameters: α and γ as the shape parameters, μ as the location parameter and β as the scale parameter. With each distribution, our survival and hazard functions varies. Some of the most important distributions for our work are: Weibull, Gumbel, Normal, Log-Normal, Logistic, Log-logistic, Four-Parameter Beta Distribution, Exponential Power Distribution and Exponentiated Weibull distribution, with its hazard rates.

Then, when we choose any parametric model for our survival analysis, we need some type of fit (assess goodness of fit). We have, basically, two types of techniques, the graphical, valid for non-censored and censored data (although in the latter case, we must readjust sometimes our plot, because we have less valid points). And the analytical, that can be regarded as measures of proximity between the empirical distribution and the hypothesised. For the first, the techniques are probability plots (P-P and Q-Q), stablised probability. For the second, Kolmogorov-Smirnov, Cramér-von Mises statistics, Anderson-Darling statistics (weighted-L^2 distance).

1.1.7.1 Distributions

For any distribution, we start with this data, with length n:

Y	δ
Y_1	δ_1
Y_2	δ_2
$\dots$	$\dots$
Y_n	δ_n

Table 1.2: A basic structure of data for survival analysis

And the random variable T_i (event time). $T_1, T_2, \dots, T_n$ are i.i.d. random variables with unknown cumulative distribution F. For that, we need some fitting for our chosen distribution with that hypothesis:

$$H_0 : F(\cdot) = F_0(\cdot; \theta) \tag{1.21}$$

$F(\cdot)$ is the unknown distribution and $F_0(\cdot, \theta)$ is our desired theoretical distribution. With that, we fit our data and we assess our goodness of fit. If we don't specify parameters, we might use the maximum likelihood estimate of θ, $\hat{\theta}$. If the data are censored, it's a good idea to use Kaplan-Meier or Nelson-Aalen estimators ($\hat{F}$).

In first place, we have probability plots, useful for evaluate our chosen distribution at a glimpse. The most known are P-P and Q-Q plots, with its modifications and Stablised Probability Plot (which transforms the axes to approximately get the same variance in each plotted point) and Empirically Rescaled plot (useful for data heavily censored). In the P-P plot, we plot the theoretical cumulative distribution against the estimated cumulative distribution from data (can be estimated using Kaplan-Meier or Nelson-Aalen). The graph can take numerous shapes. If we have a straight line from (0, 0) to (1, 1), then the data follow the theoretical distribution. If we have, for example, a S-shaped graph, then our distribution doesn't adjust well to the data.

For the Q-Q plot, we use theoretical and estimated quartiles ($\hat{F}_0^{-1}(\hat{F}(t))$ against t). It's very similar to P-P.

In last place, we have the SP plot and ER plot. For the first, our purpose is to stabilise the variance of the plotted points, because that enhances its interpretability.

For the latter, this plot is useful for randomly right-censored cases. In the ER, we plot $\hat{F}_u(\hat{F}_0^{-1}(\hat{F}(y_i)))$ versus the empirical cumulative distribution of the points corresponding to the uncensored observations, $\hat{F}_u(y_i)$. The visual appearence, in overall, is less sensitive to the effects of different censoring patterns.

The **cumulative hazard plot**, similar to the plots showed above, is used to assess the goodness of fit visually from a distribution. There, we transform our CHF (cumulative hazard function) Λ into something linear in t or $\log(t)$. For each distribution, that changes, but we have a common idea: 1) data for computing the Nelson-Aalen estimator ($\hat{\Lambda}$) of the cumulative hazard function and the MLE of the parameters of our theoretical distribution; 2) look for a transformation $A(\cdot)$, relative to the cumulative hazard function of the theoretical distribution; 3) $A(\hat{\Lambda})$ will be linear in natural or logarithmic scale. Sometimes, we plot $\Lambda^{-1}(\Lambda(t))$ versus t (probability plot).

1.1.7.2 State of the art of the goodness-of-fit models

Sometimes, when we deal with parametric models, we need to check how our chosen model fits with the data set (and its degree of accuracy). We assess, then, the goodness of fit in two ways, graphically and analytically.

1.1.7.2.1 Graphical goodness of fit For that, we have the cumulative hazard plots, the probability plots P-P and Q-Q plots and the stabilised probability plot, a modification of the P-P plot with the purpose of stabilise the variance of the plotted points, introduced by Michael (1983). These methods are valid for both types of data, right-censored and non-censored, although for the latter, the plots mentioned above have points not evenly spread. But we can fix that with the empirically rescaled plot (Waller, 1992).

1.1.7.2.2 Statistics for the goodness of fit Most goodness of fit statistics are measures of proximity between two distributions, the empirical and hypothesized, for example the Kolmogorov-Smirnov statistic, based on the supremum distance, the Crámer-von Mises statistic and the Anderson-Darling statistics, using a weighted-L^2 distance. These statistics are useful for right-censored data, but with issues as an unknown asymptotic distribution. We have few goodness of fit procedures for the time-continuous data with censoring, and most of them are modificated versions of statistics for the non-censored data, as a modification of Kolmogorov-Smirnov statistic for the type I censored data[4]. For a overall view and a manual for many leading methods of testing fit studied (centered, overall, in the three major distributions, normal, exponential and uniform) plus the handling of censoring data, we have Ralph B. D'Agostino (1986).

1.2 Goals of the master's thesis

In this master thesis, our main purpose is to continue the work of Febrer Galvany (2015) and Besalú i Mayol (2016) in the goodness of fit for right-censored data, a neccesary matter in many survival analyses, because if our data doesn't fit well with our chosen distribution, then our survival analysis can fail. For that, we build a R package, called `GofCens`, with two functions, `prob.plot` and `cum.haz.plot`, and two datasets, which will be updated in a foreseeable future (squashing bugs, adding features, improving performance, adding datasets, upload it to CRAN and adding functions). We have four chapters and two appendices.

In the Chapter 1, we explored some matters of the survival analysis, the distributions and its density, survival, cumulative hazard and hazard functions, the censored data, with special incidence in the right-censored data, the non-parametric methods as the well known Kaplan-Meier estimator and the parametric methods, one of the most important matters here, because these contains the goodness of fit in both ways, graphical and quantitative.

In the Chapter 2, we'll review some of the parametrical techniques, quantitative and graphical goodness of fit. The first will be reviewed briefly, because we won't use techniques such Kolmogorov-Smirnov in our package by the moment. By contrast, we'll talk extensively about the graphical goodness of fit, basically an application of four types of functions for the distributions, density, survival, cumulative hazard and hazard functions, because the two functions of our packages is about analysis. For last, we'll review the original R functions by Anna Febrer. Plus, we'll talk about the R state of art in survival analysis, incluiding graphical techniques.

In the Chapter 3, we'll talk about the R package `GofCens`: its archives, functions and dataset. Also, we'll execute our functions for first time with dataset found in the `survival` package (Therneau, 2015).

For the Chapter 4, we'll test our functions with the two datasets of the package, `gbcs` and `whas500` and we'll comment about that. This part could be useful for further corrections of our package.

We conclude this thesis with the conclussions and the appendices about the R code

[4] See Dufour (1978) and Baar (1973)

of the functions explained in the Chapter 3 and two tables of our distributions and its functions.

Chapter 2

QUANTITATIVE AND GRAPHICAL TECHNIQUES OF GOODNESS OF FIT AND AN OVERVIEW OF R SOFTWARE

Following the brief explanation of the chapter 1, in this chapter we'll explain the analytical and graphical techniques of the goodness-of-fit tests. Plus, we'll present a global vision of the R software for data analysis and we'll explain and test the two functions created by Febrer Galvany (2015), `prob.plots` and `CumHazPlot` for a glimpse of the graphical tools of goodness of fit in R.

2.1 Quantitative goodness of fit techniques

For this sort of techniques, we have a distinction between complete data (when for some individuals it is only known that their failure time exceeds a certain censoring timee) and right-censored cases (when, at least, an individual don't have any failure time). With that in mind, we can proceed to the basic notation of the goodness of fit tests.

We have the data (Y_i, δ_i), where $Y_i = \min(T_i, C_i)$[1] and (δ_i) is the censoring indicator, a binary variable with the following values, 0, equal to right censoring, and 1, equal to $\{T_i \leq C_i\}$.

In second place, we'll show the hypothesis for assessing the goodness of fit, in where an inappropiate theoretical distribution F_0 can lead to incorrect results:

$$H_0 : F(\cdot) = F_0(\cdot; \theta) \tag{2.1}$$

There, $F(\cdot)$ is the unknown distribution of the event times and $F_0(\cdot, \theta)$ is our desired theoretical distribution (for adjusting to the data and assess its goodness of fit). We can completely specify F_0 or let F_0 undetermined except a finite dimensional parameter θ. Then, we'll calculate the maximum likelihood estimate of θ, $\hat{\theta}$. Then:

[1] For each individual, we have two parts: 1) T_i is a random variable which describe the event times that are independent and identically distributed with unknown cumulative distribution F and C_i are the censoring times for each individual

$$\hat{F}_0(t) = F_0(t; \hat{\theta}) \tag{2.2}$$

If we have uncensored data, we can estimate F by the empirical cumulative distribution function. But if our data have right-censored observations, then we'll estimate F by the Kaplan-Meier or Nelson-Aalen estimators. The result is denoted by $\hat{F}$.

Now, we can explain two of the following techniques, very well known in survival analysis, as a brief introduction, although we don't detail too much, because the main protagonist of this report will be the graphical techniques and its uses in R.

2.1.1 Kolmogorov-Smirnov goodness of fit test

That's the most known and used analytical method to test goodness-of-fit with complete data. Then, the Kolmogorov-Smirnov statistic D_n for any given cumulative distribution function F_0 is the following:

$$D_n = \sup_t |F_0(t) - \hat{F}_n(t)| \tag{2.3}$$

F_n is the empirical distribution of the data, with n as the data sample size. If the sample comes from distribution F_0, by the Glivenko-Cantelli theorem, D_n converges to 0 when n goes to infinity. This result was carefully studied by Kolmogorov (1933), leading him to find the asymptotic distribution of D_n under the null hypothesis, the Kolmogorov distribution. N. Smirnov (1948) provided us with a table for the goodness-of-fit, after that its study of the corresponding one-sided bounds (N. V. Smirnov, 1939).

The Kolmogorov distribution is the following random variable, where $B(t)$ is the Brownian bridge:

$$K = \sup_{t \in [0,1]} |B(t)| \tag{2.4}$$

The cumulative distribution is:

$$P(K \leq k) = 1 - 2\sum_{j=1}^{\infty}(-1)^{j-1}e^{-2j^2k^2} \tag{2.5}$$

If the sample come from the hypothesised distribution F_0, Kolmogorov showed that:

$$\sqrt{n}D_n \xrightarrow[n\to\infty]{\mathcal{D}} \sup_t |B(F_0(t))| \tag{2.6}$$

2.1.2 Grané Goodness of Fit test for Type I and Type II Right-censored Data

Recently, a goodness of fit test, proposed by Grané (2012) has been published and it's adequate for data with particular patterns of right, left and double censoring. We will refer to this as the Grané test. More recently, in a new paper (Strzalkowska-Kominiak & Grané, 2014), we have a new test statistic for the general right-censoring, with an unknown censoring distribution.

We have a set of independent and identically distributed random variables, $T_1, \ldots, T_n$, with cumulative distribution function F and the order statistics $T_{(1)} < \cdots < T_{(n)}$. With that test, we can assess goodness of fit when the observations are less than C_L and (or) greater than C_R are censored. For this report, we'll just consider the right-censored data, so we consider these time event values greater than C_R. This test is suitable for Type I and II censoring, but not for random-censored variables. We'll test our null hypothesis for every t:

$$H_0 : F(t) = F_0(t) \tag{2.7}$$

F_0 is a completely specific cumulative distribution function, but we can only use it for testing $t_{(1)}, \ldots, t_{(r)}$, where r is the number of observed events.

$$H_0 : F(t_{(i)}) = F_0(t_{(i)}), \qquad \text{for } i = 1, \ldots, r \tag{2.8}$$

If T is a random variable with cumulative distribution function F, $F(T)$ follows an uniform distribution in [0, 1], indeed:

$$P(F(T) < t) = P(T < F^{-1}(t)) = F(F^{-1}(t)) = t \tag{2.9}$$

With that, with T_i for $i = 1, \ldots, n$ following a distribution with cdf F, $F(T_i)$ follows a Uniform in [0, 1]. If the null hypothesis (2.7) is true, $F_0(T_i)$ will also follow an Uniform in [0, 1].

2.2 Graphical techniques of goodness-of-fit

In that subsection, we'll talk about the visual side of the goodness of fit tests, the plotting, the contrast between the theoretical and empirical distributions. At a first glance, the graphical techniques are very useful (and simple) exploratory tools for understanding the distribution of the data and testing the following null hypothesis:

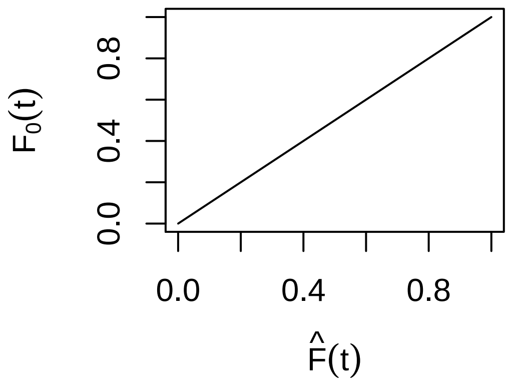
$$H_0 : F(\cdot) = F_0(\cdot|\theta) \tag{2.10}$$

2.2.1 Probability plots

With these tools, we can check rapidly if our chosen distribution is appropiate or not. We have four of them, P-P, Q-Q, stabilised probability and empirically rescaled.

2.2.1.1 P-P plot

Called the Probability-Probability plot, this plot consists of plotting $F_0(t)$, theoretical cumulative distribution with its parameters estimated by maximum likelihood if are unknown, against $\hat{F}(t)$, the estimated cumulative distribution function derived from data. For the right censored cases, we estimate the empirical distribution with Kaplan-Meier or Nelson-Aalen techniques. The graph must be a straight line from $(0, 0)$ to $(1, 1)$, depicted below, if the data really follows the theoretical distribution:



If the data doesn't fit, then we have a S-shape, as shown below:

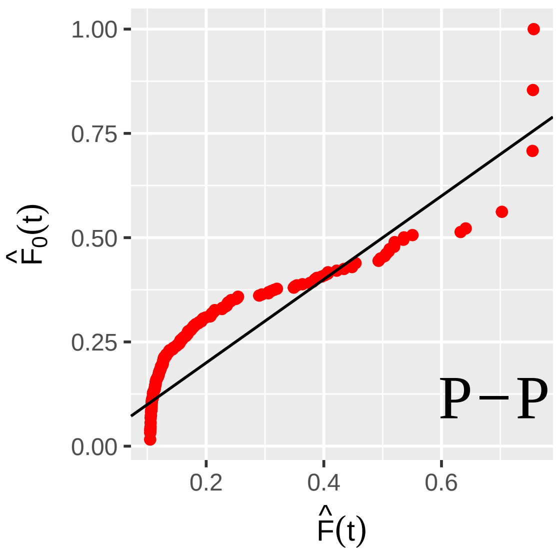


Figure 2.1: An example of a unfitted probability-probability plot for the normal distribution

Then, we can discard that distribution. A reminder: the estimates of the distribution F_0 only can be changed with the uncensored observations, so the points plotted in any P-P plot will be equal to y_i when $\delta_i = 1$. If the data are uncensored or have a censoring of Type I or II, then all the plotted points are evenly distributed in a line from $(0,0)$ to $(\frac{r}{n}, \frac{r}{n})$, where r is the number of observed events and n, the sample size. If the proportion of random censored data are high, this distribution is not longer the right one. Then, we can observe a situation where some points are more concentrated near to $(0,0)$ than $(1,1)$

2.2.1.2 Q-Q plot

Known as Quartile-Quartile and similar to P-P plot, there we'll plot the theoretical quartiles, $\hat{F}_0^{-1}(\hat{F}(t))$ against the estimated quartiles, t. If the theoretical distribution F_0 fits well the data, the plot should be close to a straight line; otherwise, we'll get a curved plot, as depicted below. This means that the data doesn't follow very well F_0.

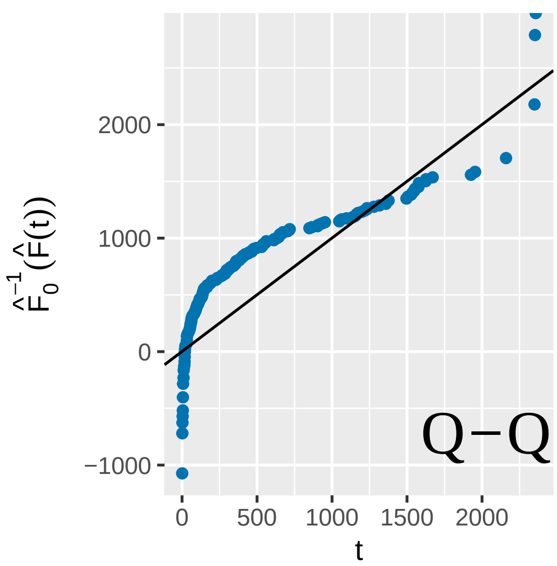


Figure 2.2: An example of unfitted Q-Q plot for the normal distribution.

But we have a caveat: the plotted points may be not evenly spread even if our data are complete (in a certain area of our graph, we could have most of the points), leading to potential biased conclusions.

2.2.1.3 Stabilised Probability Plot

Michael (1983) introduced the stabilised probability plot as a transformation of the P-P plot with the purpose of stabilise the variance of the plotted points, which are approximately equal and, hence, that enhances its interpretability. Originally, in the P-P and Q-Q plots, the variance of the points differ.

If $F_0 = F$ and its parameters are known, $\hat{F}_0(y_i)$ can be regarded as an uniform order statistic. If these parameters are unknown but efficiently estimated, then that's true asympotically. We can use an arcsin transformation for stabilise the variance of $\hat{F}_0(y_i)$ across a uniform order statistic. For example, if we have $S = \frac{2}{\pi}\text{arcsin}(\sqrt{U})$ where $U \sim Uniform[0,1]$, the probability density function of S is given, for $0 \leq s \leq 1$ by:

$$f(s) = \frac{\pi}{2}\sin(\pi s) \tag{2.11}$$

That's the sine distribution, with an interesting property, same asymptotic variance for its order statistics, equal to $1/\pi^2$, independent of its position. Then, we define the stabilised probability plot as:

$$\frac{2}{\pi}\text{arcsin}(\sqrt{\hat{F}_0(y_i)}) \tag{2.12}$$

If F_0 fits the data, we have a line from (0, 0) to (1, 1). Otherwise, the points will be plotted as a S-shaped figure, such the plot depicted below:

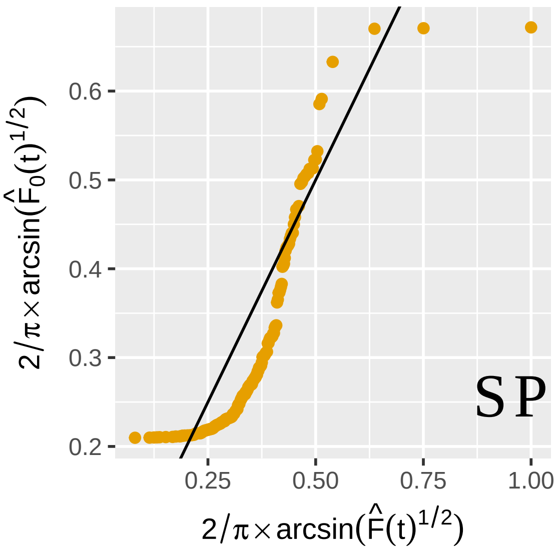


Figure 2.3: An example of fitted SP plot for the normal distribution, with some outliers.

2.2.1.4 Empirically Rescaled plot

If we have uncensored data or have a Type I or II censoring for our data, we can use a P-P plot, better than the Q-Q plot, due to some issues, an uneven spread for all its points, fixed with an uniform horizontal spacing between points. But if we have some random right-censored data, the P-P plot isn't so useful because we have a non-uniform size for the empirical estimates of the probability function and its jumps

Due to that, Walter and Turnbull (Waller, 1992) proposed the empirically rescaled plot, in where we plot $\hat{F}_u(\hat{F}_0^{-1}(\hat{F}(y_i)))$ versus $\hat{F}_u(y_i)$:

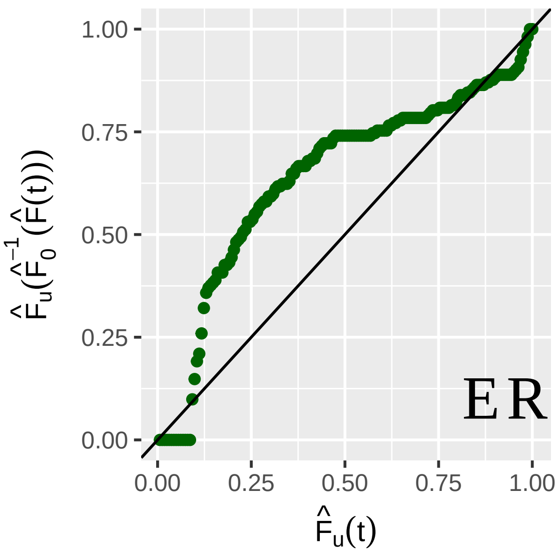


Figure 2.4: An example of unfitted ER plot for the normal distribution, with the real data points above the line.

$\hat{F}_u$ is the empirical cumulative distribution function of the points corresponding the uncensored observations. If the theoretical distribution fit the data, we have a straight line from $(0,0)$ to $(1,1)$. Otherwise, we'll have a S-shape plot. For last, among all the probabilistic plots, its appearence is the less sensitive to the different censoring patterns.

2.2.2 Cumulative Hazard plots

For these plots, we transform the cumulative hazard function Λ in somewhat linear in t or $\log(t)$. This transformation varies for each distribution, with the same premise:

1. Computing the Nelson-Aalen estimator $\hat{\Lambda}$ from data and the maximum likelihood of the parameters of our chosen distribution

2. For the cumulative hazard function of our chosen distribution, we seek a transformation $A(\cdot)$ with the purpose of get linearity in $A(\hat{\Lambda})$.

In Febrer Galvany (2015), in the chapter 2, we have a table with the expressions for the cumulative hazard plot for each distribution. We show an example below, in where we measure the adequateness for each distribution.

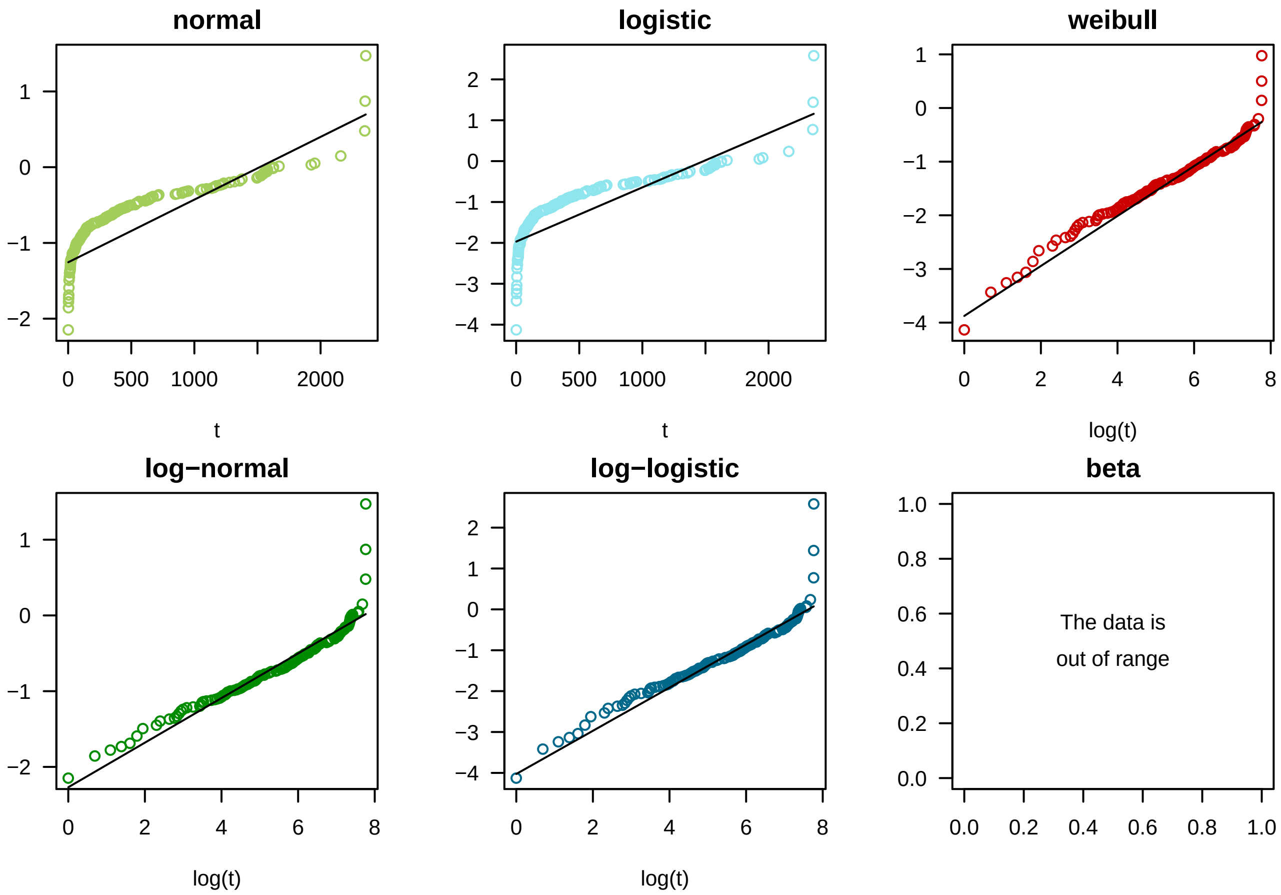


Figure 2.5: A grid of cumulative hazard plots for six distributions. The three others are discarded by errors in maximum likelihood estimations.

Sometimes, the cumulative hazard plot can be considered a probability plot between $\Lambda^{-1}(\Lambda(t))$ and t. With that and an appropiate distribution, the cummulative hazard plots will draw a straight line between (0, 0) and (1, 1). Otherwise, we'll have the same straight line but with a different scale, slope and intercept.

2.3 An overview of R software for graphical tests and two examples of functions, ProbPlots and CumHazPlot

2.3.1 R software

In first place, in the R software (R Core Team, 2017), we have two views for the survival analysis:

- https://cran.r-project.org/web/views/Survival.html
- https://cran.r-project.org/web/views/Distributions.html

For the first view, we have many packages related with the survival analysis in overall, some of them already incorporated with R, as the survival package for computing the Kaplan-Meier estimator. For the censored data, muhaz can be very useful (hazard function), as the package ICE (for interval censored data). But for many analysis, we must to consider the following facts: the time variable, in many cases, is a numeric variable. For example, the number of days for a treatment (written as c(234, 221, 245,...) in R). The censoring variable is a binary variable with two values, 0 and 1.

For the second view, we cover the distributions in overall, important for all sorts of statistical techniques, but for the survival analysis we just interested in the following distributions, Weibull, Gumbel, Normal, Log-Normal, Logistic, Log-logistic, Four-Parameter Beta Distribution, Exponential Power Distribution and Exponentiated Weibull distribution, with its parameters, shape or location, depending of the distribution, and scale. Mainly, for many of these distributions, we have the base R functions for the ditributions, d, p, q, and r. For some more specialized distributions, as the Gumbel, we need extra packages such ActuDistns.

At last, for fitting distributions, neccesary under many statistical techniques, as the goodness of fit, we have the very useful package fitdistrplus (Delignette-Muller & Dutang, 2015a), which saves us work because, for example, the fitdistr function will estimate the parameters of our desired distribution with maximum likelihood using the function mledist, although we can choose other methods such the moment matching estimation.

In second place, we'll briefly comment how to make graphics (Murrell, 2012) in R, fundamental for some aspects in the survival analysis. For that, we must differentiate between the classic plotting system, inside one of the base R packages, graphics, and the graphical system based in the grammar of graphics (Wilkinson, 2005), ggplot2. For the first case, we have a system in what we draw in a empty plot, created with the command plot(), like a sheet of paper. With commands as lines or abline, we draw new layers for our plot. We can't delete one of these layers if we don't create a new plot.

Based in the grammar of graphics, which, ggplot2, by Hadley Wickham, provides us another approach in plotting, separating any graph in basic components and thinking beyond concepts as scatterplot (Wickham, 2010), such the y-axis and x-axis or the aesthetics. Basically, we think in terms of layers. In first place, in ‘ggplot2‘, we choose the data and we make a aesthetics. The first argument for any plot can be the following:

```
ggplot(data, mapping=aes(x=data$x, y=data$y))
```

Later, we can draw the shape of our plot adding with layer. For a scatterplot, we can write the following:

```
ggplot(data, mapping=aes(x=data$x, y=data$y)) + geom_point()
```

And we can aggregate more layers. That allows us a better level of personalization and, in overall, better aesthetics with less effort. Even, we can save the plot in an independent object.

```
my_graph <- ggplot(data, mapping=aes(x=data$x, y=data$y))
my_graph + geom_line()
```

With the R landscape about survival analysis in mind and a brief description of the two more used plotting techniques, we'll show two functions (Febrer Galvany, 2015) that do a nice job of checking in a glimpse the goodness-of-fit of our data.

2.3.2 prob.plots

The `prob.plots` function have its foundations in the idea of plots for checking the goodness of fit, with four main types of plots, the ones presented in the subsection 2.2.1: stabilised probability, empirically rescaled, P-P and Q-Q. We have, there, two realms: the theoretical and the real distribution. Our goal is to check the fitting of the theoretical distribution.

For that, we test the following hypothesis, where F is the real cumulative distribution of our data and F_0, our selected distribution, whose parameters can be calculated for maximum likelihood with the function `fitdistcens` (Delignette-Muller & Dutang, 2015b) or we can specify our desired parameters:

$$H_0 : F = F_0(\cdot;\theta)$$

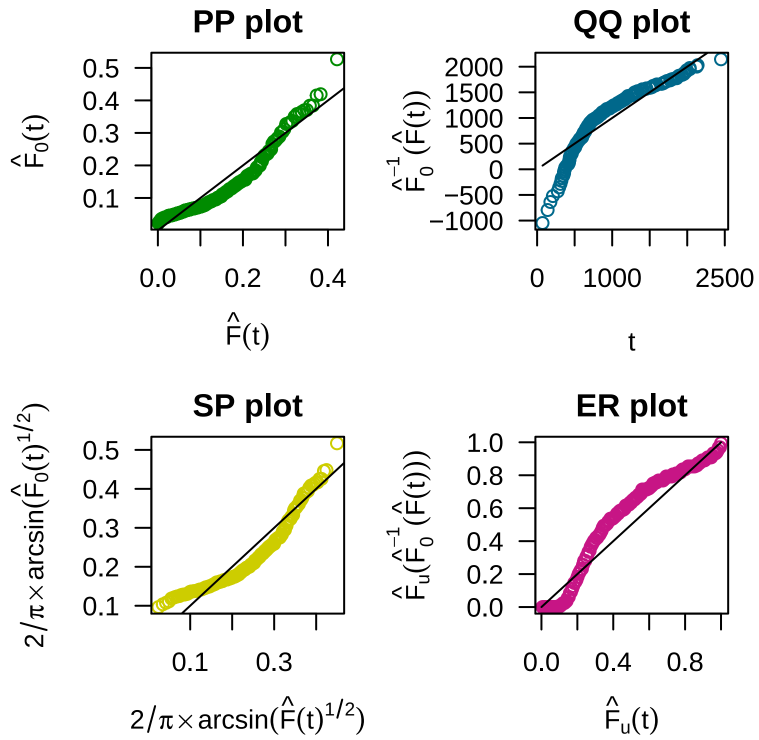
Then, with that, we have the following function, with seven arguments in total.

```
prob.plots <- function(time,
                       cens,
                       distribution,
                       beta.limits=c(0,1),
                       plots = c("PP","QQ","SP","ER"),
                       colour = c("green4","deepskyblue4",
                                  "yellow3","mediumvioletred"),
                       parameters = list(shape = NULL, shape2 = NULL,
                                         location = NULL, scale = NULL))
```

We list the explanation for each argument:

- `time`: A variable of times until the event of interest.
- `cens`: A binary variable for the censored observations.
- `distribution`: Our chosen distribution from nine: normal, Weibull, Gumbel, exponential, log-normal, logistic, logg-logistic, beta, exponential power and exponentiated Weibull.
- `beta.limits`: If we choose the beta distribution, we have to determine its bounds, lower and upper. By default, is `c(0, 1)`.
- `plots`: A vector with our plots, `PP` (P-P plot), `QQ` (Q-Q plot), `SP` (Stabilised Probability Plot) and `ER`, Empirically Rescaled Plot. By default, we display the four plots.
- `colour`: For default, is a vector of four colours (one for each plot)
- `parameters`: With that, we can specify by hand the parameters of the chosen theoretical distribution

If we want to execute the function, both variables, `time` and `cens` must be numeric. For default, our output will be an figure with the four probability plots, as the chunk shown below, with a dataset, from a German Breast Cancer study (David W. Hosmer, 2008), of N=686, testing it against an theoretical normal distribution:



```
## $distr
## [1] "Normal"
##
## $location
## [1] 2373.418
##
## $scale
## [1] 1151.843
```

In the output, we can see that the data doesn’t adjust very well to the normal distribution and, below, we’ll see the parameters of our distribution that we can extract as a single value, as `output$location` or the scale.

2.3.3 CumHazPlot

In this function, our purpose is to check the fitting of our data for different distributions:

```
CumHazPlot<-function(time, cens,
                     distributions=c("gumbel","norm","logis",
                                     "weibull","lnorm","loglogis"),
                     beta.limits=c(0,1),
                     colour = c("orangered","darkolivegreen3","cadetblue2",
                                "red3","green4","deepskyblue4",
                                "hotpink","yellow3","mediumvioletred")
```

We have five arguments:

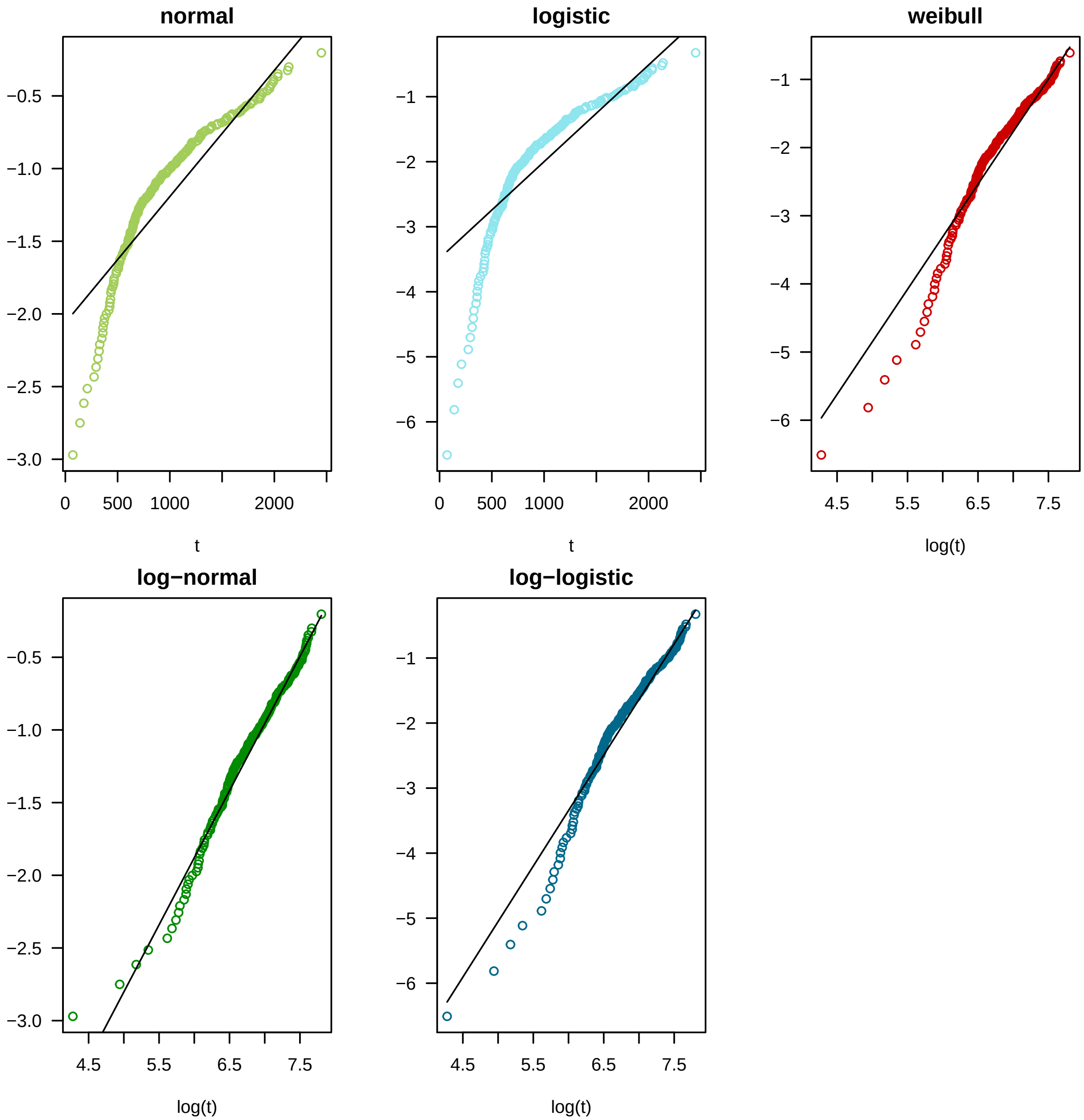
- `time`: A variable of times until the event of interest.
- `cens`: A binary variable for the censored observations.
- `distribution`: Our vector of distributions from nine: normal, Weibull, Gumbel, exponential, log-normal, logistic, log-logistic, beta, exponential power and exponentiated Weibull. By default, we have six distributions, although we can use ‘all‘ for considering the nine distributions
- `beta.limits`: If we choose the beta distribution, we have to determine its bounds, lower and upper. By default, is `c(0, 1)`
- `colour`: For default, is a vector of four colours (one for each plot)

For the output, we have, for default, six cumulative hazard plots corresponding to the Gumbel, Normal, Logistic, Weibull, Log-normal and Log-logistic distributions. We need just two arguments, ‘time‘, the times until the event, and ‘cens‘, the censoring indicator for every failure time:

```
CumHazPlot(time, cens)
```

Now, we'll execute the function with the same dataset about breast cancer:

```
## <simpleError in optim(par = vstart, fn = fnobjcens, fix.arg = fix.arg, gr = gradient,
```



We find, that, the best distribution are the log-normal, with a very well fitted curve. Plus, we see an error message for the beta distribution, the output of the MLE, and the location or shape and scale for every disstribution in numerical terms, parameters that can be calculated by hand or via maximum likelihood. We can extract it with the following command, `functionoutput\$distribution`

And with that, we conclude the presentation of the functions and the second chapter. In the following chapter, we'll present an improvement, a R package called *GofCens*.

Chapter 3

THE R PACKAGE GOFCENS

In this chapter, we'll write, mainly, a brief description of the creation of the R package and a documentation with every object of our package, the functions and the datasets, shown below.

```
## [1] "cum.haz.plot" "gbcs"         "prob.plot"    "whas500"
```

3.1 Overall vision

3.1.1 Description

In the context of the development of graphical techniques for the goodness of fit in survival analysis, as shown in the first two chapters of this report, this package is a logical step after the functions of Febrer Galvany (2015), shown in the Chapter 2. Also, for a proper package, I made some modifications and I added documentation and two datasets, which I'll show later, with explanations for each variable. These modifications have the purpose of making the functions more fast and intuitive for the final user, although we must pay a price there: the loss of options in our function for the personalization.

3.1.2 Some files of the package

In this subsection, we'll talk about essential files of our package.

3.1.2.1 DESCRIPTION

The most important file of the package, contains the basic information for our package: version number, authors, lazy loading, imports, title, maintainer, license type, a brief description of our package, encoding and many other options. We'll put the description file of our package for a glimpse:

```
packageDescription("GofCens")

## Package: GofCens
## Type: Package
## Title: Graphical tests for the goodness-of-fit of right-censored
##        data in survival analysis
## Version: 0.5
## Authors@R: c(person("Daniel", "García Carrasco", role=c("aut",
##        "cre")), person("Guadalupe", "González Melis",
##        role=c("aut")), person("Klaus", "Langohr", role=c("aut")),
##        person("Mireia", "Besalú", role=c("aut")))
## Author: Daniel García Carrasco [aut, cre], Guadalupe González
##        Melis [aut], Klaus Langohr [aut], Mireia Besalú [aut]
## Maintainer: Daniel García Carrasco
##        <daniel.garcia.carrasco@estudiant.upc.edu>
## Description: This package is useful for making graphical
##        diagnostics of the goodness of fit for right-censored data
##        in two forms, the probability plots and cumulative hazard
##        plots
## Imports: ggplot2, gridExtra, survival, grid, fitdistrplus,
##        ActuDistns
## License: GPL
## Date/Publication: 2017-05-15 15:57:48
## Encoding: UTF-8
## LazyData: true
## LazyLoad: Yes
## Built: R 3.4.0; ; 2017-06-05 16:18:59 UTC; unix
##
## -- File: /home/daniel/R/x86_64-pc-linux-gnu-library/3.4/GofCens/Meta/package.rds
```

3.1.2.2 NAMESPACE

One of the most important parts of building a package is specifing the namespace, albeit can be complex and confusing (Wickham, 2015), due to its conceptions, spaces for names. With a namespace, we can make our package self-contained thanks to defining the imports, when we define how a function in one package finds a function in another, and exports, which helps us to avoid conflicts with other packages. For this package, we mantain a simple namespace file, with importing all the functions from some needed packages (although in a future we can specify some functions for optimization) and exporting all the functions that match a pattern.

```
exportPattern("^[[:alpha:]]+")
import(survival)
import(fitdistrplus)
import(grid)
import(ggplot2)
import(gridExtra)
import(ActuDistns)
```

3.1.2.3 CITATION

Inside the folder `./inst`, this file swows how to cite our package:

```
citation("GofCens")


##
## To cite GofCens in publications, please use:
##
##   Daniel García et al. Goodness-of-fit R package for
##   Right-censored data.
##
## A BibTeX entry for LaTeX users is
##
##   @Manual{,
##     author = {Daniel García Carrasco & et al.},
##     title = {Goodness-of-fit R package for Right-censored data},
##     year = {2017},
##   }
```

The output is in BibTeX, very useful if you write a report, book or article in LaTeXand you want to have your bibliography, usually in a separated `.bib` file.

3.1.2.4 INDEX

After creating some functions and datasets, when we install our package, an `INDEX` file is generated automatically, with a brief description of any object from our package. If we write `library(help = "GofCens")`, we have an output with the description of our package and the index of the functions and datasets. Also, if we write `help(package = "GofCens")`, we have the index in a dedicated HTML page.

3.2 Functions

Another novelty of this package is the change of the names for both functions with respect to Anna Febrer in a classic programming convention, separated by dots, less personalization options with the purpose of making the code simpler and a change of plotting package to ggplot2 for better aesthetics, although we pay the price in form of slower execution due to the package gridExtra (Auguie, 2016). With the microbenchmark package (Mersmann, 2015), we checked the performance of both functions. 7 to 8 times slower, and with lineprof (Wickham, 2017), we found the main bottleneck in our functions, the already mentioned grid.

3.2.1 Function prob.plot

The purpose of the function is to check the fitting of the theoretical distribution versus the empirical distribution. The arguments are the following:

```
prob.plot <- function(time, cens, distribution = c("exp", "weibull", "gumbel",
    "norm", "lnorm", "logis", "loglogis", "beta", "exppower"),
    beta.limits = c(0, 1), parameters = list(shape = NULL, shape2 = NULL,
        location = NULL, scale = NULL))
```

In the first argument, time, we specify a numeric variable with the time of our subjects for the survival analysis. For the next argument, cens, we specify a binary variable, usually numeric if we follow the R notation for data[1], with the following values, 0 and 1, where 0 indicates right censoring. The third and last mandatory argument, distribution, is the theoretical distribution we want to test for our data.

For last, we have two optional arguments. The first is beta.limits, useful if we choose the beta distribution, and the second is parameters, if we want to input specify parameters when we choose our desired distribution, those will be calculated by maximum likelihood.

When we execute our function, we have a grid of four plots, under ggplot2 (Wickham, 2009) with its general parameters, shape and theoretical distribution. With the shape, we can explain the fit of our data with respect to the theoretical distribution, and with the location, we know the spread of our data (the location is related to parameters as μ in the normal distribution). The output is the following, with the dataset aml from the package survival, with the variables time and status, indicating us the survival in patients with Acute Myelogenous Leukemia:

[1] although can be a integer with the following values, 0L and 1L

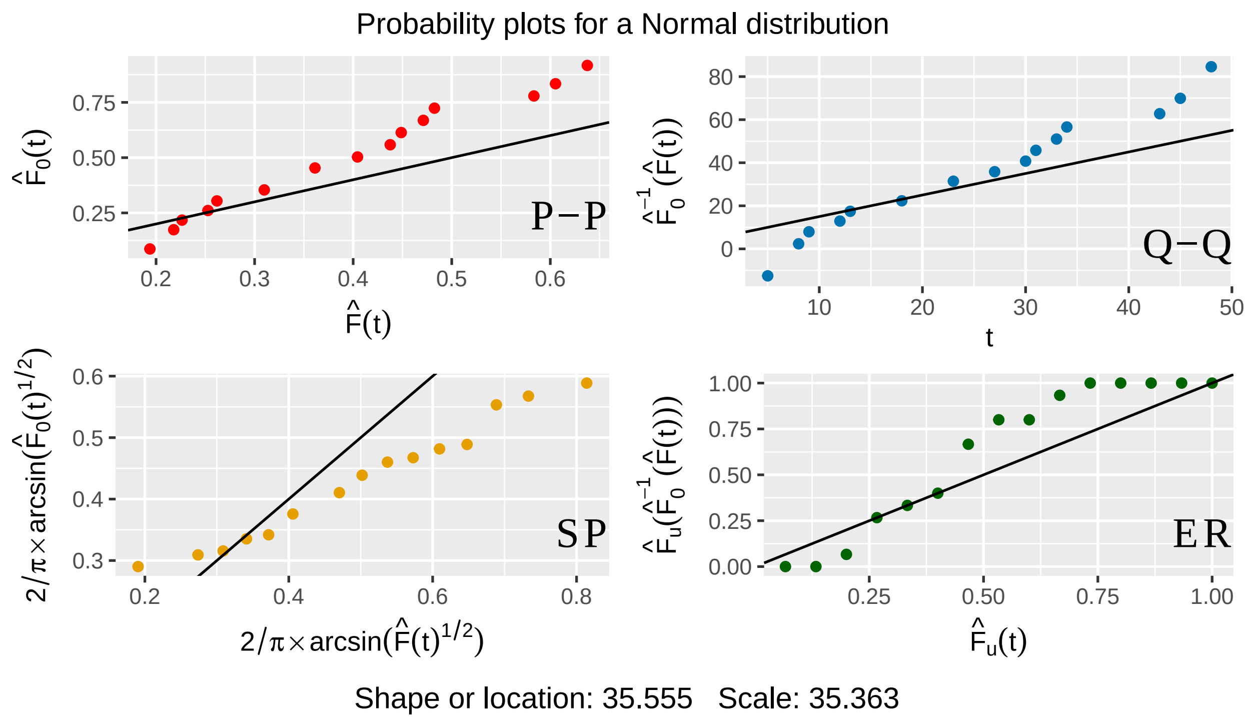


Figure 3.1: Example of prob.plot function

We have, then, a grid of four plots. In overall, the variable doesn't fit very well with the probability plots. That suggests us to choose another distribution.

3.2.2 Function `cum.haz.plot`

The purpose of this function is to check the fitting of our survival data for each function. The arguments are the following:

```
cum.haz.plot <- function(time, cens, beta.limits = c(0, 1))
```

We have the mandatory arguments `time`, for the time variable, numeric, and `cens`, for the censoring variable, with two values, 0 and 1. Plus, we have the argument `beta.limits`, with two values, the lower and upper bounds, useful if we want to take in account the beta distribution too. When we execute the function, we obtain a grid of nine plots, an improvement, although for that, I used a trick, changing the values of the parameters calculated by maximum likelihood. Plus the calculated parameters of shape/location and scale for all the distributions. If we don't specify any values for the beta limits or we use data between 0 and 1, then the plot related to the beta distribution will have a message, "out of range". In the figure shown below, we show an example applying our function to the variables `stop` and `event` from the dataset `bladder`, from `survival` package (Therneau, 2015).

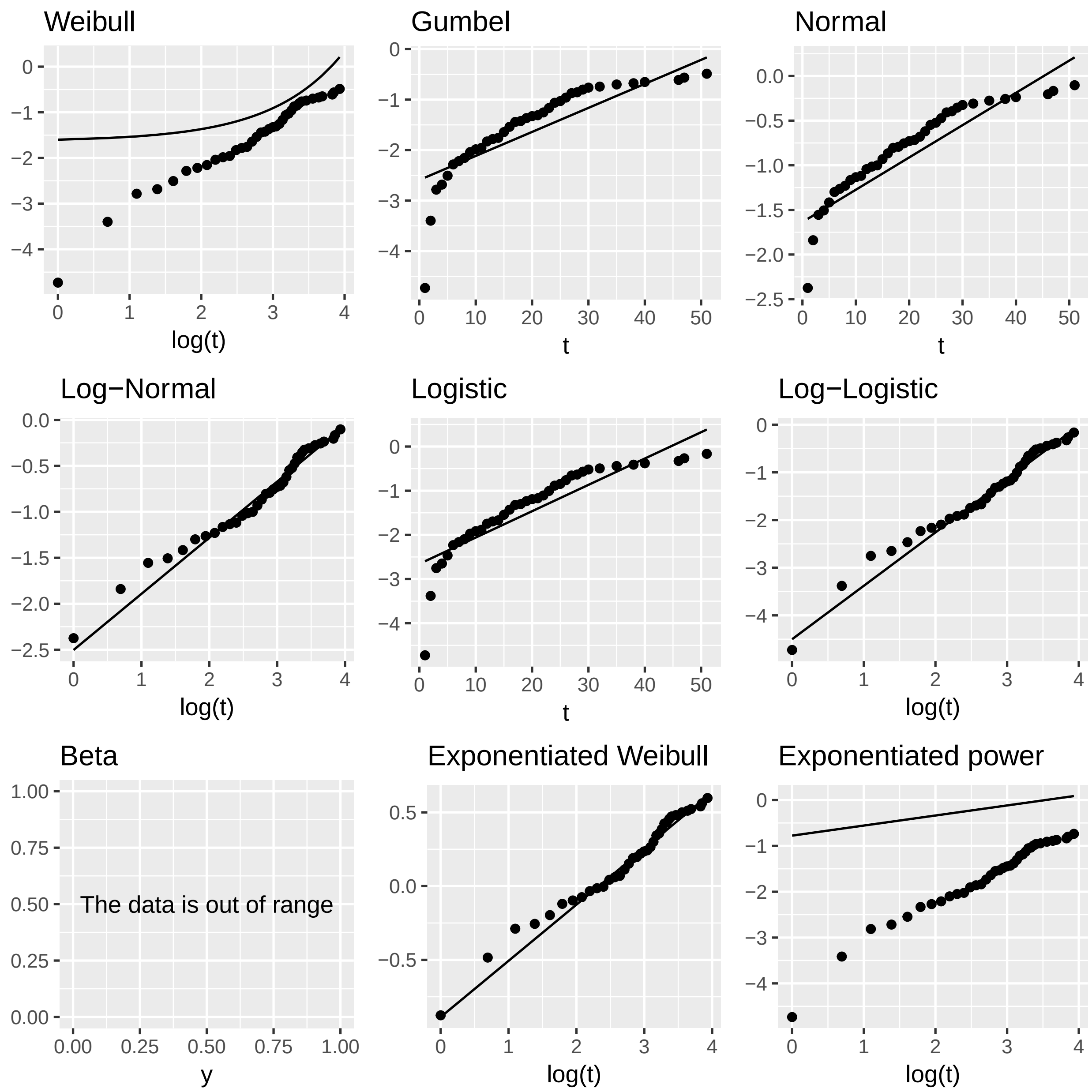


Figure 3.2: Example of function cum.haz.plot

```
## $weibull
##      shape      scale
##  0.9859713 77.0350620
##
## $lognormal
## location    scale
## 4.099860 1.638241
##
## $loglogistic
##     shape     scale
##  1.119871 55.595922
##
## $expweibull
##     shape1     shape2      scale
## 0.38114162 4.39537849 0.09733064
##
## $exppower
##      shape      scale
## 0.21949836 0.02923544
##
## $gumbel
## location    scale
## 54.45799 21.01467
##
## $normal
## location    scale
## 45.21056 27.63363
##
## $logistic
## location    scale
## 44.55244 16.77359
```

We have nine graphs for each distribution, plus a list of parameters for each distribution. We can have, then, a glimpse for choosing the best distribution. In this case, the better distributions could be the log normal, exponentiated weibull or log-logistic (the Normal distribution, for example, has some outliers).

3.3 Datasets

Our package has two datasets, `gbcs` and `whas500`, although in a future we can incorporate more.

3.3.1 Dataset `gbcs`

This dataset, from David W. Hosmer (2008), is from several cancer clinical trials in Germany, from the German Breast Cancer Study Group, with the purpose to illustrate methods for building prognostic models (Sauerbrei & Royston, 1999).

We can invoke that using the R command `GofCens::gbcs`. Its variables, 16, are the following:

1. `id`: Study ID, an integer between 1 and 686.

2. diagdate: Date of diagnosis in the following format, a factor variable with the ddmmyyyy. Can be transformed with as.date().

3. recdate: Date of Recurrence or of Recurrence Free Survival, with the format ddmmyyyy.

4. deathdate: Date of Death, with the format ddmmyyyy.

5. age: Age at diagnosis, an integer.

6. menopause: Menopausal status, an integer with two possible states, 1 = Yes, 2 = No.

7. hormone: Hormone Therapy, an integer with two possible states, 1 = Yes, 2 = No.

8. size: Tumor size in milimeters, an integer.

9. grade: Tumor grade, a factorial variable with three values, 1 to 3.

10. nodes: Number of nodes involved, an integer with values from 1 to 51.

11. prog_recp: Number of Progesterone Receptors, an integer between 1 and 2380.

12. estrg_recp: Number of Estrogen Receptors, from 1 to 1144 in integer format.

13. rectime: Time to recurrence, counted in days.

14. censrec: Recurrence Censoring, a binary variable with the following values, 0 = Censored and 1 = Recurrence.

15. survtime: Time to death, counted in days.

16. censdead: Death Censoring, a binary variable with two values, the censored, equal to 0, and the death, equal to 1.

For a brief exploration:

```
head(gbcs, 10)

##    id  diagdateb    recdate  deathdate age menopause hormone size grade
## 1   1 17/08/1984 15/04/1988 16/11/1990  38         1       1   18     3
## 2   2 25/04/1985 15/03/1989 22/10/1990  52         1       1   20     1
## 3   3 11/10/1984 12/04/1988 06/10/1988  47         1       1   30     2
## 4   4 29/06/1984 24/11/1984 24/11/1984  40         1       1   24     1
## 5   5 03/07/1984 09/08/1989 09/08/1989  64         2       2   19     2
## 6   6 24/07/1984 08/11/1989 08/11/1989  49         2       2   56     1
## 7   7 26/06/1985 19/06/1986 16/08/1986  53         2       1   52     2
## 8   8 10/09/1984 10/03/1991 10/09/1991  61         2       2   22     2
## 9   9 05/11/1984 12/11/1991 12/11/1991  43         1       1   30     2
## 10 10 14/06/1985 12/12/1991 12/12/1991  74         2       2   20     2
##    nodes prog_recp estrg_recp rectime censrec survtime censdead
## 1      5       141        105    1337       1     2282        0
## 2      1        78         14    1420       1     2006        0
## 3      1       422         89    1279       1     1456        1
## 4      3        25         11     148       0      148        0
## 5      1        19          9    1863       0     1863        0
## 6      3       356         64    1933       0     1933        0
## 7      9         6         29     358       1      416        1
## 8      2         6        173    2372       1     2556        0
## 9      1        22          0    2563       0     2563        0
## 10     1       462        240    2372       0     2372        0
```

```
summary(gbcs)
```

```
##        id             diagdateb         recdate          deathdate  
##  Min.   :  1.0   05/02/1986:  4   15/10/1990:  9   25/09/1991:  8  
##  1st Qu.:172.2   15/04/1987:  4   25/09/1991:  9   04/12/1991:  7  
##  Median :343.5   15/12/1987:  4   04/12/1991:  7   15/10/1990:  7  
##  Mean   :343.5   18/06/1986:  4   22/01/1992:  6   15/01/1991:  6  
##  3rd Qu.:514.8   22/01/1986:  4   23/10/1991:  6   22/01/1992:  6  
##  Max.   :686.0   23/01/1986:  4   15/01/1992:  5   23/10/1991:  6  
##                  (Other)   :662   (Other)   :644   (Other)   :646  
##       age          menopause        hormone           size       
##  Min.   :21.00   Min.   :1.000   Min.   :1.000   Min.   :  3.00  
##  1st Qu.:46.00   1st Qu.:1.000   1st Qu.:1.000   1st Qu.: 20.00  
##  Median :53.00   Median :2.000   Median :1.000   Median : 25.00  
##  Mean   :53.05   Mean   :1.577   Mean   :1.359   Mean   : 29.33  
##  3rd Qu.:61.00   3rd Qu.:2.000   3rd Qu.:2.000   3rd Qu.: 35.00  
##  Max.   :80.00   Max.   :2.000   Max.   :2.000   Max.   :120.00  
##                                                                  
##      grade           nodes          prog_recp         estrg_recp     
##  Min.   :1.000   Min.   : 1.00   Min.   :   0.0   Min.   :   0.00  
##  1st Qu.:2.000   1st Qu.: 1.00   1st Qu.:   7.0   1st Qu.:   8.00  
##  Median :2.000   Median : 3.00   Median :  32.5   Median :  36.00  
##  Mean   :2.117   Mean   : 5.01   Mean   : 110.0   Mean   :  96.25  
##  3rd Qu.:2.000   3rd Qu.: 7.00   3rd Qu.: 131.8   3rd Qu.: 114.00  
##  Max.   :3.000   Max.   :51.00   Max.   :2380.0   Max.   :1144.00  
##                                                                    
##     rectime          censrec          survtime         censdead     
##  Min.   :   8.0   Min.   :0.0000   Min.   :   8.0   Min.   :0.0000  
##  1st Qu.: 567.8   1st Qu.:0.0000   1st Qu.: 798.8   1st Qu.:0.0000  
##  Median :1084.0   Median :0.0000   Median :1338.0   Median :0.0000  
##  Mean   :1124.5   Mean   :0.4359   Mean   :1320.6   Mean   :0.2493  
##  3rd Qu.:1684.8   3rd Qu.:1.0000   3rd Qu.:1824.8   3rd Qu.:0.0000  
##  Max.   :2659.0   Max.   :1.0000   Max.   :2668.0   Max.   :1.0000
##
```

The average age is 53. Other data of interest are the incidence of the hormone therapy, with a 35.9 per cent of women with hormone therapy, an average of 29.33 milimeters for the tumor size and, for last, the average number of receptors for progesterone and estrogen are 110 and NA, respectively.

The median time to recurrence is 1807 days.

3.3.2 Dataset `whas500`

This dataset shows us the results of a longitudinal study with 1-year periods from 1975 to 2001 for patients admitted in the hospital of Worcester, in Massachusetts, for acute myocardial infarction (David W. Hosmer, 2008)[2], with the purpose of describing factors associated with trends over time in the incidence and survival rates following hospital admission for acute myocardial infarction. The variables are the following:

1. `id`: Identification number, a numeric vector with values from 1 to 500.
2. `age`: Age for each patient at the time of hospital addmision, a numeric vector.
3. `gender`: Gender, a binary variable with two values, 0 = Male and 1 = Female. Can be converted to an factorial with `as.factor`.
4. `hr`: Initial Heart Rate, beats per minute.
5. `sysbp`: Initial Systolic Blood Pressure, calculated in mmHg (millimeters of mercury).
6. `diasbp`: Initial Diastolic Blood Pressure, calculated in mmHg (millimeters of mercury).
7. `bmi`: Body Mass Index, a numeric vector, calculated in the following way, kg/m^2.
8. `cvd`: History of Cardiovascular Disease, a binary variable with two values, 0 = No and 1 = Yes.
9. `afb`: Atrial Fibrillation, a binary variable with two values, 0 = No and 1 = Yes.
10. `sho`: Cardiogenic Shock, a binary variable with two values, 0 = No and 1 = Yes.
11. `chf`: Congestive Heart Complications, a binary variable with two values, 0 = No and 1 = Yes.
12. `av3`: Complete Heart Block, a binary variable with two values, 0 = No and 1 = Yes.

[2] Originally, from Robert J. Goldberg, from the Departament of Cardiology at the Medical School in the University of Massachusets

13. miord: MI Order, a binary variable with two values, 0 = First and 1 = Recurrent.

14. mitype: MI type, a binary variable with two values, 0 = non Q-wave and 1 = Q-Wave.

15. year: Cohort Year, a numeric vector that can be converted in a factor with the following values: 1 = 1997, 2 = 1999 and 3 = 2001.

16. admitdate: Hospital Admission Date, a factor with levels mmddyyyy.

17. disdate: Hospital Discharge Rate, a factor with levels mmddyyyy.

18. fdate: Date of last Follow Up, a factor with levels mmddyyyy.

19. los: Length of Hospital Stay, a numeric vector with the days from hospital admission to hospital discharge.

20. dstat: Discharge Status from Hospital, a binary vector with two values, 0 = Alive, 1 = Dead.

21. lenfol: Total Length of Follow-up, a numeric vector that counts the days from the hospital admission date to the date of last follow-up.

22. fstat: Vital Status at Last Follow-up, a binary vector with two values, 0 = Alive, 1 = Dead.

A brief exploration in R is the following:

```
head(whas500, 10)

##    id age gender  hr sysbp diasbp      bmi cvd afb sho chf av3 miord
## 1   1  83      0  89   152     78 25.54051   1   1   0   0   0     1
## 2   2  49      0  84   120     60 24.02398   1   0   0   0   0     0
## 3   3  70      1  83   147     88 22.14290   0   0   0   0   0     0
## 4   4  70      0  65   123     76 26.63187   1   0   0   1   0     0
## 5   5  70      0  63   135     85 24.41255   1   0   0   0   0     0
## 6   6  70      0  76    83     54 23.24236   1   0   0   0   1     0
## 7   7  57      0  73   191    116 39.49046   1   0   0   0   0     0
## 8   8  55      0  91   147     95 27.11609   1   0   0   0   0     0
## 9   9  88      1  63   209    100 27.43554   1   0   0   1   0     0
## 10 10  54      0 104   166    106 25.54448   1   0   0   0   0     0
##    mitype year  admitdate    disdate      fdate los dstat lenfol fstat
## 1       0    1 13/01/1997 18/01/1997 31/12/2002   5     0   2178     0
## 2       1    1 19/01/1997 24/01/1997 31/12/2002   5     0   2172     0
## 3       1    1 01/01/1997 06/01/1997 31/12/2002   5     0   2190     0
## 4       1    1 17/02/1997 27/02/1997 11/12/1997  10     0    297     1
## 5       1    1 01/03/1997 07/03/1997 31/12/2002   6     0   2131     0
## 6       0    1 11/03/1997 12/03/1997 12/03/1997   1     1      1     1
## 7       1    1 10/03/1997 15/03/1997 31/12/2002   5     0   2122     0
## 8       1    1 11/01/1997 15/01/1997 15/02/2001   4     0   1496     1
## 9       0    1 31/12/1996 04/01/1997 09/07/1999   4     0    920     1
## 10      0    1 16/01/1997 21/01/1997 31/12/2002   5     0   2175     0
```

```
summary(whas500)
```

```
##        id             age             gender           hr
##  Min.   :  1.0   Min.   : 30.00   Min.   :0.0   Min.   : 35.00
##  1st Qu.:125.8   1st Qu.: 59.00   1st Qu.:0.0   1st Qu.: 69.00
##  Median :250.5   Median : 72.00   Median :0.0   Median : 85.00
##  Mean   :250.5   Mean   : 69.85   Mean   :0.4   Mean   : 87.02
##  3rd Qu.:375.2   3rd Qu.: 82.00   3rd Qu.:1.0   3rd Qu.:100.25
##  Max.   :500.0   Max.   :104.00   Max.   :1.0   Max.   :186.00
##
##      sysbp           diasbp            bmi             cvd
##  Min.   : 57.0   Min.   :  6.00   Min.   :13.05   Min.   :0.00
##  1st Qu.:123.0   1st Qu.: 63.00   1st Qu.:23.22   1st Qu.:0.75
##  Median :141.5   Median : 79.00   Median :25.95   Median :1.00
##  Mean   :144.7   Mean   : 78.27   Mean   :26.61   Mean   :0.75
##  3rd Qu.:164.0   3rd Qu.: 91.25   3rd Qu.:29.39   3rd Qu.:1.00
##  Max.   :244.0   Max.   :198.00   Max.   :44.84   Max.   :1.00
##
##       afb             sho             chf            av3
##  Min.   :0.000   Min.   :0.000   Min.   :0.00   Min.   :0.000
##  1st Qu.:0.000   1st Qu.:0.000   1st Qu.:0.00   1st Qu.:0.000
##  Median :0.000   Median :0.000   Median :0.00   Median :0.000
##  Mean   :0.156   Mean   :0.044   Mean   :0.31   Mean   :0.022
##  3rd Qu.:0.000   3rd Qu.:0.000   3rd Qu.:1.00   3rd Qu.:0.000
##  Max.   :1.000   Max.   :1.000   Max.   :1.00   Max.   :1.000
##
##      miord           mitype           year            admitdate
##  Min.   :0.000   Min.   :0.000   Min.   :1.000   12/10/2001:  4
##  1st Qu.:0.000   1st Qu.:0.000   1st Qu.:1.000   25/10/1999:  4
##  Median :0.000   Median :0.000   Median :2.000   29/06/2001:  4
##  Mean   :0.342   Mean   :0.306   Mean   :1.984   01/08/1999:  3
##  3rd Qu.:1.000   3rd Qu.:1.000   3rd Qu.:3.000   02/08/1999:  3
##  Max.   :1.000   Max.   :1.000   Max.   :3.000   04/10/1997:  3
##                                                  (Other)   :479
##        disdate             fdate          los             dstat
##  12/11/2001:  4   31/12/2002:285   Min.   : 0.000   Min.   :0.000
##  03/10/2001:  3   21/08/1999:  3   1st Qu.: 3.000   1st Qu.:0.000
##  04/08/1997:  3   07/12/2001:  2   Median : 5.000   Median :0.000
##  06/01/1997:  3   10/12/2001:  2   Mean   : 6.116   Mean   :0.078
##  08/06/2001:  3   11/12/2001:  2   3rd Qu.: 7.000   3rd Qu.:0.000
##  10/11/1997:  3   12/11/2001:  2   Max.   :47.000   Max.   :1.000
##  (Other)   :481   (Other)   :204
##      lenfol           fstat
##  Min.   :   1.0   Min.   :0.00
##  1st Qu.: 296.5   1st Qu.:0.00
##  Median : 631.5   Median :0.00
##  Mean   : 882.4   Mean   :0.43
##  3rd Qu.:1363.5   3rd Qu.:1.00
##  Max.   :2358.0   Max.   :1.00
##
```

The average age is 69.846, with an average of 66 years for the 300 males and an average of 74 years for the females. Other data of interest are the average BMI, 26.6137799, which means overweight, or the median length of the follow-up, 1627 days.

Chapter 4

APPLICATIONS WITH REAL DATA SETS

In this chapter, we'll try the two functions of our package, presented with detail in the former chapter, with two datasets, gbcs and whas500, also presented and explained. We'll use the function with() for a cleaner code, as in the following example:

```
with(gbcs, prob.plot(time, cens, "distribution"))
```

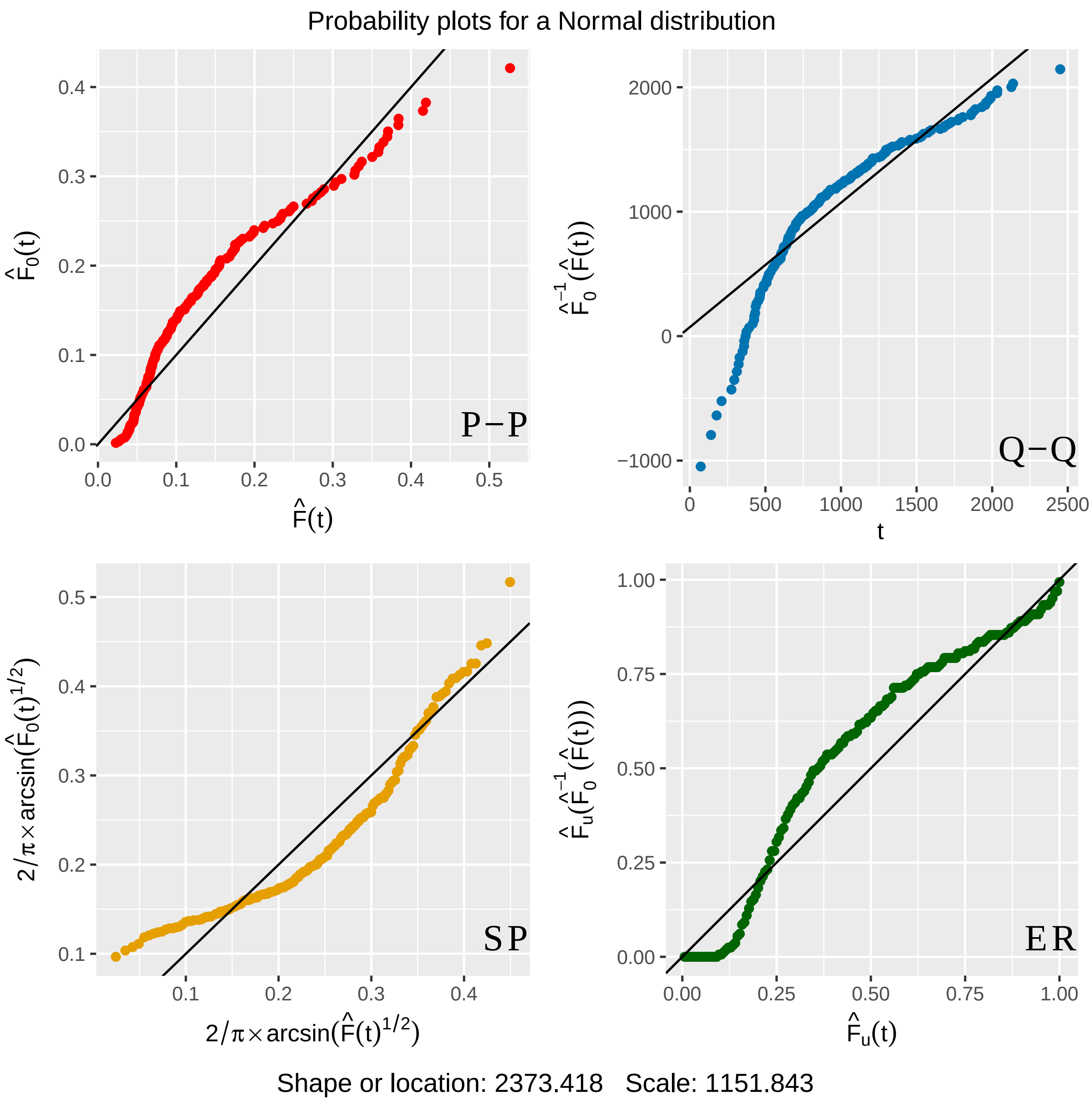
4.1 Probability Plots

We'll try this function with the ten possible distributions, showed below:

```
##  [1] "exp"         "weibull"    "gumbel"     "norm"       "lnorm"
##  [6] "logis"       "loglogis"   "beta"       "expweibull" "exppower"
```

We try this function with the variables of gbcs, survtime, the time to death, and censdead, a binary variable related to the censoring or not of the deaths. We'll start for the normal distribution:

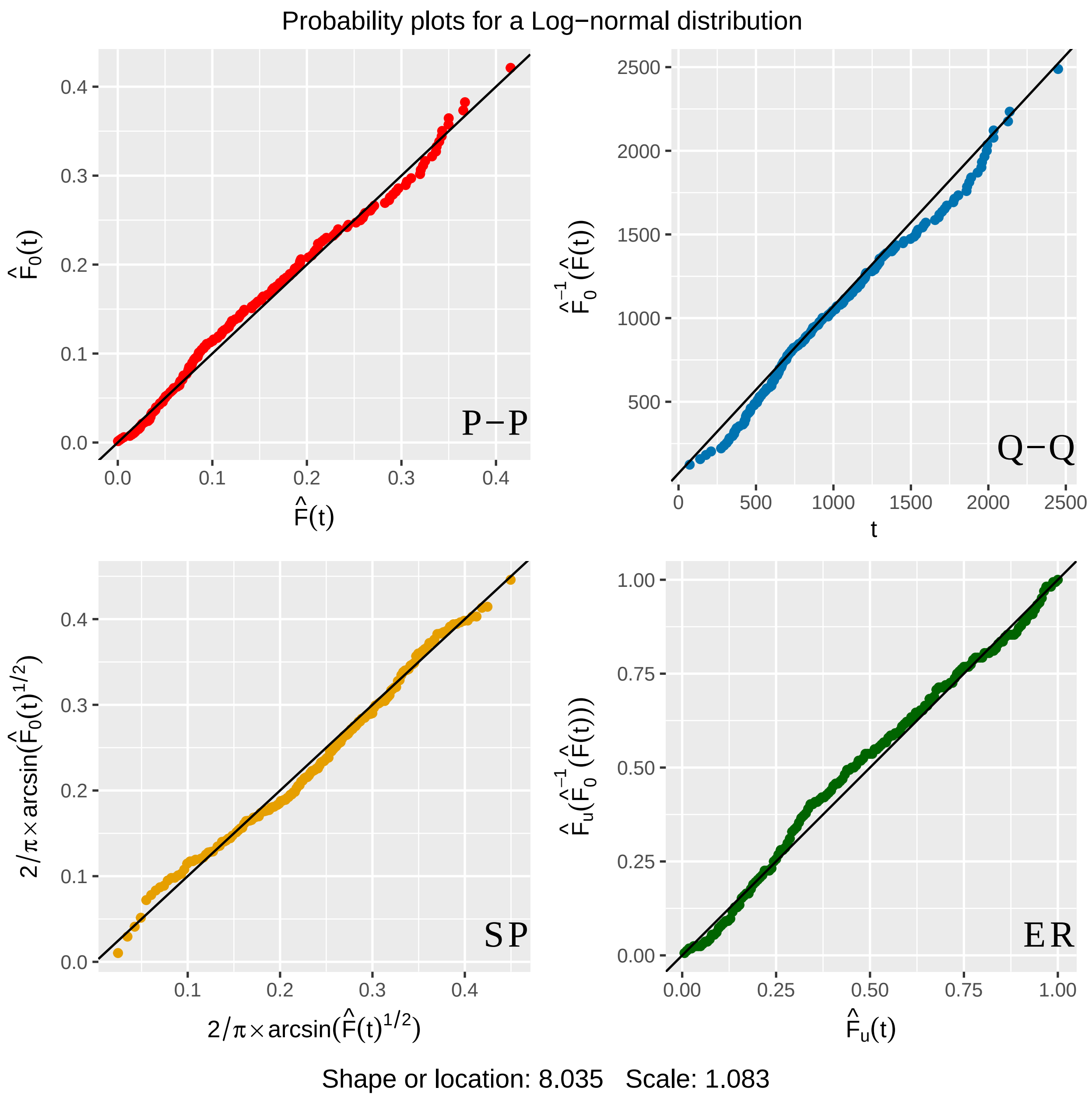
```
with(gbcs, prob.plot(survtime, censdead, "norm"))
```



We can check in this graph that the probability-probability and stabilised probability plots are the better, while the empirically rescaled plot are less adequate. In last place, we can discard the quartile-quartile plot, isn't adequate to the data.

Logistic-normal distribution:

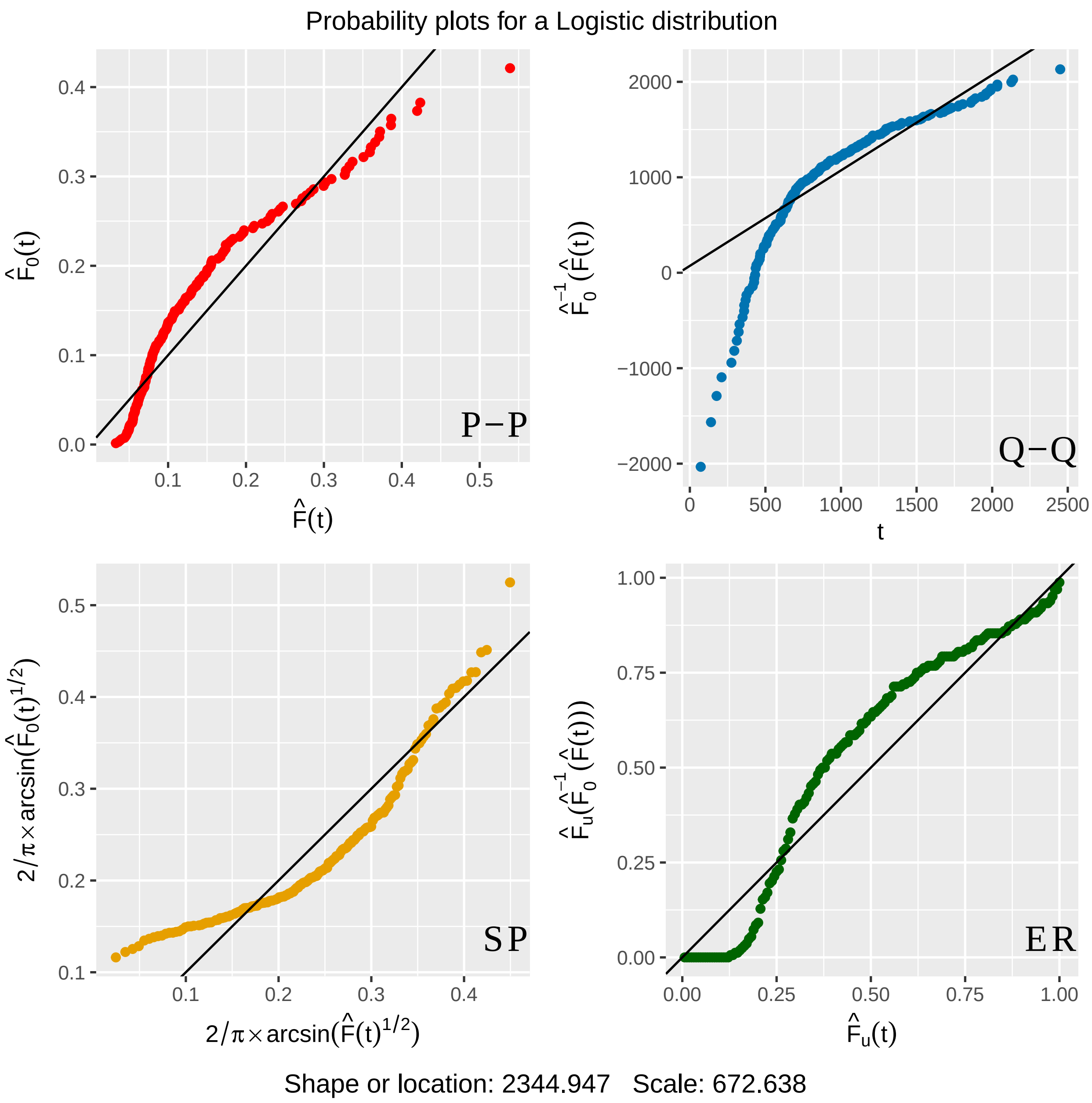
```
with(gbcs, prob.plot(survtime, censdead, "lnorm"))
```



For this distribution, all the four plots work very well. In overall, we can choose this distribution.

Logistic distribution:

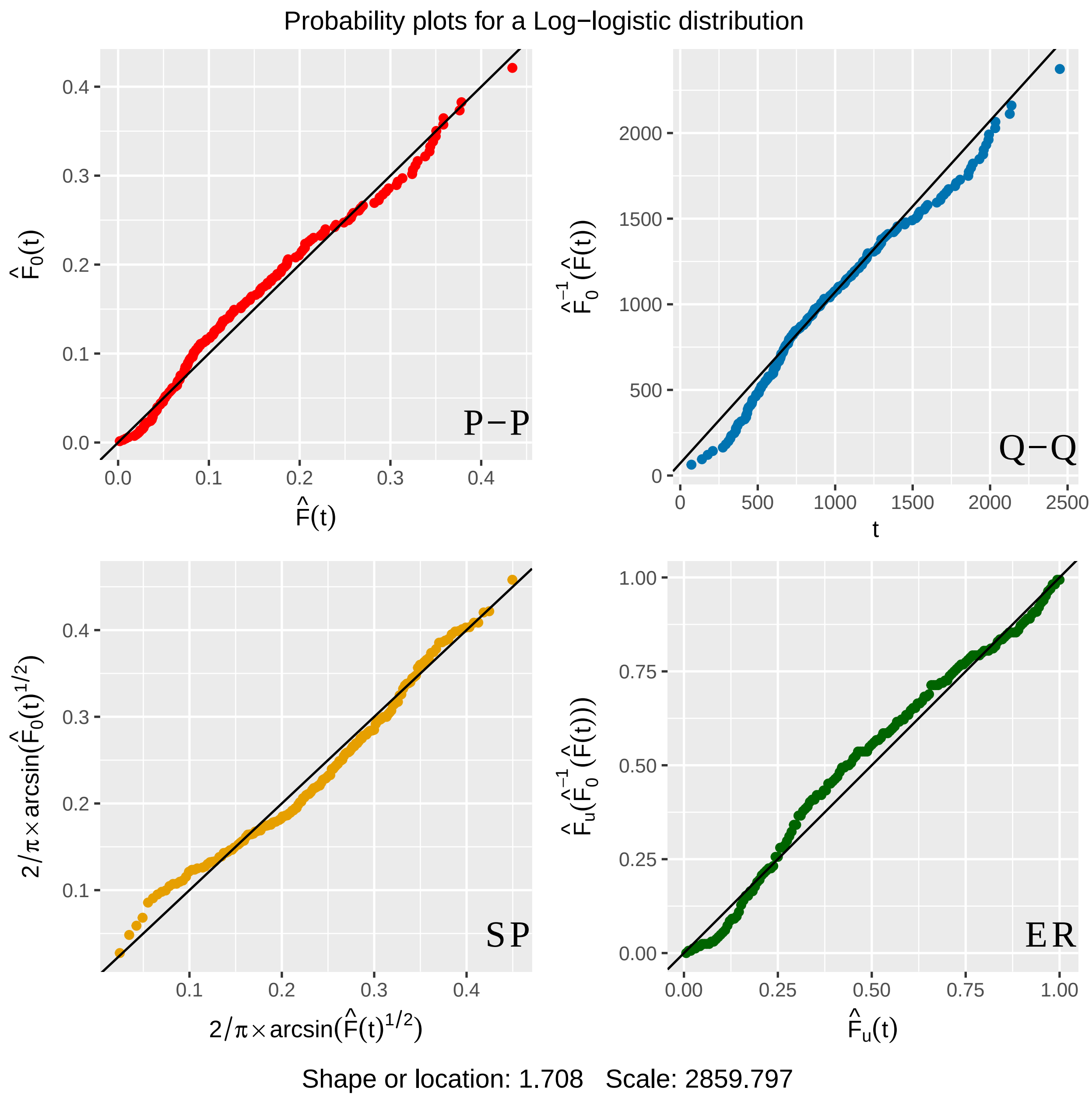
```r
with(gbcs, prob.plot(survtime, censdead, "logis"))
```



There, only the P-P plot and, in a lesser extent, the SP plot fits relatively well with the data for a logistic. We can discard the Q-Q plot, definitely.

Log-logistic distribution:

```
with(gbcs, prob.plot(survtime, censdead, "loglogis"))
```



There, we see a strange value. The slope of the Q-Q plot are very inusual and, in a lesser extent, the slopes of the P-P and SP plots. Only the ER plot, that fits very well, has a normal slope. We can check that, in overall, the better distribution is the log-normal and, in a lesser sense, the normal distribution

Now, we have other five distributions which present different types of errors, starting for the exponential distribution.

```
with(gbcs, prob.plot(survtime, censdead, "exp"))

## <simpleError in optim(par = vstart, fn = fnobjcens, fix.arg = fix.arg, gr = gradient,

## Error in fitdistcens(data, "exp"):  the function mle failed to estimate the
```

```
parameters,
##        with the error code 100
```

We can see that the exponential distribution fails due to an incorrect calculation in the maximum likelihood parameters. That indicates us that maybe the data doesn't follow on exponential distribution. We can try to change the default values for the MLE or indicate some parameters. We'll try the same version with parameters.

```
with(gbcs, prob.plot(survtime, censdead, "exp", parameters = list(shape = 0.091, shape2
        location = NULL, scale = NULL)))

## <simpleError in optim(par = vstart, fn = fnobjcens, fix.arg = fix.arg, gr = gradient,

## Error in fitdistcens(data, "exp"):  the function mle failed to estimate the
parameters,
##        with the error code 100
```

We find the same issue for the shape argument (although in the exponential distribution is the scale parameter). With two values, 1 and 0.091, we have the same error in MLE. We'll try the other distributions.

Weibull distribution:

```
with(gbcs, prob.plot(survtime, censdead, "weibull"))

## Error in while (theor.QQ[i] > empirical_f[1, index + 1]) index <- index + :
missing value where TRUE/FALSE needed
```

We found the next issue:

```
## Error in while (theor.QQ[i] > empirical_f[1, index + 1]) index <- index +
:missing value where TRUE/FALSE needed
```

That's an interesting issue for a future revision of the package, related with a possible error in the coding of the `theor.QQ`.

Gumbel distribution:

```
with(gbcs, prob.plot(survtime, censdead, "gumbel"))

## <simpleError in optim(par = vstart, fn = fnobjcens, fix.arg = fix.arg, gr = gradient,

## Error in fitdistcens(data, "gumbel", start = list(mu = -3, beta = 3)):  the
function mle failed to estimate the parameters,
##        with the error code 100
```

We have the same issue with the maximum likelihood estimation.

Beta distribution:

```
with(gbcs, prob.plot(survtime, censdead, "beta", beta.limits=c(as.numeric(0), as.numeric

## Error in start.arg.default(data10, distr = distname):  values must be in [0-1]
to fit a beta distribution
```

We have an error with the beta distribution, because the values of the variable `survtime` must be between 0 and 1.

For the exponential power distribution, we also have problems for estimating the parameters:

```
with(gbcs, prob.plot(survtime, censdead, "exppower"))

## <simpleError in optim(par = vstart, fn = fnobjcens, fix.arg = fix.arg, gr = gradient,

## Error in fitdistcens(data, "exppow", start = list(alpha = 0.5, beta = 0.5)):
the function mle failed to estimate the parameters,
##         with the error code 100
```

For last, we use the exponential Weibull distribution:

```
with(gbcs, prob.plot(survtime, censdead, "expweibull"))

## <simpleError in optim(par = vstart, fn = fnobjcens, fix.arg = fix.arg, gr = gradient,

## Error in fitdistcens(data, "expwei", start = list(alpha = 1, gamma = 1, :  the
function mle failed to estimate the parameters,
##         with the error code 100
```

There, we'll found another error 100 when we estimate the parameters.

Now, we change to the `whas500` dataset and we have the following results for all the distributions. We start with the exponential distribution and since the function won't estimate the parameters, we choose to try with a scale of 0.1:

```
# exp with parameters
with(whas500, prob.plot(lenfol, fstat, "exp"))

## Error in fitdistcens(data, "exp"):  the function mle failed to estimate the
parameters,
##         with the error code 100
```

```
# exponential
with(whas500, prob.plot(lenfol, fstat, "exp", parameters = list(shape = NULL, shape2 = N
        location = NULL, scale = 0.1)))

## Error in round(out.p$shape, 3):  non-numeric argument to mathematical function

# gumbel
with(whas500, prob.plot(lenfol, fstat, "gumbel"))

## Error in fitdistcens(data, "gumbel", start = list(mu = -3, beta = 3)):  the
function mle failed to estimate the parameters,
##          with the error code 100

# exponential power
with(whas500, prob.plot(lenfol, fstat, "exppower"))

## Error in fitdistcens(data, "exppow", start = list(alpha = 0.5, beta = 0.5)):
the function mle failed to estimate the parameters,
##          with the error code 100

# exp-weibull
with(whas500, prob.plot(lenfol, fstat, "expweibull"))

## Error in fitdistcens(data, "expwei", start = list(alpha = 1, gamma = 1, :  the
function mle failed to estimate the parameters,
##          with the error code 100
```

With the exponential without parameters, we found the already known error 100. With the scale parameter, now we found an error related with a non-numeric argument. For the other three distributions, as with the exponential distribution with parameters, Gumbel, exponential power and exponential Weibull, we have the same issues, the error 100, related to the maximum likelihood estimation.

For the Weibull distribution:

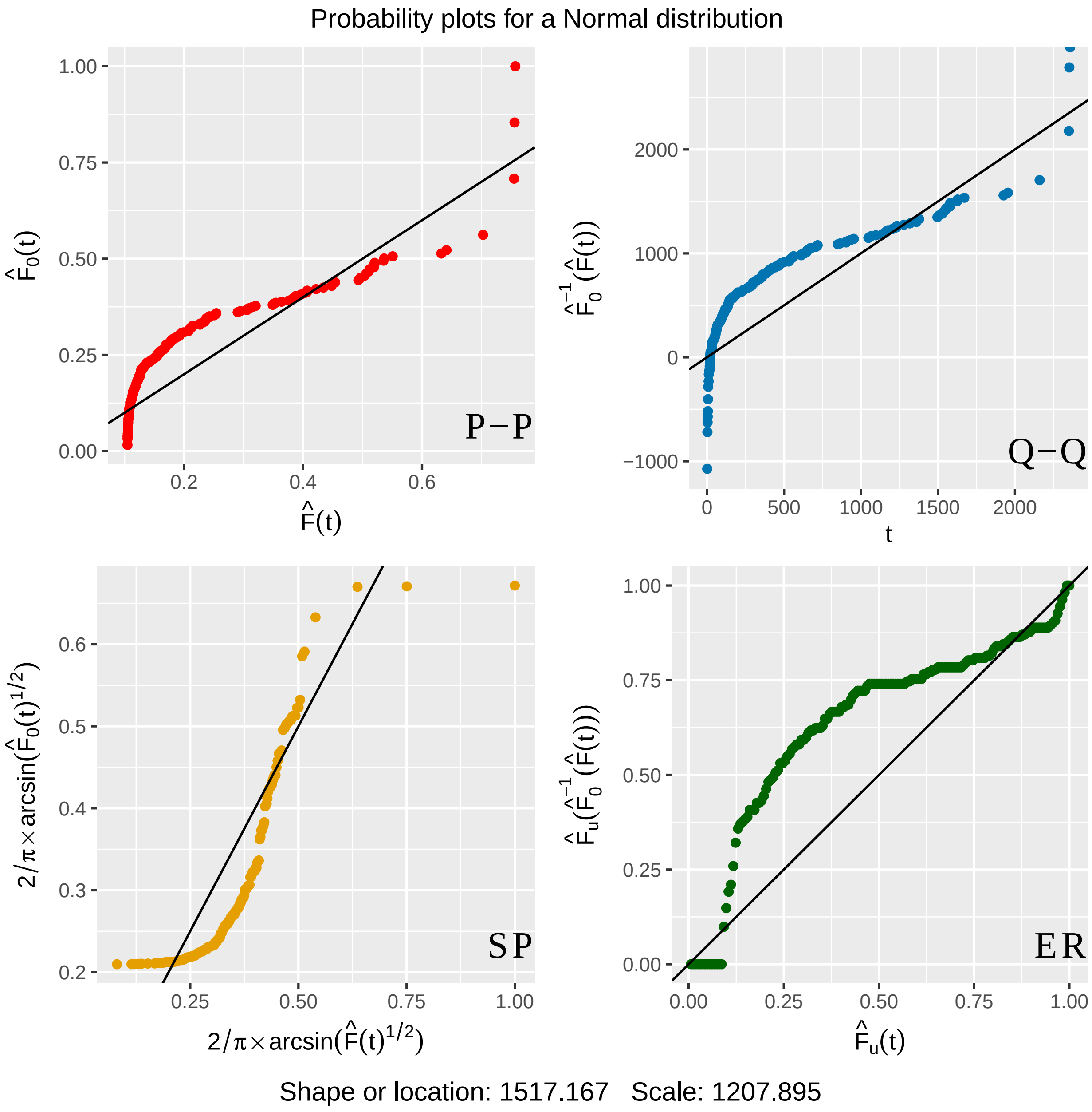
```
with(whas500, prob.plot(lenfol, fstat, "weibull"))

## Error in while (theor.QQ[i] > empirical_f[1, index + 1]) index <- index + :
missing value where TRUE/FALSE needed
```

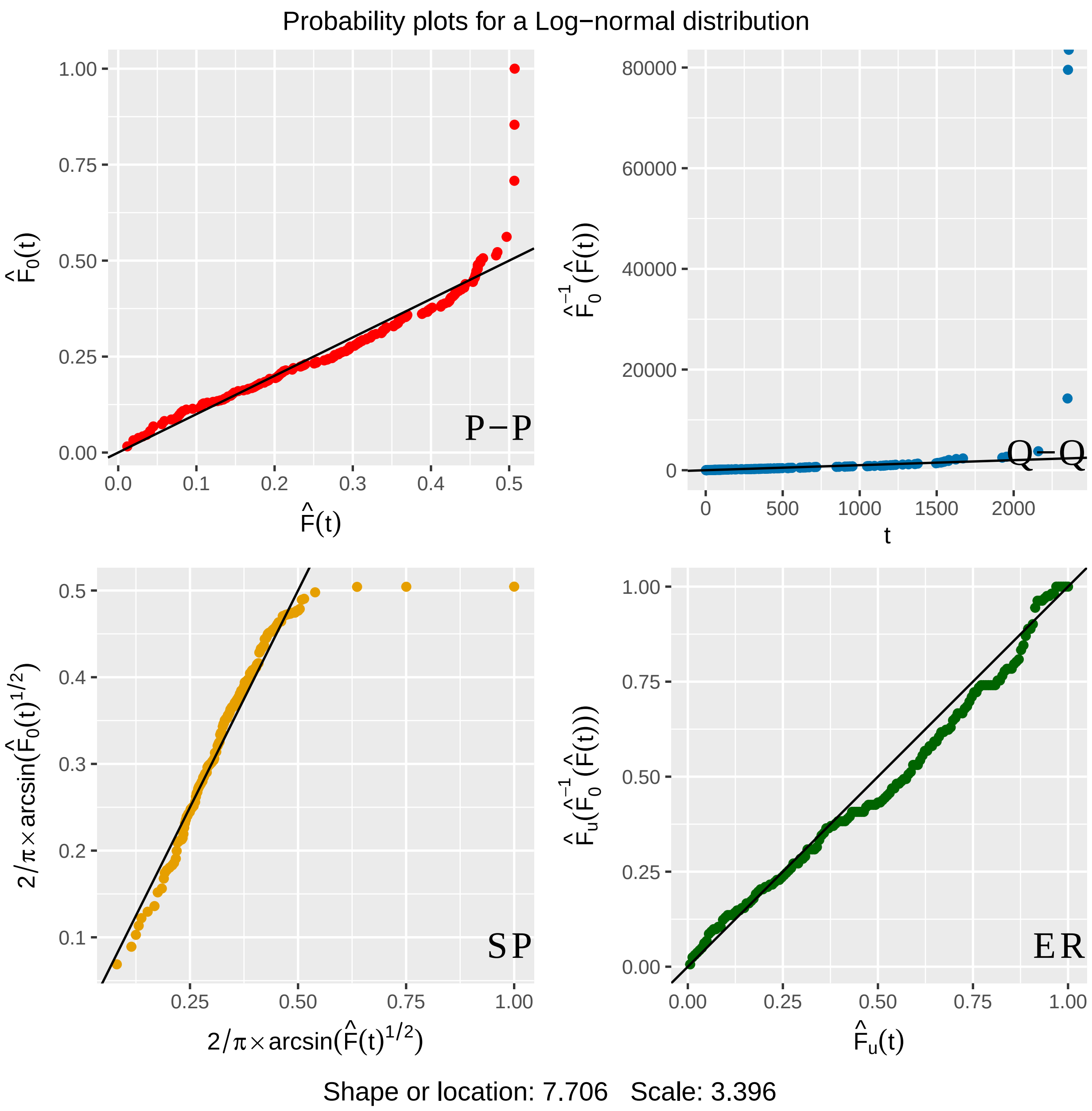
There, we have the same coding issue already commented.

Now, we'll try with the four functioning distributions, normal, logistic normal, logistic and log-logistic:

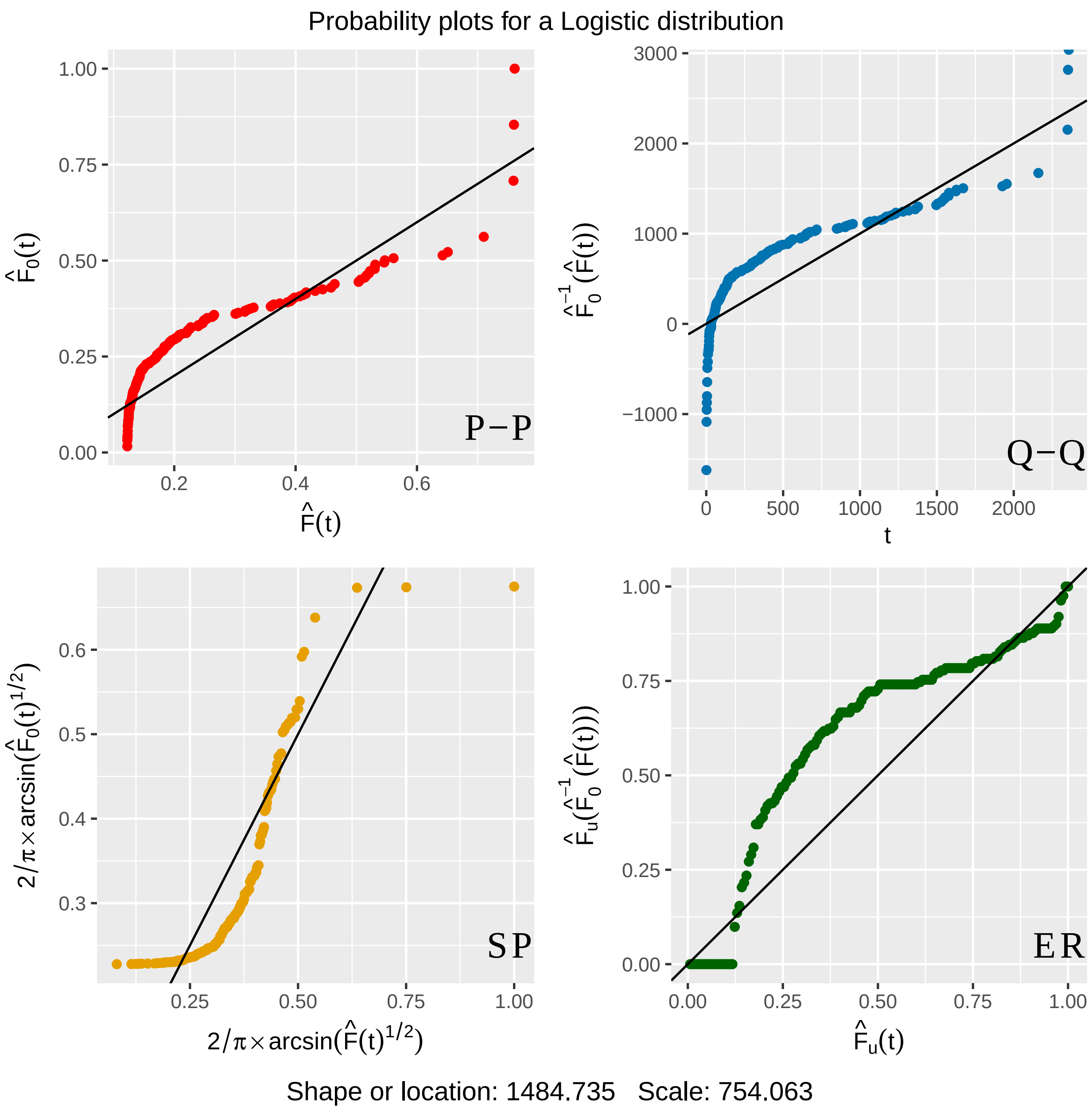
```
# normal
with(whas500, prob.plot(lenfol, fstat, "norm"))
```



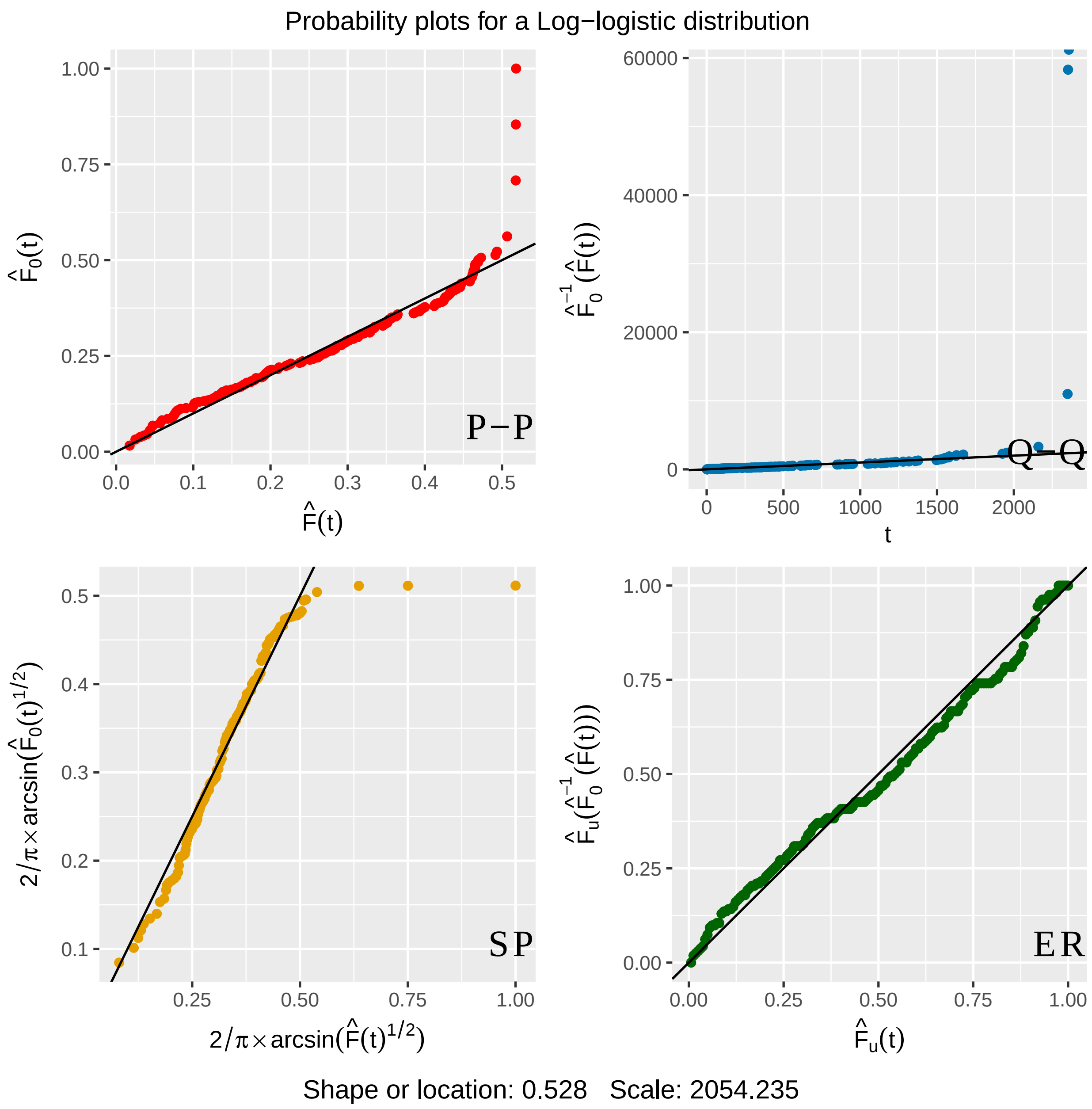
```
# logistic normal
with(whas500, prob.plot(lenfol, fstat, "lnorm"))
```



```r
# logistic
with(whas500, prob.plot(lenfol, fstat, "logis"))
```



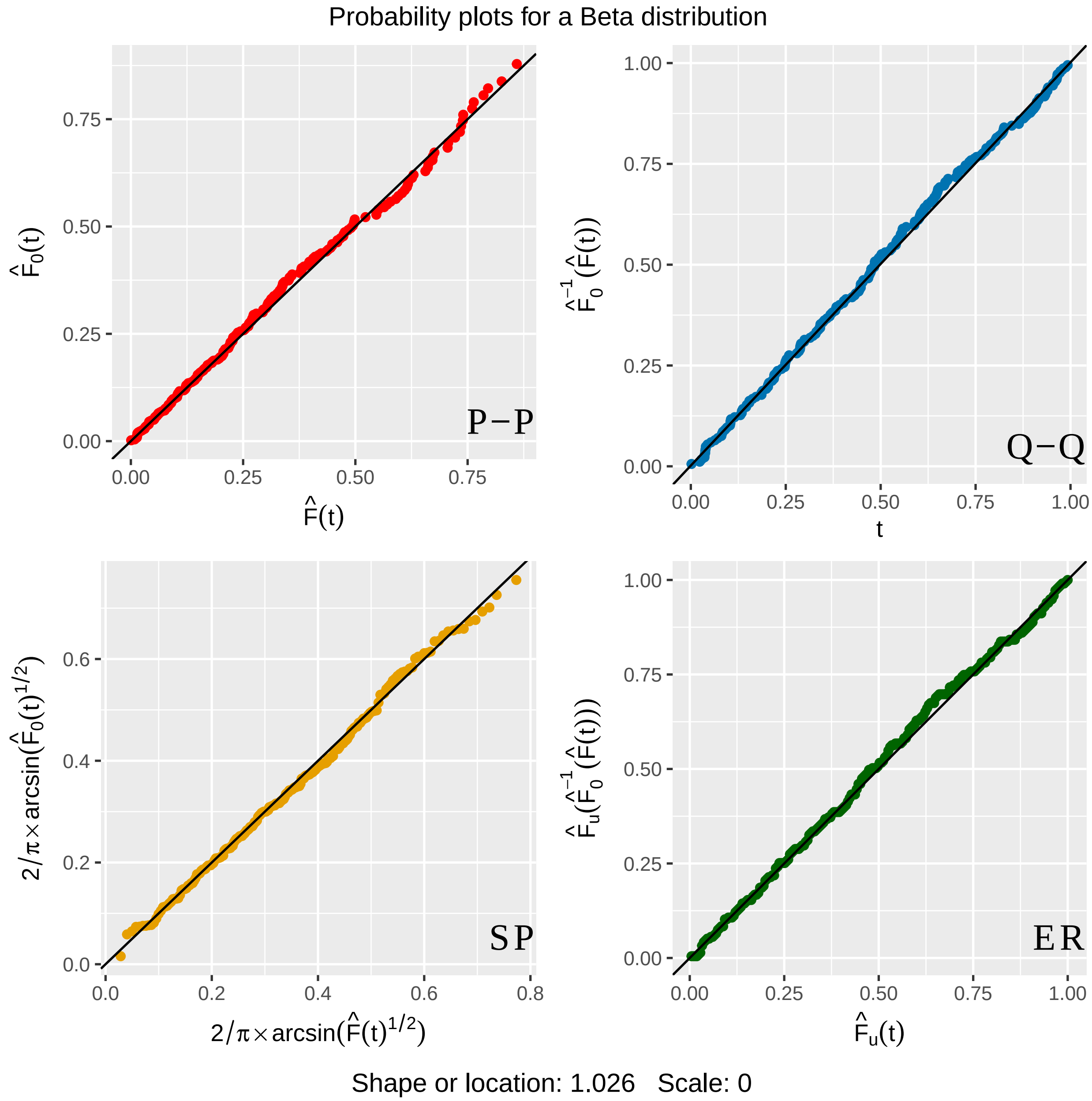
```
#log-logistic distribution
with(whas500, prob.plot(lenfol, fstat, "loglogis"))
```



The probability plots for all four distributions don't adjust totally well. For example, for the best functioning distributions, logistic normal and log-logistic, we have some outliers below or above the line. The other two distributions, normal and logistic, can be discarded. The points follow, in overall, a S-shape. The empirical distribution doesn't fit well with the theoretical.

For last, we can try the beta distribution with data between 0 and 1. For that, we create with the function `runif()` an uniformly distributed random variable:

```
with(whas500, prob.plot(runif(500), fstat, "beta"))
```



There, we'll find an almost perfect fit with all the plots.

4.1.1 Comments

For both datasets, we have four working distributions, normal, logistic, beta and log-normal. For the other datasets, diagnoses are the following: we need to improve the maximum likelihood estimations for several distributions: gumbel, exponential and exponential power, due to the error 100, or change the code. For example, we could use the package `ActuDistns`. For the other two distributions, log-logistic and weibull, we have different issues, related to coding. For last, for the log-logistic distribution, we solved the issue with the out.p, from this piece of code, so finally we have five working distributions:

```
qloglogis <<- function(p,alpha,beta) beta*(p/(1-p))^(1/alpha)
```

In this case, we'll save the function in the environment of a function with just the following assignment character, `<-`. Other issue is related to the naming of the location or shape. A possible solution for that could be introducing a equal convention for all the distributions, `shape` or `location`, although that can be confusing, but the structure of the function (the use of grids and ggplot2, for example) implies some generalization.

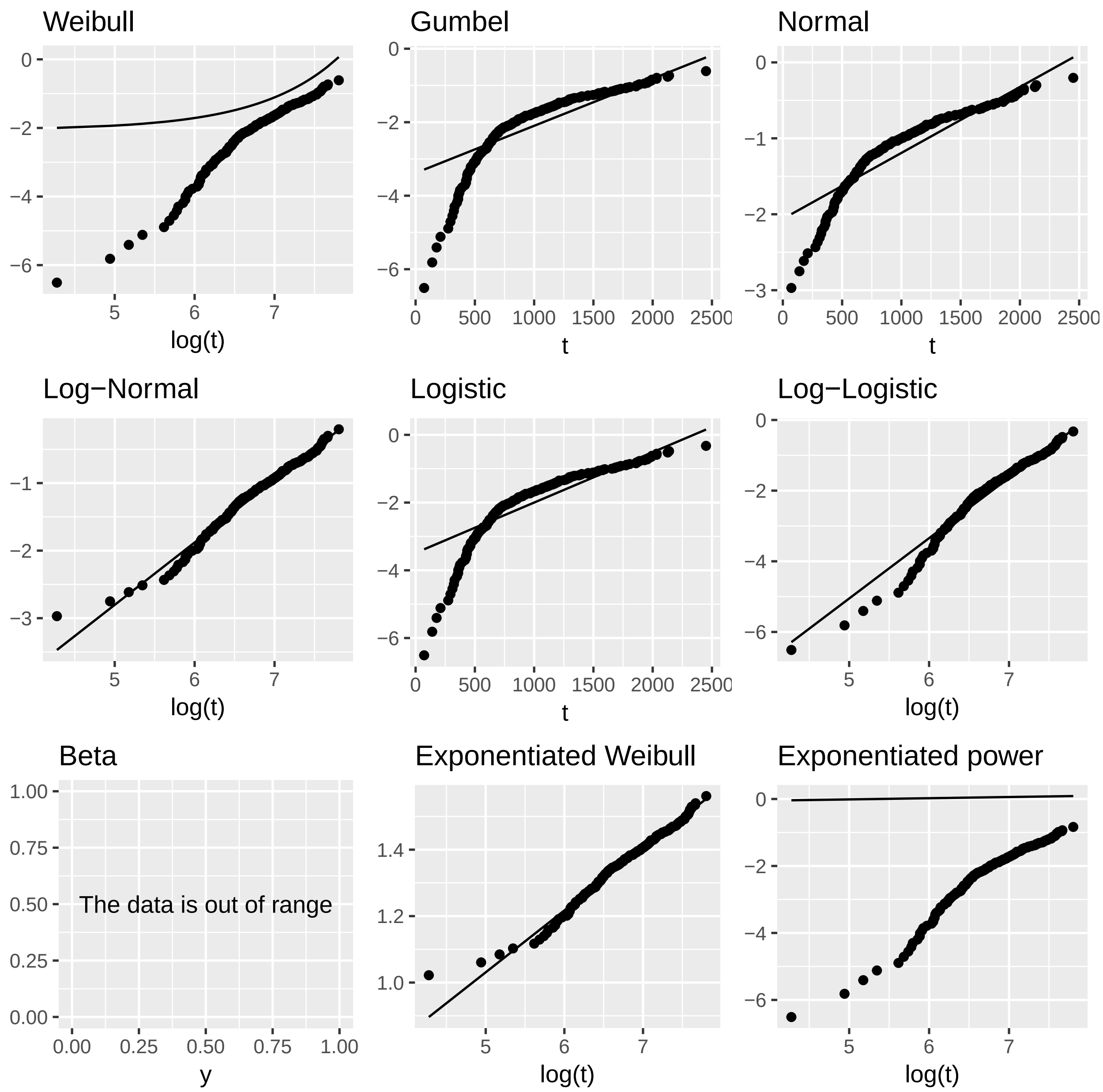
In overall, this function needs some reworking in future versions of this package.

4.2 Cumulative Hazard Plots

For this section, we have, for default, two ways. For default, we can test the cumulative hazard plots for the Weibull, Gumbel, Normal, Log-Normal, Logistic, Log-Logistic, Exponentiated Weibull and Exponentiated Power distributions. If our data are adequate, as explained in the former section, and we can use the beta distribution, then we can have the cumulative hazard for all distributions.

We execute the first example with the `gbcs` dataset.

```
with(gbcs, cum.haz.plot(survtime, censdead))
```



```
## $weibull
##       shape       scale
##    1.542139 3447.907817
##
## $lognormal
## location    scale
## 8.034958 1.082831
##
## $loglogistic
##       shape       scale
##    1.707972 2859.797238
```

```
##
## $expweibull
##      shape1      shape2       scale
##   0.1862231 101.5483373   1.7088390
##
## $exppower
##       shape       scale
## 0.035520058 0.004731005
##
## $gumbel
## location    scale
## 2634.958  779.245
##
## $normal
## location    scale
## 2373.418 1151.843
##
## $logistic
##  location     scale
## 2344.9468  672.6382
```

In this example, we can observe that the better distribution for our data are the log-normal and exponentiated weibull distributions, because the curve is very well fitted, with some outliers. To a lesser extent, we have the log-logistic distribution. For the exponential power distribution, we have a strange graph due to the maximum likelihood estimation of the parameters, 0.000000005 for the shape and scale. For last, the other distributions don't fit very well, as the weibull distribution, because there, we have our data points under the fit line. Under the grid of plots, we have a list of parameters for each distribution, shape or location and scale.

By other hand, with the `whas500` dataset, the plot doesn't work because in some . For the variables `los`, `dstat`, `lenfol` and `fstat`, we have the same issue, an error code 100. For the first pair of variables, we have as before an issue with the maximum likelihood estimation of the parameters. We can check that:

```
with(whas500, cum.haz.plot(los, dstat))


## <simpleError in optim(par = vstart, fn = fnobjcens, fix.arg = fix.arg, gr = gradient,


## Error in fitdistcens(data, "weibull"):  the function mle failed to estimate
the parameters,
##        with the error code 100


with(whas500, cum.haz.plot(lenfol, fstat))
```

```
## <simpleError in optim(par = vstart, fn = fnobjcens, fix.arg = fix.arg, gr = gradient,

## Error in fitdistcens(data, "expwei", start = list(shape1 = 0.01, shape2 = 0.01,
:  the function mle failed to estimate the parameters,
##        with the error code 100
```

We found issues with two different distributions, Weibull and Exponential Weibull.

4.2.1 Comments

The function brings us a grid of nine plots, although if one distribution fails, for example with the gbcs dataset, then we don't have the grid, because for default we choose to show all the distributions. For fixing that, we could use an universal conditional. If a distribution doesn't have the error 100, then we graph its hazard. If not (else), then we don't plot the distribution with this error. We can see it with an example: if we estimate some values for the parameters inside the code, with some data, as gbcs, will work, but with other data, as whas500, won't work.

This function, compared to prob.plot, is more independent for each calculation, so we don't need an universal naming convention for the parameters when we graph our hazards. Plus, some distributions, due to its parameters, have strange graphs. For next versions, we can recheck that and even use packages as ActuDistns.

In overall, the function also needs more reworking.

Chapter 5

CONCLUSIONS

In this present work, we presented a new R package or the goodness of fit with right-censored data. This package can be very useful for check the fit of our desired distribution according to our chosen data before to make our desired survival analysis. Without a good fit, we could fail.

We provided four parts: an overview of the survival analysis, the state of the art in goodness of fit for right-censored data (quantitative and graphical techniques), a brief exploration of the R landscape in these matters and a R package for the goodness of fit for the right-censored data and its applications, called `GofCens`.

For the first part, corresponding to the Chapters 1 and 2, we presented some building blocks of the survival analysis, as the Kaplan-Meier estimator, the censored data (especially the right censoring) or the distributions.

For the second part, we presented briefly two quantitative goodness of fit techniques, the Kolmogorov-Smirnov goodness of fit test, a well known statistical test, and Grané test of goodness-of-fit, especialized for right-censored data.

Connecting the two former parts, we presented the state of the art of R functions in survival analysis, especially in the distributions, the right-censored data and the goodness of fit. Also, we presented the two original functions from Anna Febrer, `ProbPlots` and `CumHazPlot`, our starting point for the R package.

Finally, we presented a R package dedicated to the graphical goodness of fit, `GofCens`, with two functions, its help pages, some essential files such the `DESCRIPTION` and two datasets. Also, we tested it under many possible circumstances, with the purpose of knowing its inner workings and limitations. The functions, in overall, have some issues, especially with the calculations of the parameters for our distributions by maximum likelihood and the speed of execution in both functions due to the `gridExtra` package.

The future

As we saw in the Chapter 4, the package have many issues, but this bring us an opportunity for improving many parts and, even more important, a long-term purpose for our package. Our intention is to mantain this package in a R repository for packages as

CRAN, so in a future we'll fix, add or recheck functions of this package. We have some examples of proposals:

1. Present an alternative, the base graphics. Two main functions there: `plot()` and `par()` for the grid.
2. Add more datasets.
3. Fix `prob.plot` and `cum.haz.plot`, make them less dependent on maximum likelihood calculations. One possible idea is to use the package `ActuDistns`, albeit we can lose flexibility.
4. Fix and improve the help pages.

For last, this thesis is the continuation of a long term project in the GRBIO (Grup de Recerca en Bioestadística i Bioinformàtica) related to goodness-of-fit techniques for right-censored data. In this package, we implemented R functions related to the graphic analysis. In a future, we can incorporate functions related to the other groups of techniques: chi-squared statistic, empirical survival function and correlation measures.

Appendices

Appendix A

R CODE

A.1 prob.plot

```
prob.plots <- function(time, cens, distribution=c("exp", "weibull", "gumbel", "
    norm", "lnorm", "logis", "loglogis", "beta", "expweibull", "exppower"), beta.
    limits=c(0,1), parameters=list(shape = NULL, shape2 = NULL, location = NULL,
    scale = NULL)){
  # check variables
  stopifnot(is.numeric(time), is.numeric(cens))

  # Transform the input data to the needed format
  input <- list(n = length(time), survKM = survfit(Surv(time, cens)~1, type='
      kaplan-meier'), data = data.frame(left=time, right=ifelse(cens==1, time, NA)
      ))

  # Compute the event times
  t = summary(input[[2]])$time

  # Compute the survival at the event times
  surv.value = summary(input[[2]])$surv
  uncensored = rep(1, length(t))
  u.point.surv = survfit(Surv(t, uncensored)~1, type='kaplan-meier')
  u.point = 1 - u.point.surv$surv
  empirical_f = rbind(c(0, t, Inf), c(0, u.point, 1))
  u.estimate = rep(0, length(u.point))
  in.p = parameters

  # distributions

  data <- input$data

  # exponential

  if(distribution=="exp") {
    if(is.null(in.p$scale)) {
      fit.exp <- fitdistcens(data, "exp")
      rate.exp <- unname(fit.exp$estimate[1])
    } else rate.exp <- 1/in.p$scale
    theor.PP <- pexp(t, rate.exp)
    theor.QQ <- qexp(1 - surv.value, rate.exp)
    out.p <- list(distrib = "Exponential", scale = 1/rate.exp)
  }
  # weibull
  if(distribution=="weibull"){
    if(is.null(in.p$shape) || is.null(in.p$scale)){
      fit.wei <- fitdistcens(data,"weibull")
      shape.wei <- unname(fit.wei$estimate[1])
      scale.wei <- unname(fit.wei$estimate[2])
    }
    else{
      shape.wei <- in.p$shape
      scale.wei <- in.p$scale
    }
    theor.PP <- pweibull(t, shape.wei, scale.wei)
    theor.QQ <- qweibull(1-surv.value, shape.wei, scale.wei)
    out.p <- list(distr = "Weibull", shape = shape.wei, scale = scale.wei)
  }
  # log-weibull (gumbel)
  if(distribution=="gumbel"){
    dgumbel <<- function(x,mu,beta){
      1/beta*exp((x-mu)/beta)*exp(-exp((x-mu)/beta))}
    pgumbel <<- function(q,mu,beta) 1-exp(-exp((q-mu)/beta))
    qgumbel <<- function(p,mu,beta) log(log(1/(1-p)))*beta+mu
    if(is.null(in.p$location) || is.null(in.p$scale)){
      fit.gum <- fitdistcens(data,"gumbel",start=list(mu=-3,beta=3))
      loc.gum <- unname(fit.gum$estimate[1])
      scale.gum <- unname(fit.gum$estimate[2])
    }
    else{
      loc.gum <- in.p$location
      scale.gum <- in.p$scale
    }
    theor.PP <- pgumbel(t, loc.gum, scale.gum)
    theor.QQ <- qgumbel(1-surv.value, loc.gum, scale.gum)
    out.p <- list(distr = "Gumbel", shape = loc.gum, scale = scale.gum)
  }
  # normal
  if(distribution=="norm"){
    if(is.null(in.p$location) || is.null(in.p$scale)){
      fit.norm <- fitdistcens(data,"norm")
      loc.norm <- unname(fit.norm$estimate[1])
      scale.norm <- unname(fit.norm$estimate[2])
    }
```

```
    else{
      loc.norm <- in.p$location
      scale.norm <- in.p$scale
    }
    theor.PP <- pnorm(t, loc.norm, scale.norm)
    theor.QQ <- qnorm(1-surv.value, loc.norm, scale.norm)
    out.p <- list(distr = "Normal", shape = loc.norm, scale = scale.norm)
  }
  # log-normal
  if(distribution=="lnorm"){
    if(is.null(in.p$location) || is.null(in.p$scale)){
      fit.lnorm <- fitdistcens(data, "lnorm")
      loc.lnorm <- unname(fit.lnorm$estimate[1])
      scale.lnorm <- unname(fit.lnorm$estimate[2])
    }
    else{
      loc.lnorm <- in.p$location
      scale.lnorm <- in.p$scale
    }
    theor.PP <- plnorm(t, loc.lnorm, scale.lnorm)
    theor.QQ <- qlnorm(1-surv.value, loc.lnorm, scale.lnorm)
    out.p <- list(distr = "Log-normal", shape = loc.lnorm, scale = scale.lnorm)
  }
  # logistic
  if(distribution=="logis"){
    if(is.null(in.p$location) || is.null(in.p$scale)){
      fit.log <- fitdistcens(data,"logis")
      loc.logis <- unname(fit.log$estimate[1])
      scale.logis <- unname(fit.log$estimate[2])
    }
    else{
      loc.logis <- in.p$location
      scale.logis <- in.p$scale
    }
    theor.PP <- plogis(t, loc.logis, scale.logis)
    theor.QQ <- qlogis(1-surv.value, loc.logis, scale.logis)
    out.p <- list(distr = "Logistic", shape = loc.logis, scale = scale.logis)
  }
  # log-logistic
  if(distribution=="loglogis"){
    dloglogis <<- function(x,alpha,beta) {
      (alpha*beta^(-alpha)*x^(alpha-1))/(1+(x/beta)^alpha)^2}
    ploglogis <<- function(q,alpha,beta) 1/(1+(q/beta)^(-alpha))
    qloglogis <- function(p,alpha,beta) beta*(p/(1-p))^(1/alpha)
    if(is.null(in.p$shape) || is.null(in.p$scale)){
      fit.loglog <- fitdistcens(data,"loglogis", start=list(alpha=1,beta=1))
      shape.loglogis <- unname(fit.loglog$estimate[1])
      scale.loglogis <- unname(fit.loglog$estimate[2])
    }
    else{
      shape.loglogis <- in.p$shape
      scale.loglogis <- in.p$scale
    }
    theor.PP <- ploglogis(t, shape.loglogis, scale.loglogis)
    theor.QQ <- qloglogis(1-surv.value, shape.loglogis, scale.loglogis)
    out.p <- list(distr = "Log-logistic", shape = shape.loglogis, scale = scale.
       loglogis)
  }
  # beta
  if(distribution=="beta"){
    a.beta<-beta.limits[1]
    b.beta<-beta.limits[2]
    if(is.null(in.p$shape) || is.null(in.p$shape2)){
      fit.beta <- fitdistcens((data-a.beta)/(b.beta-a.beta),"beta")
      shape1.beta <- unname(fit.beta$estimate[1])
      shape2.beta <- unname(fit.beta$estimate[2])
    }
    else{
      shape1.beta <- in.p$shape
      shape2.beta <- in.p$shape2
    }
    theor.PP <- pbeta((t-a.beta)/(b.beta-a.beta), shape1.beta, shape2.beta)
    theor.QQ <- qbeta((1-surv.value), shape1.beta, shape2.beta)*
      (b.beta-a.beta)+a.beta
    out.p <- list(distr = "Beta", shape = shape1.beta, shape2 = shape2.beta,
                 scale = beta.limits)
  }
  # Exponentiated Weibull
  if(distribution=="expweibull"){
    dexpwei <<- function(x,alpha,gamma,beta){
      gamma*alpha*beta^alpha*x^(alpha-1)*
        exp(-(beta*x)^alpha)*(1-exp(-(beta*x)^alpha))^(gamma-1)}
    pexpwei <<- function(q,alpha,gamma,beta) (1-exp(-(beta*q)^alpha))^gamma
    qexpwei <<- function(p,alpha,gamma,beta){
      (log(1/(1-p^(1/gamma))))^(1/alpha)/beta}
    if(is.null(in.p$shape) || is.null(in.p$shape2) || is.null(in.p$scale)){
      fit.expwei <- fitdistcens(data,"expwei",
                              start=list(alpha=1,gamma=1,beta=1))
      shape1.expwei <- unname(fit.expwei$estimate[1])
      shape2.expwei <- unname(fit.expwei$estimate[2])
      scale.expwei <- unname(fit.expwei$estimate[3])
    }
    else{
      shape1.expwei <- in.p$shape
      shape2.expwei <- in.p$shape2
      scale.expwei <- in.p$scale
    }
    theor.PP <- pexpwei(t, shape1.expwei, shape2.expwei, scale.expwei)
    theor.QQ <- qexpwei(1-surv.value, shape1.expwei,
                      shape2.expwei, scale.expwei)
    out.p <- list(distr = "Exponetiated␣Weibull", shape = shape1.expwei,
                 shape2 = shape2.expwei, scale = scale.expwei)
  }
  # Exponential power
  if(distribution=="exppower"){
    dexppow <<- function(x,alpha,beta){
      alpha*beta^alpha*x^(alpha-1)*
        exp((beta*x)^alpha)*exp(1-exp((beta*x)^alpha))}
    pexppow <<- function(q,alpha,beta) 1-exp(1-exp((beta*q)^alpha))
    qexppow <<- function(p,alpha,beta) (log(1-log(1-p)))^(1/alpha)/beta
    if(is.null(in.p$shape) || is.null(in.p$scale)){
      fit.exppow <- fitdistcens(data,"exppow",start=list(alpha=0.5,beta=0.5))
      shape.exppow <- unname(fit.exppow$estimate[1])
      scale.exppow <- unname(fit.exppow$estimate[2])
    }
    else{
      shape.exppow <- in.p$shape
      scale.exppow <- in.p$scale
    }
    theor.PP <- pexppow(t, shape.exppow, scale.exppow)
    theor.QQ <- qexppow(1-surv.value, shape.exppow, scale.exppow)
    out.p <- list(distr = "Exponential␣Power", shape = shape.exppow,
                 scale = scale.exppow)
  }
  # building the data
  index <- 0
  for (i in 1:length(u.point)){
    while (theor.QQ[i]>empirical_f[1,index+1]) index <- index + 1
    if(index!=0) u.estimate[i] <- empirical_f[2,index]
  }
  ggdat <- data.frame(PPx=1-surv.value, PPy=theor.PP, QQx=t, QQy=theor.QQ, SPx=2/
      pi*asin(sqrt(1-surv.value)), SPy=2/pi*asin(sqrt(theor.PP)), ERx=u.point, ERy
      =u.estimate)
  # plots
  PP <- ggplot(data=ggdat, aes(x=PPy, y=PPx)) + geom_point(colour="red") + xlab(
      expression(hat(F)(t))) + ylab(expression(hat(F)[0](t))) + geom_abline(
      intercept=0) + annotate("text", label="P-P", x=Inf, y=-Inf, hjust=1, vjust
      =-1, size=6, fontface="bold", family="URWBookman")
  QQ <- ggplot(data=ggdat, aes(x=QQx, y=QQy)) + geom_point(colour="#0073B1") +
      xlab(expression(t)) + ylab(expression(paste(hat(F)[0]^{-1})( hat(F)(t)) )) +
       geom_abline(intercept = min(t)) + annotate("text", label="Q-Q", x=Inf, y=-
      Inf, hjust=1, vjust=-1, size=6, fontface="bold", family="URWBookman")
  SP <- ggplot(data=ggdat, aes(x=SPx, y=SPy)) + geom_point(colour="#E69F00") +
      xlab(expression(paste(2/pi %*% arcsin(hat(F)(t)^{1/2})))) + ylab(expression(
      paste(2/pi %*% arcsin(hat(F)[0](t)^{1/2})))) + geom_abline(intercept=0) +
      annotate("text", label="SP", x=Inf, y=-Inf, hjust=1, vjust=-1, size=6,
      fontface="bold", family="URWBookman")
  ER <- ggplot(data=ggdat, aes(x=ERx, y=ERy)) + geom_point(colour="#006400") +
      xlab(expression(hat(F)[u](t))) + ylab(expression(hat(F)[u](paste(hat(F)
      [0]^{-1})(hat(F)(t))))) + geom_abline(intercept=0) + annotate("text", label=
      "ER", x=Inf, y=-Inf, hjust=1, vjust=-1, size=6, fontface="bold", family="
      URWBookman")
  grid.arrange(PP, QQ, SP, ER, ncol=2, top=paste("Probability␣plots␣for␣a", out.p
      $distr, "distribution"), bottom=paste("Shape␣or␣location:", round(out.p$
      shape, 3), "\t", "Scale:", round(out.p$scale, 3)))
}
```

A.2 cum.haz.plot

```
cum.haz.plot <- function(time, cens, beta.limits=c(0,1)){
  # Transform the input data to the needed format
  n <- length(time)
  data <- data.frame(left=time, right=ifelse(cens==1,time,NA))
  survNA <- survfit(Surv(time, cens)~1, type='fleming')
  # Compute the Cumulative Hazard
  Haz <- round(with(summary(survNA), -log(surv)), 6)
  # Compute the event times
  t<-summary(survNA)$time
  # Compute the survival at the event times
  surv.value<-summary(survNA)$surv
  # Computing parameters
  parameters <- list()
  # some graphical definitions
  blank <- theme(axis.title.y=element_blank())
  outrange <- ggplot(data=data.frame(y=1, x=1)) + coord_flip(ylim = c(0, 1), xlim
       = c(0, 1)) + annotate("text", x=0.5, y=0.5, label="The␣data␣is␣out␣of␣range
      ") + theme(axis.title.y=element_blank())
  # functions
  # gumbel
  dgumbel <<- function(x,location,scale){
    1/scale*exp((x-location)/scale)*exp(-exp((x-location)/scale))
  }
  pgumbel <<- function(q,location,scale){ 1-exp(-exp((q-location)/scale)) }
  # log-logis
  dloglogis <<- function(x,shape,scale){
    (shape*scale^(-shape)*x^(shape-1))/(1+(x/scale)^shape)^2
  }
  ploglogis <<- function(q,shape,scale) 1/(1+(q/scale)^(-shape))
  # beta
  a.beta<-beta.limits[1]
  b.beta<-beta.limits[2]
  # exponentiated weibull
  dexpwei <<- function(x,shape1,shape2,scale){
    shape2*shape1*scale^shape1*x^(shape1-1)*exp(-(scale*x)^shape1)*(1-exp(-(scale*
        x)^shape1))^(shape2-1)
  }
  pexpwei <<- function(q,shape1,shape2,scale){ (1-exp(-(scale*q)^shape1))^shape2
      }
  # exponential power
  dexppow <<- function(x,shape,scale){
    shape*scale^shape*x^(shape-1)*exp((scale*x)^shape)*exp(1-exp((scale*x)^shape))
  }
  pexppow <<- function(q,shape,scale) 1-exp(1-exp((scale*q)^shape))
  # distributions with the following condition, if(min(data[,1]) < 0): exp-
```

```
    weibull, exppower, log-logis, log-normal and weibull (5/9)
# distributions without the condition: normal, logistic and gumbel
# distributions with if(max(data[,1])>b.beta || min(data[,1])<a.beta): beta
if(min(data[,1]) < 0) {
  weibull <- outrange + ggtitle("Weibull")
  lognormal <- outrange + ggtitle("lognormal")
  loglogis <- outrange + ggtitle("Log-Logistic")
  expweibull <- outrange + ggtitle("Exponentiated␣Weibull")
  exppower <- outrange + ggtitle("Exponential␣Power")
} else {
  # weibull
  fit.wei <- fitdistcens(data,"weibull")
  shape.wei <-fit.wei$estimate[1]
  scale.wei <- fit.wei$estimate[2]
  trans.wei <- function(Haz) log(Haz)
  parameters$weibull <- c(shape.wei, scale.wei)
  reg.wei <- function(t) shape.wei*(-log(scale.wei)+log(t))
  weibull <- qplot(x=log(t), y=trans.wei(Haz)) + geom_line(aes(x=log(t), y=reg.
     norm(t))) + ggtitle("Weibull") + blank
  # log-normal
  fit.lnorm <- fitdistcens(data, "lnorm")
  loc.lnorm <- fit.lnorm$estimate[1]
  names(loc.lnorm) <- "location"
  scale.lnorm <- fit.lnorm$estimate[2]
  names(scale.lnorm) <- "scale"
  trans.lnorm <- function(Haz) qnorm(1-exp(-Haz))
  parameters$lognormal <- c(loc.lnorm, scale.lnorm)
  reg.lnorm <- function(t) (log(t)-loc.lnorm)/scale.lnorm
  lognormal <- qplot(x=log(t), y=trans.lnorm(Haz)) + geom_line(aes(x=log(t), y=
     reg.lnorm(t))) + ggtitle("Log-Normal") + blank
  # log-logistic
  fit.loglog <- fitdistcens(data,"loglogis", start=list(shape=1,scale=1))
  shape.loglogis <- fit.loglog$estimate[1]
  scale.loglogis <- fit.loglog$estimate[2]
  trans.loglogis <- function(Haz) log(exp(Haz)-1)
  parameters$loglogistic <- c(shape.loglogis, scale.loglogis)
  reg.loglogis <- function(t) shape.loglogis*(log(t)-log(scale.loglogis))
  loglogis <- qplot(x=log(t), y=trans.loglogis(Haz)) + geom_line(aes(x=log(t), y
     =reg.loglogis(t))) + ggtitle("Log-Logistic") + blank
  # exponentiated weibull
  fit.expwei <- fitdistcens(data, "expwei", start=list(shape1=0.01,shape2=0.01,
     scale=0.01))
  shape1.expwei <- fit.expwei$estimate[1]
  shape2.expwei <- fit.expwei$estimate[2]
  scale.expwei <- fit.expwei$estimate[3]
  trans.expwei <- function(Haz){
    log(-log(1-(1-exp(-Haz))^(1/shape2.expwei)))}
  parameters$expweibull <- c(shape1.expwei, shape2.expwei, scale.expwei)
  reg.expwei <- function(t) shape1.expwei*log(scale.expwei*t)
  expweibull <- qplot(x=log(t), y=trans.expwei(Haz)) + geom_line(aes(x=log(t), y
     =reg.expwei(t))) + ggtitle("Exponentiated␣Weibull") + blank
  # exponential power: 0.000000005
  fit.exppow <- fitdistcens(data,"exppow", start=list(shape=0.000000005,scale
     =0.000000005))
  shape.exppow <- fit.exppow$estimate[1]
  scale.exppow <- fit.exppow$estimate[2]
  trans.exppow <- function(Haz) log(log(Haz+1))
  parameters$exppower <- c(shape.exppow, scale.exppow)
  reg.exppow <- function(t) shape.exppow*log(scale.exppow*t)
  exppower <- qplot(x=log(t), y=trans.exppow(Haz)) + geom_line(aes(x=log(t), y=
     reg.exppow(t))) + ggtitle("Exponentiated␣power") + blank
}
# the three distributions that not depends of the conditions
# gumbel
fit.gum <- fitdistcens(data,"gumbel", start=list(location=0,scale=3000))
shape.gum <- fit.gum$estimate[1]
scale.gum <- fit.gum$estimate[2]
trans.gum <- function(Haz) log(Haz)
parameters$gumbel <- c(shape.gum, scale.gum)
reg.gum <- function(t) (t-shape.gum)/scale.gum
gumbel <<- qplot(x=t, y=trans.gum(Haz)) + geom_line(aes(x=t, y=reg.gum(t))) +
   ggtitle("Gumbel") + blank
# normal
fit.norm <- fitdistcens(data,"norm")
loc.norm<-fit.norm$estimate[1]
names(loc.norm) <- "location"
scale.norm <- fit.norm$estimate[2]
names(scale.norm) <- "scale"
trans.norm <- function(Haz) qnorm(1-exp(-Haz))
parameters$normal <- c(loc.norm, scale.norm)
reg.norm <- function(t) (t-loc.norm)/scale.norm
normal <- qplot(x=t, y=trans.norm(Haz)) + geom_line(aes(x=t, y=reg.norm(t))) +
   ggtitle("Normal") + blank
# logistic
fit.log <- fitdistcens(data,"logis")
loc.logis <- fit.log$estimate[1]
scale.logis <- fit.log$estimate[2]
trans.logis <- function(Haz) log(exp(Haz)-1)
parameters$logistic <- c(loc.logis, scale.logis)
reg.logis <- function(t) (t-loc.logis)/scale.logis
logistic <- qplot(x=t, y=trans.logis(Haz)) + geom_line(aes(x=t, y=reg.logis(t))
   ) + ggtitle("Logistic") + blank
# beta distribution
if(max(data[,1])>b.beta || min(data[,1])<a.beta) {
    betap <- outrange + ggtitle("Beta")
  } else {
    fit.beta <- fitdistcens((data-a.beta)/(b.beta-a.beta),"beta")
    shape1.beta <- fit.beta$estimate[1]
    shape2.beta <- fit.beta$estimate[2]
    trans.beta <- function(Haz) qbeta(1-exp(-Haz),shape1.beta,shape2.beta)
    parameters$beta <- list(param = c(shape1.beta, shape2.beta), domain = beta.
       limits)
    reg.beta <- function(t) (t-a.beta)/(b.beta-a.beta)
    betap <- qplot(x=t, y=trans.beta(Haz)) + geom_line(aes(x=t, y=reg.beta(t))) +
       ggtitle("Beta") + blank
  }
  # a grid of plots
  grid.arrange(weibull, gumbel, normal, lognormal, logistic, loglogis, betap,
     expweibull, exppower, ncol=3)
  parameters
}
```

Appendix B

DISTRIBUTIONS

In the Chapter 1 we already explored the basic mechanisms of the distributions applied to survival analysis, so we'll talk about the most used distributions, Weibull, Exponential, Gumbel, normal, log-normal, logistic, log-logistic, beta, exponential power and exponentiated Weibull, very suitable for most of our survival analyses from a very practical approach, with the R software available. The purpose is to know better the mechanisms behind our R package, with the numerical and graphical methods for goodness of fit, reviewed in detail in the Chapter 2.

We'll denote the shape parameters as α and γ, the location parameter as μ and the scale parameter as β, following the criterion of Febrer Galvany (2015). Now, we'll show two tables with the distributions mentioned, one with the density functions and its survival functions and the other with the cumulative hazard and hazard functions.

B.1 Density and survival functions

Distribution	Density Function $[f(t)]$	Survival Function $[S(t)]$
Weibull	$\alpha\beta^\alpha t^{\alpha-1}e^{-(\beta t)^\alpha}$	$e^{-(\beta t)^\alpha}$
Exponential	$\beta e^{\beta t}$	$e^{-\beta t}$
Gumbel	$\frac{1}{\beta}e^{\frac{t-\mu}{\beta}}e^{-e^{\frac{t-\mu}{\beta}}}$	$e^{-e^{\frac{t-\mu}{\beta}}}$
Normal	$\frac{1}{\beta\sqrt{2\pi}}e^{-\frac{(x-\mu)^2}{2\beta^2}}$	$\int_t^\infty \frac{1}{\beta\sqrt{2\pi}}e^{-\frac{(x-\mu)^2}{2\beta^2}}dx$
Log-normal	$\frac{1}{\beta t\sqrt{2\pi}}\exp\left(-\frac{[\log t-\mu^2]}{2\beta^2}\right)$	$1-\phi\left(\frac{\log t-\mu}{\beta}\right)=\int_{\frac{\log t-\mu}{\beta}}^\infty \frac{1}{2\pi}e^{-\frac{x^2}{2}}dx$
Logistic	$\frac{e^{-\frac{t-\mu}{\beta}}}{\beta(1+e^{-\frac{t-\mu}{\beta}})^2}$	$\frac{e^{-\frac{t-\mu}{\beta}}}{1+e^{-\frac{t-\mu}{\beta}}}$
Log-logistic	$\frac{\alpha t^{\alpha-1}\beta^{-\alpha}}{\left[1+\left(\frac{t}{\beta}\right)^\alpha\right]^2}$	$\frac{1}{1+\left(\frac{t}{\beta}\right)^\alpha}$
Beta	$\frac{1}{B(\alpha,\gamma)}\frac{(t-a)^{\alpha-1}(b-t)^{\gamma-1}}{(b-a)^{\alpha+\gamma-1}}$	$\frac{B(\alpha,\gamma)-B_{\frac{t-a}{b-a}}(\alpha,\gamma)}{B(\alpha,\gamma)}$
Exp-Power	$\alpha\beta^\alpha t^{\alpha-1}e^{(\beta t)^\alpha}e^{1-e^{(\beta t)^\alpha}}$	$e^{1-e^{(\beta t)^\alpha}}$
Exp-Weibull	$\gamma\alpha\beta^\alpha t^{\alpha-1}e^{-(\beta t)^\alpha}[1-e^{-(\beta t)^\alpha}]^{\gamma-1}$	$1-[1-e^{-(\beta t)^\alpha}]^\gamma$

Table B.1: Survival and density functions

B.2 Cumulative hazard and hazard functions

Distribution	Cumulative Hazard Function [$\Lambda(t)$]	Hazard Function [$\lambda(t)$]
Weibull	$(\beta t)^\alpha$	$\alpha\beta^\alpha t^{\alpha-1}$
Exponential	βt	β
Gumbel	$e^{\frac{t-\mu}{\beta}}$	$\frac{1}{\beta}e^{\frac{t-\mu}{\beta}}$
Normal	$-\log\left(1-\phi\left(\frac{t-\mu}{\beta}\right)\right)$	$\frac{1}{\beta\sqrt{2\pi}} \times \frac{e^{-\frac{(t-\mu)^2}{2\beta^2}}}{1-\phi(\frac{t-\mu}{\beta})}$
Log-normal	$-\log\left(1-\phi\left(\frac{t-\mu}{\beta}\right)\right)$	$\frac{\frac{1}{\beta t\sqrt{2\pi}}e^{-\frac{(\log t-\mu)^2}{2\beta^2}}}{1-\phi\left(\frac{\log t-\mu}{\beta}\right)}$
Logistic	$\log\left(1+e^{\frac{t-\mu}{\beta}}\right)$	$\frac{e^{\frac{t-\mu}{\beta}}}{\beta\left(1+e^{\frac{t-\mu}{\beta}}\right)}$
Log-logistic	$\log\left[1+\left(\frac{t}{\beta}\right)^\alpha\right]$	$\frac{\alpha\beta^{-\alpha}t^{\alpha-1}}{1+(\frac{t}{\beta})^\alpha}$
Beta	$-\log\left(\frac{B(\alpha,\gamma)-B_{\frac{t-a}{b-a}}(\alpha,\gamma)}{B(\alpha,\gamma)}\right)$	$\frac{[(t-a)^{\alpha-1}(b-t)^{\gamma-1}]/(b-a)^{\alpha+\gamma-1}}{B(\alpha,\gamma)-B_t(\alpha,\gamma)}$
Exp-Power	$e^{(\beta t)^\alpha}-1$	$\alpha\beta^\alpha t^{\alpha-1}e^{(\beta t)^\alpha}$
Exp-Weibull	$-\log(1-[1-e^{-(\beta t)^\alpha}]^\gamma)$	$\frac{\alpha\beta\gamma(\beta t)^{\alpha-1}[1-e^{-(\beta t)^\alpha}]^{\gamma-1}e^{-(\beta t)^\alpha}}{1-[1-e^{-(\beta t)^\alpha}]^\gamma}$

Table B.2: Hazard functions

REFERENCES

Auguie, B. (2016). gridextra: Miscellaneous functions for "grid" graphics [Computer software manual]. Retrieved from `https://CRAN.R-project.org/package=gridExtra` (R package version 2.2.1)

Baar, T., Donald R. & Davidson. (1973). A Kolmogorov-Smirnov Test for Censored Samples. *Technometrics*, *15*(4), 739-757. Retrieved from `http://www.tandfonline.com/doi/abs/10.1080/00401706.1973.10489108?src=recsys` doi: 10.1080/00401706.1973.10489108

Besalú i Mayol, M. (2016). *Master's degree thesis: Testing Goodness-of-Fit of Parametric Survival Models for Right-Censored Data*.

Broström, G. (2012). *Event History Analysis with R*. CRC Press.

David W. Hosmer, S. M., Stanley Lemeshow. (2008). *Applied Survival Analysis: Regression Modeling of Time-to-Event Data* (2nd ed.).

Delignette-Muller, M. L., & Dutang, C. (2015a). fitdistrplus: An R package for fitting distributions. *Journal of Statistical Software*, *64*(4), 1–34. Retrieved from `http://www.jstatsoft.org/v64/i04/`

Delignette-Muller, M. L., & Dutang, C. (2015b). fitdistrplus: An R package for fitting distributions. *Journal of Statistical Software*, *64*(4), 1–34. Retrieved from `http://www.jstatsoft.org/v64/i04/`

Dufour, U., R & Maag. (1978). Distribution Results for Modified Kolmogorov-Smirnov Statistics for Truncated or Censored. *Technometrics*, *20*(1), 29-32. Retrieved from `http://www.tandfonline.com/doi/abs/10.1080/00401706.1978.10489613` doi: 10.1080/00401706.1978.10489613

Febrer Galvany, A. (2015). *Master's degree thesis: Analytical and Graphical Goodness of Fit Methods for Parametric Survival Models with Right-censored Data*.

Grané, A. (2012). Exact goodness-of-fit tests for censored data. *Annals of the Institute of Statistical Mathematics*, *64*(6), 1187–1203. doi: 10.1007/s10463-012-0356-y

Kolmogorov, A. N. (1933). Sulla Determinazione Empirica di una Legge di Distribuzione. *Giornale dell'Istituto Italiano degli Attuari*, *4*, 83–91.

Mersmann, O. (2015). microbenchmark: Accurate Timing Functions [Computer software manual]. Retrieved from `https://CRAN.R-project.org/package=microbenchmark` (R package version 1.4-2.1)

Michael, J. R. (1983). The stabilized probability plot. *Biometrika*, *70*(1), 11. Retrieved from `+ http://dx.doi.org/10.1093/biomet/70.1.11` doi: 10.1093/biomet/70.1.11

Murrell, P. (2012). *R graphics* (2nd ed ed.). CRC Press.

R Core Team. (2017). R: A Language and Environment for Statistical Computing [Computer software manual]. Vienna, Austria. Retrieved from https://www.R-project.org/

Ralph B. D'Agostino, M. A. S. (1986). *Goodness-of-Fit Techniques*. Marcel Dekker.

Rohde, C. A. (2014). *Introductory Statistical Inference with the Likelihood Function* (1st ed.). Springer International Publishing.

Sauerbrei, W., & Royston, P. (1999). Building multivariable prognostic and diagnostic models: transformation of the predictors by using fractional polynomials. *Journal of the Royal Statistical Society: Series A (Statistics in Society)*, *162*(1), 71–94. Retrieved from http://dx.doi.org/10.1111/1467-985X.00122 doi: 10.1111/1467-985X.00122

Smirnov, N. (1948, 06). Table for Estimating the Goodness of Fit of Empirical Distributions. *Ann. Math. Statist.*, *19*(2), 279–281. Retrieved from http://dx.doi.org/10.1214/aoms/1177730256 doi: 10.1214/aoms/1177730256

Smirnov, N. V. (1939). Estimate of deviation between empirical distributions. *Bulletin Moscow University*, *2*, 3 – 16.

Strzalkowska-Kominiak, E., & Grané, A. (2014). Goodness-of-fit test for randomly censored data based on maximum correlation.

Therneau, T. M. (2015). A package for survival analysis in s [Computer software manual]. Retrieved from https://CRAN.R-project.org/package=survival (version 2.38)

Waller, B. W., Lance A. & Turnbull. (1992). Probability Plotting with Censored Data. *The American Statistician*, *46*(1), 5-12. doi: 10.1080/00031305.1992.10475837

Wickham, H. (2009). *ggplot2: Elegant Graphics for Data Analysis*. Springer-Verlag New York. Retrieved from http://ggplot2.org

Wickham, H. (2010). A layered grammar of graphics. *Journal of Computational and Graphical Statistics*, *19*(1), 3–28. doi: 10.1198/jcgs.2009.07098

Wickham, H. (2015). *R Packages* (1st ed.). O'Reilly Media.

Wickham, H. (2017). lineprof: An alternative display for line profiling information [Computer software manual]. (R package version 0.1.9001)

Wilkinson, L. e. a. (2005). *The grammar of graphics* (2nd ed.). Springer.